

이동로봇 루트리얼

4강

ROS2 Nav2 Simulation

목차

01 강의 개요

02 시뮬레이션 환경 구성

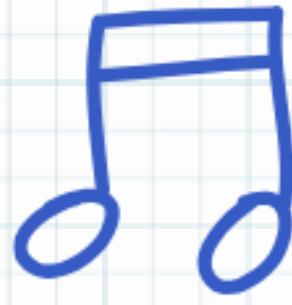
03 컨트롤러 교체 실습

04 Nav2 주행 시뮬레이션

05 시뮬레이션 과제

06 요약 및 마무리





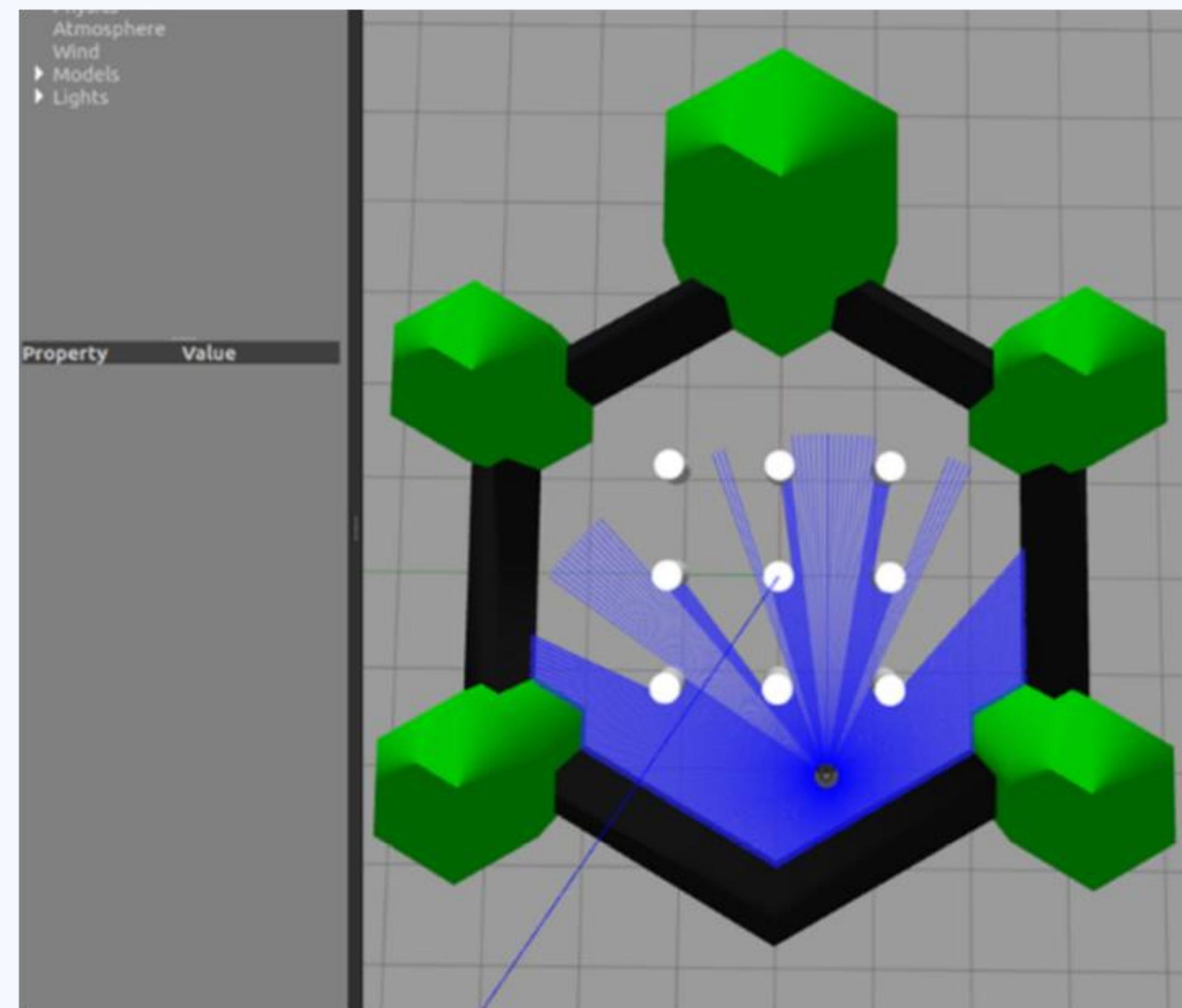
강의 개요



ROS2 Nav2 시뮬레이션 실습 소개

- 실습 환경: Ubuntu 22.04 + ROS2 Humble + Gazebo + RViz
- 로봇 플랫폼: TurtleBot3 Waffle
- 학습 목표: Nav2 bringup, 목표 주행, Controller 교체 및 튜닝
- 핵심 실습: DWB, RPP, MPPI 교체
- 동일한 지도와 파라미터 환경에서 로컬 플래너 교체
- 주행 반응성과 안정성 비교
- Unity 시뮬레이션 확장 실습

