HW1 보고서

로빛 20기 예비인턴 모시온

2025407006 로봇학부

목차

1. ROS 기본 구성

ROS의 기본 구성으로는 노드, 토픽, 메시지 및 서비스, 액션으로 구성되어 있습니다.

각 요소들은 통신 및 기동과 관련하여 역할을 담당하기 때문에 각 요소에 대해서 조사하였으며 그에 대한 결과입니다.

1. 노드

노드란 ROS에서 실행을 담당하는 프로세스로서, 지정된 단일 작업을 수행하는 요소입니다. 노드는 여러 개의 구성으로 이루어질 수 있으며 각각의 노드들은 독립적으로 실행됩니다.
각각의 노드는 맡은 역할에 따라 제어, 센서 값 수신, 상태 출력 등의 역할을 담당할 수 있습니다.

2. 토픽

토픽은 노드들간의 데이터를 교환하는 요소입니다. 토픽은 발행 – 구독의 구성으로서 발행 노드에서 전송하는 정보를 구독 노드로 전달합니다. 이때, 토픽은 지정 이름에 따라 하나의 발행 노드의 정보를 여러 개의 구독 노드에 전달할 수 있게 됩니다.

3. 메시지

메시지는 토픽의 교환을 통해 송수신되어지는 정보들의 형식을 지정합니다. 지정된 형식에 따라 각 노드들이 메시지를 읽어 정보를 해석하고 처리하도록 유도됩니다.

4. 서비스

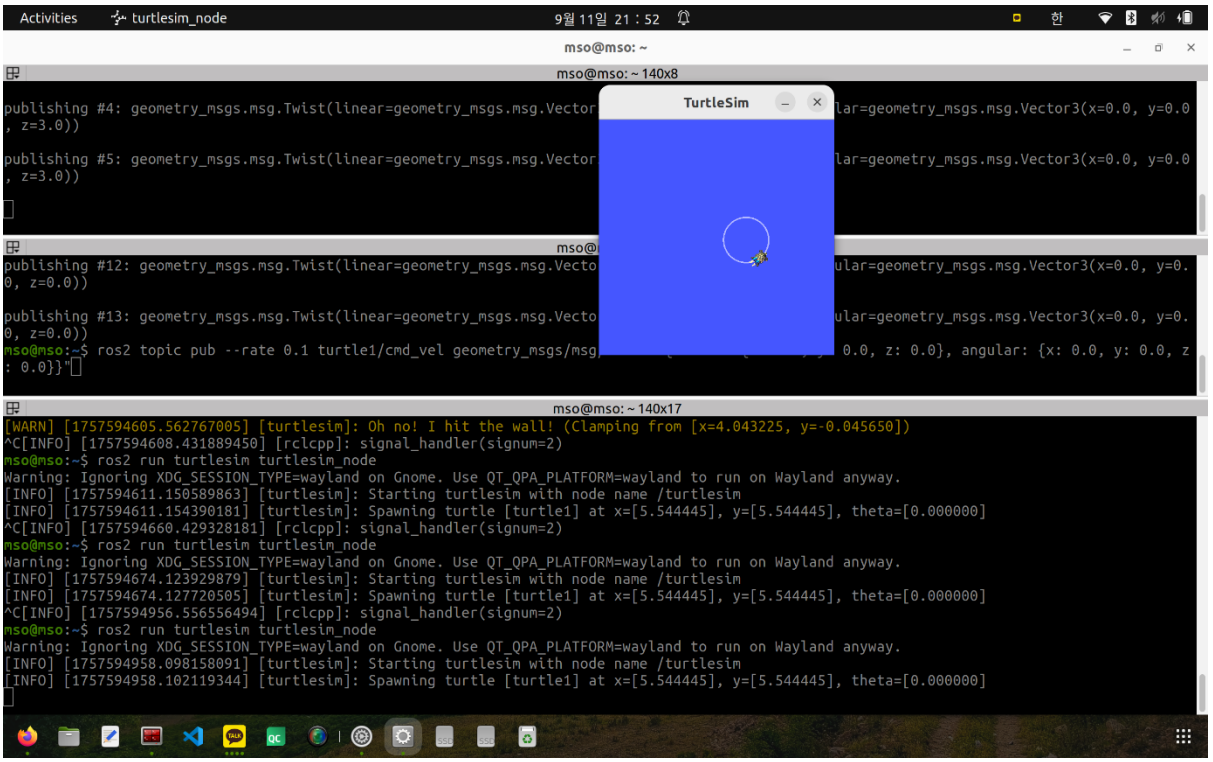
서비스 또한 노드들간의 통신을 보조합니다. 이때 요청-응답 관계의 통신을 지원하며, 특정 작업을 요청 후 그에 대한 응답(결과)을 받는 작업에 활용됩니다. 이때, 응답하는 노드를 서비스 서버, 요청하는 노드를 서비스 클라이언트라고 합니다.