Gazebo + Nav2 시뮬레이션 환경

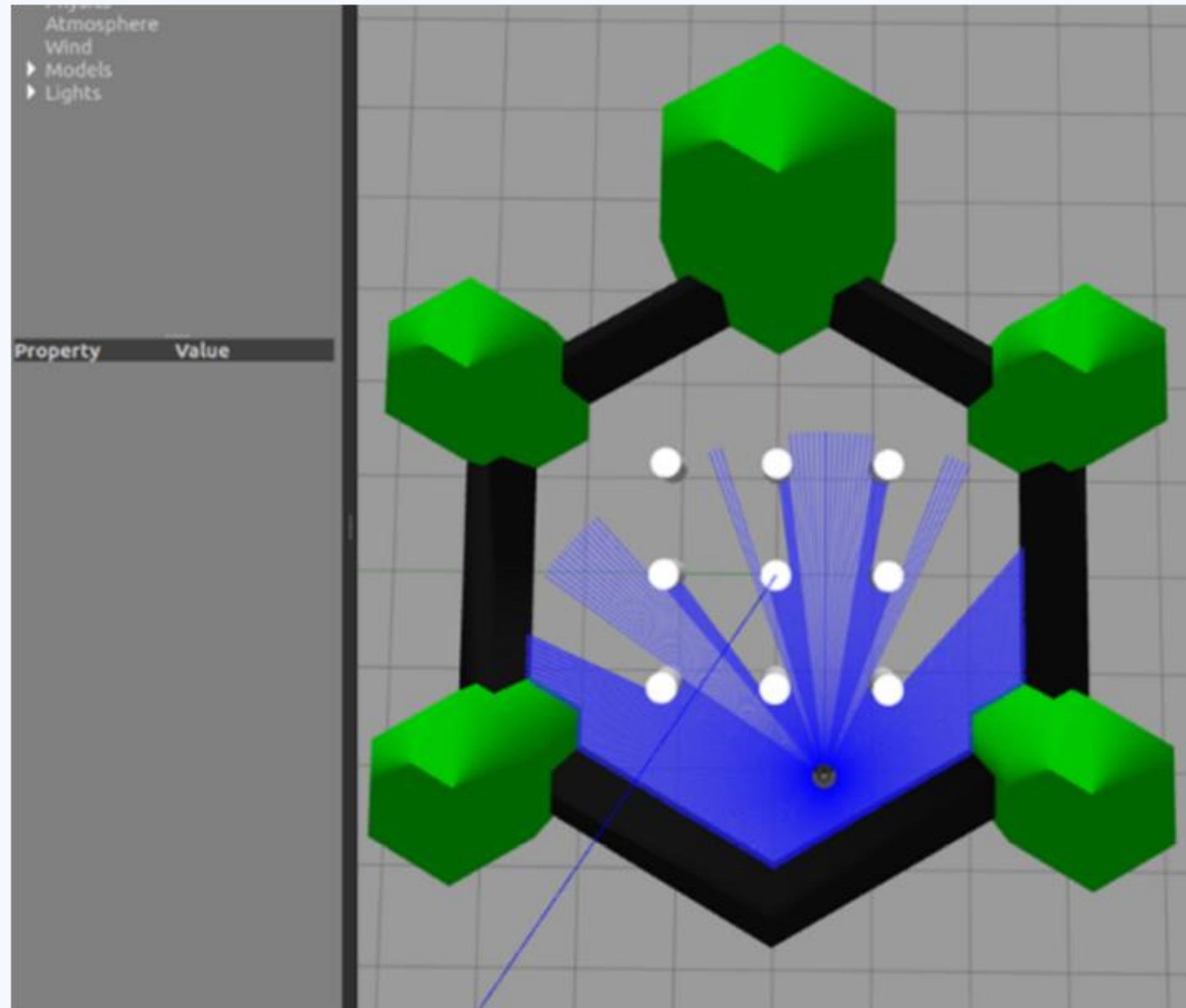


✓ 시뮬레이션 환경 설정

<https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/#gazebo-simulation>

- Gazebo 환경에서 Turtlebot3 waffle 기반 Nav2 주행
시뮬레이션

Gazebo + Nav2 시뮬레이션 환경



✓ 시뮬레이션 주행 실습

```
$ export TURTLEBOT3_MODEL=waffle
```

```
$ ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py
```

```
$ ros2 launch turtlebot3_navigation2 navigation2.launch.py
```

```
map:=/home/home/turtlebot3_ws/src/turtlebot3_navigation2/  
map/map.yaml
```

```
params_file:=/home/home/turtlebot3_ws/src/turtlebot3_navig  
ation2/param/mppi.yaml use_sim_time:=true
```