5. 액션

액션은 서비스와 유사한 구조를 지니지만, 요청에 대한 응답을 처리하는 서비스와 달리, 추가적으로 응답을 기다리는 동안의 작업을 수행하는 역할을 담당합니다. 그에 대한 예시로, 정보를 요청하여 받아드리는 시간이 길 때, 정보를 다 수신받는(응답)동안 처리할 현재 상태 표시(다운로드경과 등등)의 작업을 처리하는 데에 활용됩니다.

2.실행 코드 및 결과

이러한 ROS의 기본 구성을 활용하여 ROS에서 지원하는 기본 시뮬레이터 환경, TurtleSim에서 토픽을 활용한 통신을 알아보았습니다.



ros2 topic pub –rate 1 /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 2.0}}"

해당 코드의 대한 설명입니다.

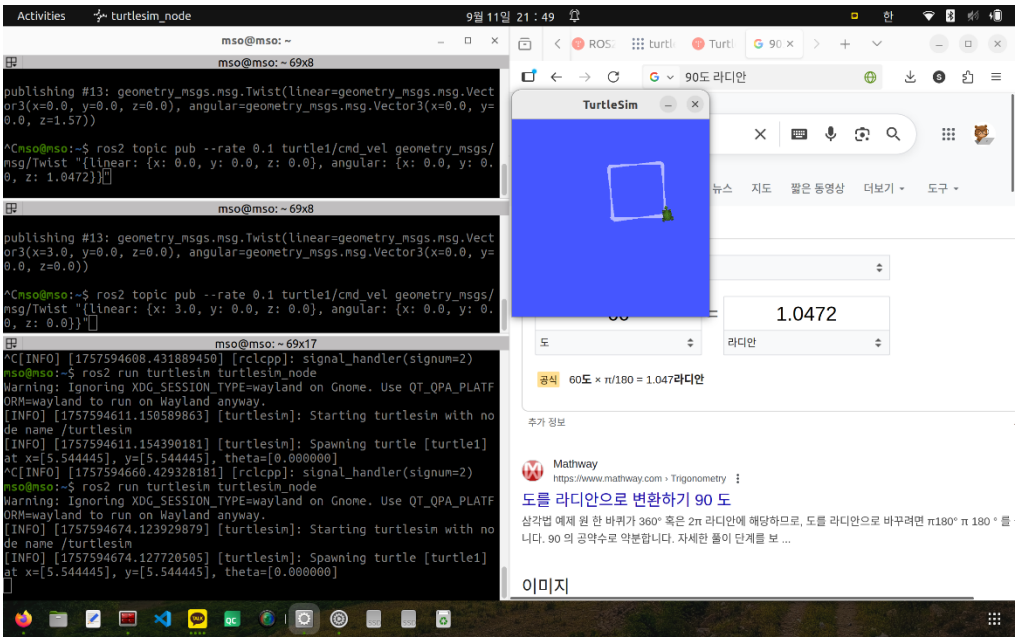
ros2 topic pub <topic_name> <msg_type> "<args>"

topic pub를 통해 발행 노드를 지정하여 전달할 토픽을 해당

<topic_name>으로 명명하여 전달하게 됩니다.
현재 topic_name은 /cmd_vel이며 해당 토픽 이름을 통해 args(arguments)인 "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 2.0}}"를 보내게 되면 msg_type(메시지 종류)로서 지정한 geometry_msgs/msg/Twist를 거쳐 해당 args를 2차원 공간 상의 움직임으로서 정의하여 동작을 이루게 됩니다.

이때, args의 내용을 살펴보면
"{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 2.0}}"
x의 값을 2만큼 변화함과 동시에 z평면 상의 값을 2만큼 변경하여 각속도를 지정해 원을 그릴 수 있게 하는 것입니다.
하지만, angular로 정의 되는 실수 값은 라디안 값으로서 π에 따라 약간의 오차가 누적되는 양상을 보입니다.
(%%이때 –once가 아닌 –rate 1를 통해 1초마다 한 번씩 동작하도록 하여 완전한 원이 그려질 수 있도록 하였습니다.)

이와 관련하여 다른 예제를 실행하였습니다.

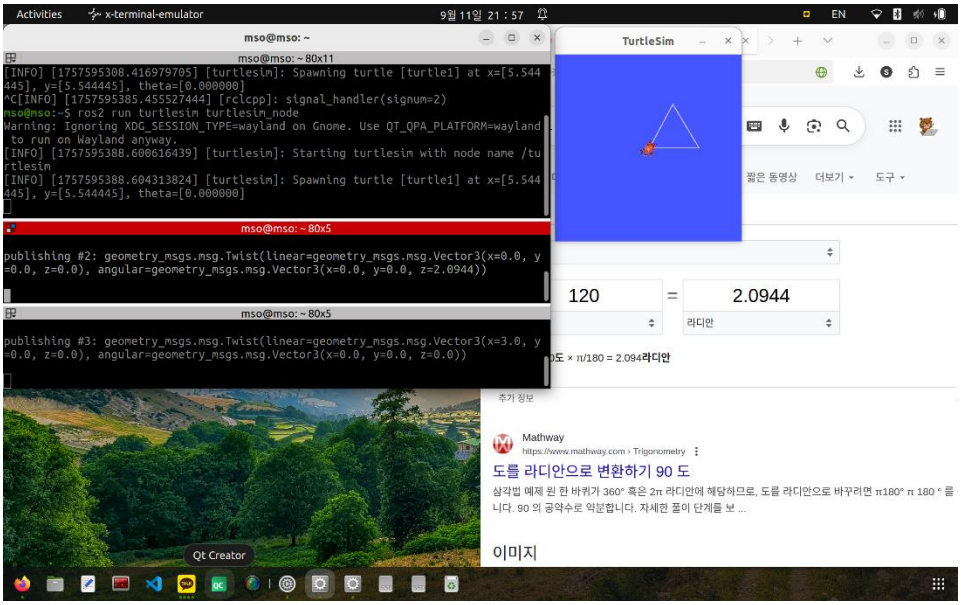


ros2 topic pub --rate 0.1 turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.57}}"
와

ros2 topic pub --rate 0.1 turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 3.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"
의 동시 실행
해당 두 코드의 실행 간격은 0.1로서 10초에 한번씩 동작하도록 하였습니다.(직선 운동 후 약 90도 회전)

첫번째 실행 코드는 10초에 한 번씩 왼쪽으로 90도 회전을, 시간 간격에 맞추어 두번째 코드를 실행할 시 x값을 3만큼 움직여 앞으로 나아가도록 하여 위 두 과정을 일련의 반복을 통해 사각형을 그리도록 하였습니다.

이를 통해 알 수 있듯이, turtlesim_node는 구독 노드이면서 동시에, 서로 다른 두 발행 노드의 값을 전달 받을 수 있다는 것입니다.
(각각, 직선 운동과 90도 회전 토픽)



ros2 topic pub --rate 0.1 turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 2.0944}}" 및
ros2 topic pub --rate 0.1 turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 3.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"
(직선 운동 후 왼쪽으로 120도 회전입니다.)

3. 참조

한국항공우주연구원 2021 우주소프트웨어 경진대회
https://blog.naver.com/PostView.naver?blogId=tvkari&logNo=222340539893&parentCategoryNo=&categoryNo=1&viewDate=&isShowPopularPosts=false&from=postView