>>> mppi.yaml 부분을 rpp, dwa로 변경하며 local planner 테스트

MPPI Local Planner

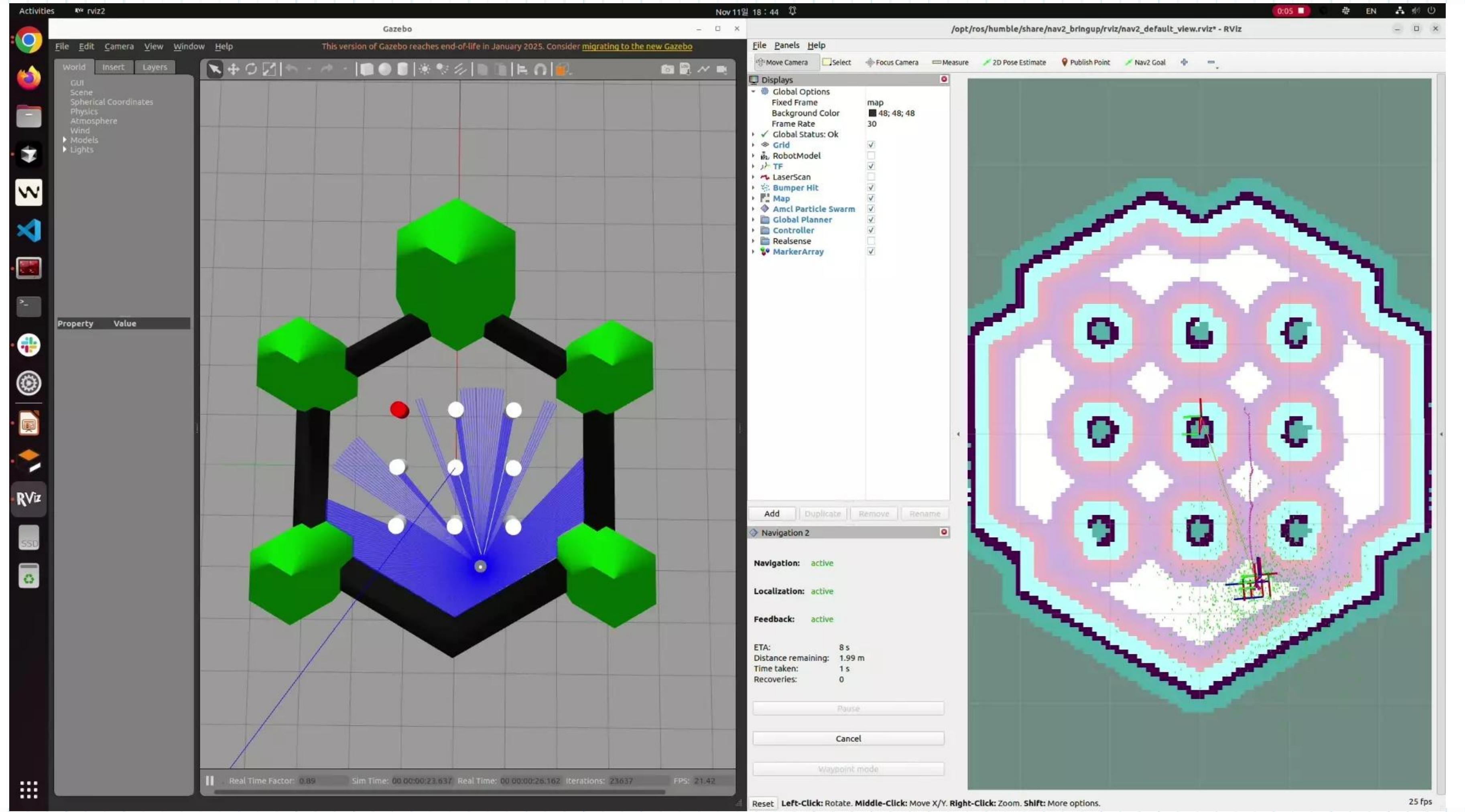
```
controller_server:  
ros_parameters:  
  controller_frequency: 20.0  
FollowPath:  
  plugin: "nav2_mppi_controller::MPPIController"  
  time_steps: 56  
  model_dt: 0.05  
  batch_size: 2000  
  vx_std: 0.2  
  vy_std: 0.2  
  wz_std: 0.4  
  vx_max: 0.5  
  vx_min: -0.35  
  vy_max: 0.5  
  wz_max: 1.9  
  ax_max: 3.0  
  ax_min: -3.0  
  ay_max: 3.0  
  ay_min: -3.0  
  az_max: 3.5  
iteration_count: 1  
prune_distance: 1.7  
transform_tolerance: 0.1  
temperature: 0.3  
gamma: 0.015  
motion_model: "DiffDrive"  
visualize: true  
reset_period: 1.0 # (only in Humble)  
regenerate_noises: false  
TrajectoryVisualizer:  
  trajectory_step: 5  
  time_step: 3  
AckermannConstraints:  
  min_turning_r: 0.2  
critics: ["ConstraintCritic", "CostCritic", "GoalCritic", "GoalAngleCritic", "PathAlignCritic", "PathFollowCritic"]  
ConstraintCritic:  
  enabled: true  
  cost_power: 1  
  cost_weight: 4.0
```

✓ mppi.yaml

- 모델 예측 기반 확률 제어
- 수백 개의 경로를 시뮬레이션 후 최소 비용 선택
- 주요 파라미터
 - batch_size: 샘플 궤적 수
 - time_steps: 예측 시간 구간
 - temperature: 탐색 확률 분포 제어



MPPI 시뮬레이션 영상



DWB Local Planner

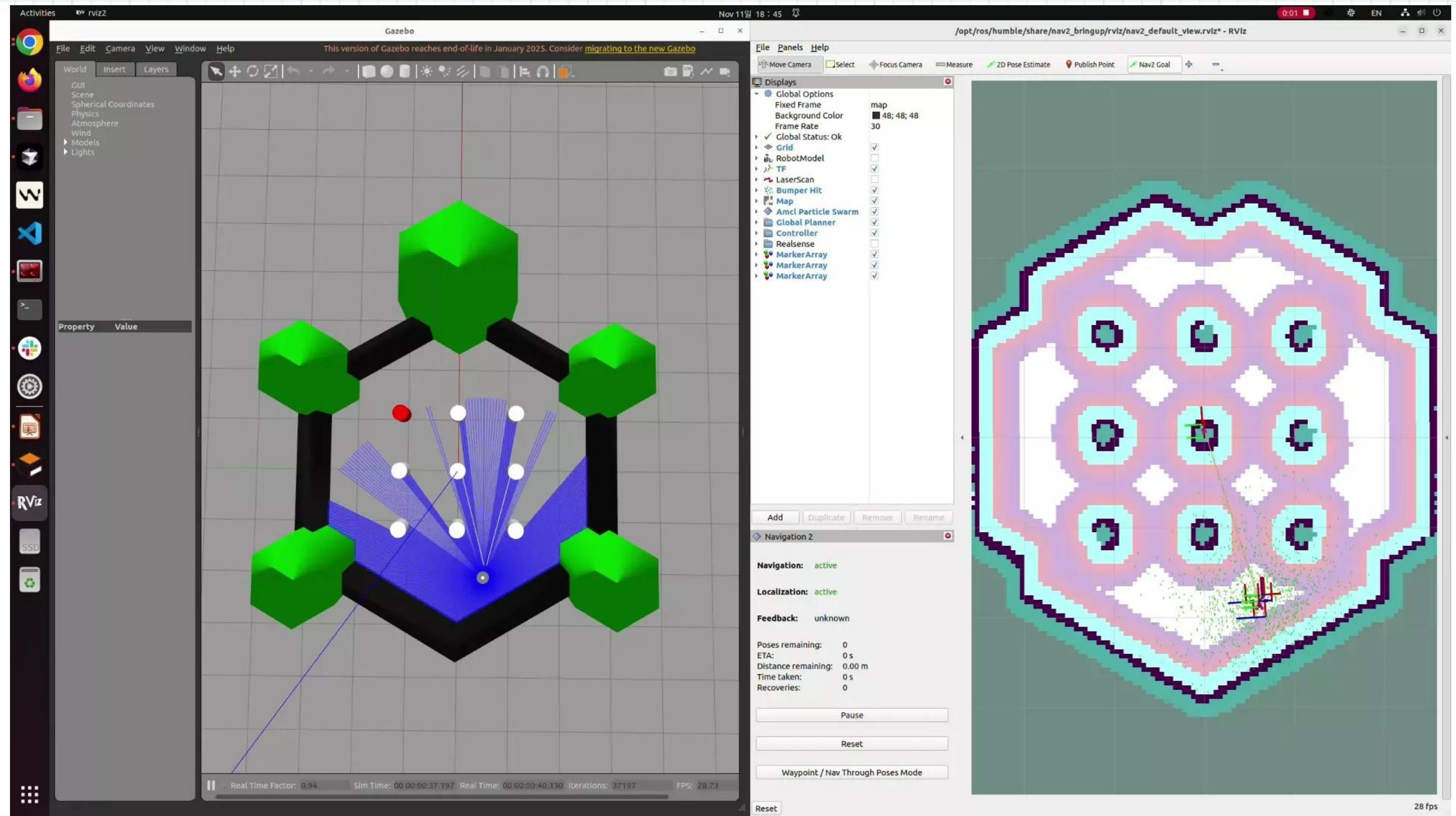
```
# DWB parameters
FollowPath:
  plugin: "dwb_core::DWBLocalPlanner"
  debug_trajectory_details: True
  min_vel_x: 0.0
  min_vel_y: 0.0
  max_vel_x: 0.26
  max_vel_y: 0.0
  max_vel_theta: 1.0
  min_speed_xy: 0.0
  max_speed_xy: 0.26
  min_speed_theta: 0.0
  # Add high threshold velocity for turtlebot 3 issue.
  # https://github.com/ROBOTIS-GIT/turtlebot3_simulations/issues/75
  acc_lim_x: 2.5
  acc_lim_y: 0.0
  acc_lim_theta: 3.2
  decel_lim_x: -2.5
  decel_lim_y: 0.0
  decel_lim_theta: -3.2
  vx_samples: 20
  vy_samples: 5
  vtheta_samples: 20
  sim_time: 1.7
  linear_granularity: 0.05
  angular_granularity: 0.025
  transform_tolerance: 0.2
  xy_goal_tolerance: 0.25
  trans_stopped_velocity: 0.25
  short_circuit_trajectory_evaluation: True
  stateful: True
  critics: ["RotateToGoal", "Oscillation", "BaseObstacle", "GoalAlign", "PathAlign", "PathDist", "GoalDist"]
  BaseObstacle.scale: 0.02
  PathAlign.scale: 32.0
  PathAlign.forward_point_distance: 0.1
```

✓ dwa.yaml

- 직선 경로에서 빠른 반응
- 장애물 회피 우수
- 급회전 시 진동 가능
- 주요 파라미터
 - sim_time: 시뮬레이션 시간
 - vx_samples: 선속도 샘플
 - path_align_scale: 경로 정렬 중요도
 - obstacle_scale: 장애물 회피 가중치



DWB 시뮬레이션 영상



RPP Local Planner

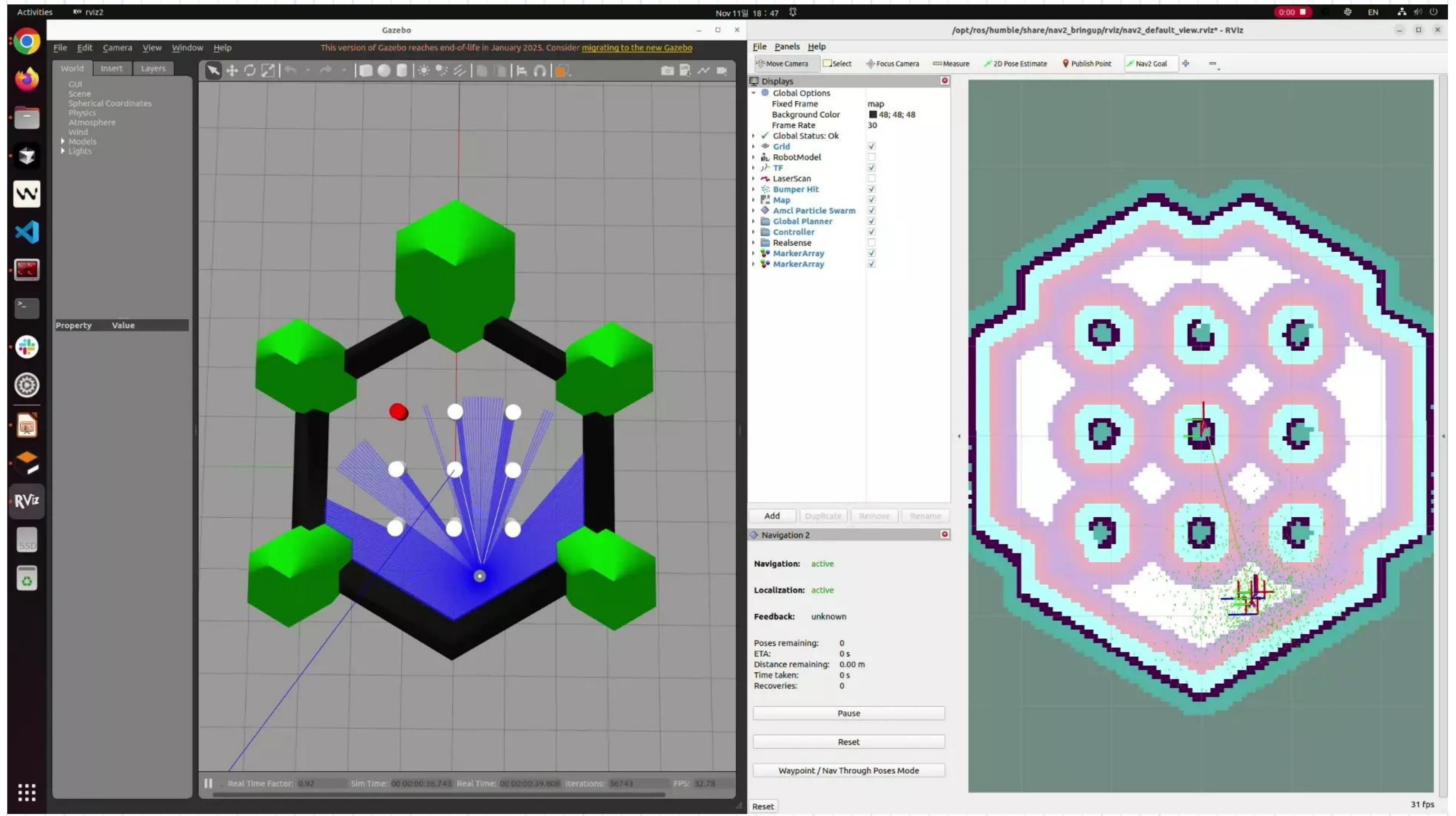
```
progress_checker:  
  plugin: "nav2_controller::SimpleProgressChecker"  
  required_movement_radius: 0.5  
  movement_time_allowance: 10.0  
  
goal_checker:  
  plugin: "nav2_controller::SimpleGoalChecker"  
  xy_goal_tolerance: 0.25  
  yaw_goal_tolerance: 0.25  
  stateful: True  
  
FollowPath:  
  plugin: "nav2_regulated_pure_pursuit_controller::RegulatedPurePursuitController"  
  desired_linear_vel: 0.5  
  lookahead_dist: 0.6  
  min_lookahead_dist: 0.3  
  max_lookahead_dist: 0.9  
  lookahead_time: 1.5  
  rotate_to_heading_angular_vel: 1.8  
  transform_tolerance: 0.1  
  use_velocity_scaled_lookahead_dist: false  
  min_approach_linear_velocity: 0.05  
  approach_velocity_scaling_dist: 0.6  
  use_collision_detection: true  
  max_allowed_time_to_collision_up_to_carrot: 1.0  
  use_regulated_linear_velocity_scaling: true  
  use_fixed_curvature_lookahead: false  
  curvature_lookahead_dist: 0.25  
  use_cost_regulated_linear_velocity_scaling: false  
  regulated_linear_scaling_min_radius: 0.9  
  regulated_linear_scaling_min_speed: 0.25  
  use_rotate_to_heading: true  
  allow_reversing: false  
  rotate_to_heading_min_angle: 0.785  
  max_angular_accel: 3.2  
  max_robot_pose_search_dist: 10.0
```

✓ rpp.yaml

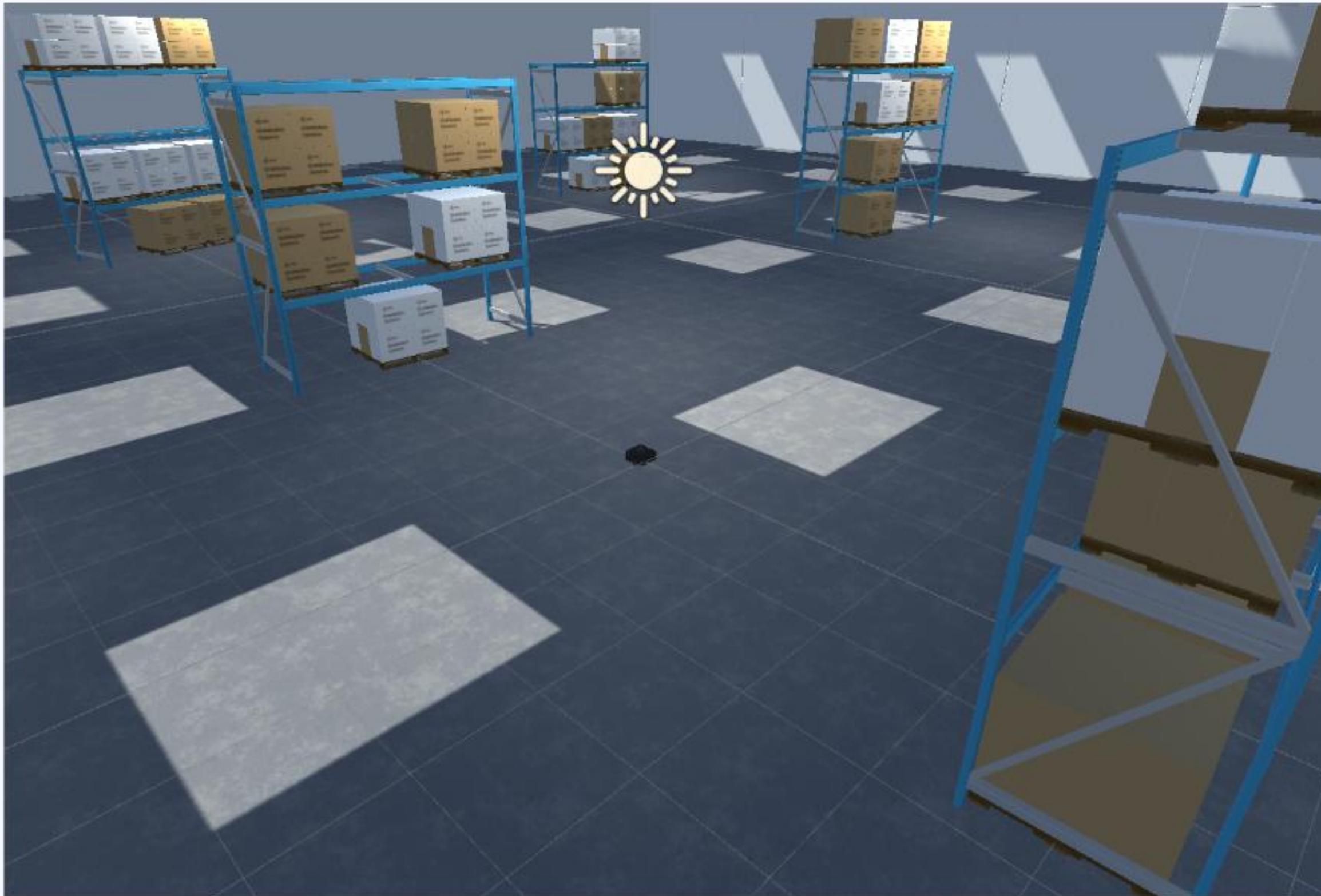
- 매우 부드러운 곡선 추종
- 장애물 근처 자동 감속
- 급선회 시 응답 늦음
- 주요 파라미터
 - lookahead_dist: 목표점 거리
 - use_cost_regulated_linear_velocity_scaling: 장애물 근접 시 감속 여부
 - min_approach_linear_velocity: 최소 접근 속도



RPP 시뮬레이션 영상



■ 실습 과제 - Unity Nav2 SLAM



✓ Unity-ROS2 연동

<https://github.com/Unity-Technologies/Robotics-Nav2-SLAM-Example?tab=readme-ov-file>

<https://unity.com/kr/blog/engine-platform/advance-your-robot-autonomy-with-ros-2-and-unity>

- Unity-ROS2 TCP Connector 설정
- Nav2 Bringup → SLAM → Goal Navigation 확인
- Gazebo와의 성능·시각화 비교

요약 및 마무리

- Gazebo 기반 TurtleBot3 + Nav2 bringup 실습을 통해 로봇 시뮬레이션 환경 구축
- AMCL 위치 초기화 및 목표 주행으로 기본 내비게이션 이해
- RPP / MPPI / DWB 컨트롤러 교체 및 비교를 통한 로컬 플래너 특성 파악
- 동일한 Nav2 구조에서 다양한 로컬 플래너를 교체하여 환경에 최적화된 주행 가능