

ROS 2 - publikowanie danych



Konfiguracja środowiska

Przed rozpoczęciem pracy z ROS, w każdym nowo otwartym terminalu należy wywołać komendę:

- po pierwszej kompilacji (istnieją katalogi `build`, `install`, `log`):

```
source install/setup.bash
```

- przed pierwszą kompilacją:

```
source /opt/ros/humble/setup.bash
```

Wprowadzenie

W tej instrukcji będziemy publikować dane z wykorzystaniem tematów (*topics*). Będziemy używać różnych robotów: `turtlesim` i symulowanego ramienia robotycznego.

Informacje na temat tematów w ROS oraz `turtlesim` znajdują się [tutaj](#).

Publikowanie w kontekście `turtlesim`

Uruchom symulację:

```
ros2 run turtlesim turtlesim_node
```

Przetestuj działanie symulacji uruchamiając w osobnym terminalu węzeł `turtle_teleop_key`:

```
ros2 run turtlesim turtle_teleop_key
```

Obejrzyj działające węzły oraz połączenia między nimi za pomocą narzędzia `rqt_graph`.

Naszym robozółwem możemy sterować przy pomocy tematów. Użyj odpowiedniego polecenia aby uzyskać ich listę

```
ros2 topic list
```

oraz wyświetlić aktualną pozę robozółwia:

```
ros2 topic echo /turtle1/pose
```

Użyj `Ctrl+C`, aby zakończyć subskrypcję danych.

Chcąc opublikować dane do tematu `/turtle1/cmd_vel` i poruszyć robozółwem powinniśmy najpierw sprawdzić typ tematu:

```
ros2 topic type /turtle1/cmd_vel
```

Polecenie zwraca typ `geometry_msgs/msg/Twist`. Typ ten możemy sprawdzić za pomocą następującego polecenia:

```
ros2 interface show geometry_msgs/msg/Twist
```

Struktura `Twist` zawiera dwa wektory 3D. Pierwszy wektor definiuje prędkość liniową, a drugi prędkość kątową. Nasz robozółw, w przeciwieństwie do dronów wielowirnikowych, jest układem nieholonomicznym. Oznacza to, że robot może poruszać się do przodu/tyłu (oś `x`) oraz obracać się wokół osi `z`. Aby poruszać robotem należy opublikować dane do tematu `/turtle1/cmd_vel` ustawiając odpowiednie wartości dla prędkości liniowej i kątowej:

```
ros2 topic pub /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: 0.0"
```



Nie musisz pamiętać typu i struktury wiadomości. Użyj przycisku `Tab`, a terminal linuksowy pomoże ci w podaniu prawidłowych nazw.



Zmień wartości parametrów tak, aby robozółw zaczął się poruszać. Przetestuj różne warianty.

Manipulacja ramieniem robotocznym

Zacznijmy od instalacji sterownika dla robotów Universal Robots:

```
apt install ros-${ROS_DISTRO}-ur-robot-driver
```

Po instalacji możemy uruchomić symulator robota `UR3`. Ten krok wewnętrznie wykorzystuje oprogramowanie Docker (w przypadku błędu, upewnij się, że dokonałeś [konfiguracji dockera](#)):

```
ros2 run ur_client_library start_ursim.sh -m ur3
```

Po uruchomieniu symulatora możemy uruchomić sterownik robota w nowym oknie terminala:

```
ros2 launch ur_robot_driver ur_control.launch.py ur_type:=ur3 robot_ip:=192.168.56.101 launch_rviz:=true
```



Jeśli program `rviz` pokazuje smutne, zwinięte w kulkę i pozbawione tekstur ramię robotyczne spróbuj przerwać działanie powyższego skryptu i uruchomić go jeszcze raz. Po ponownym uruchomieniu odczekaj kilka sekund. Jeśli kilkakrotny restart nie pomaga, otwórz panel robota w przeglądarce (link wyświetla polecenie uruchamiające symulator) i spróbuj poruszyć robotem za pomocą przycisków w panelu. Uruchom sterownik z wizualizacją ponownie (potencjalnie kilkakrotnie).



Panel robota uruchamiany w przeglądarce (PolyScope) to graficzny interfejs użytkownika (GUI) do sterowania ramieniem robota, wykonywania programów robota i łatwego tworzenia nowych. Panel robota dostępny jest pod adresem <http://192.168.56.101:6080/vnc.html>. Prawidłowa uruchomiona symulacja powinna umożliwiać sterowanie robotem w PolyScope i obserwowanie jego ruchów w RViz'ie.

Zadania

- Uruchom symulację `turtlesim`. Dokonaj następującego eksperymentu, którego celem jest poznanie możliwości jednoczesnego publikowania na tym samym topicku. W jednym terminalu publikuj dane z prędkością kątową (obrotów wokół osi z). W osobnym oknie terminala publikuj dane z prędkością liniową. Następnie spróbuj zmodyfikować częstotliwość publikowanych wiadomości wykorzystując argument `--rate` i zaobserwuj efekt najpierw dla tych samych częstotliwości, a następnie dla dziesięciokrotnej różnicy.
- Odczytaj częstotliwość pętli sterowania robota `UR3`. W tym celu wyświetl częstotliwość dla tematu `/joint_states`.
- Poruszaj ramieniem w panelu `UR3`. Sprawdź czy `rviz` poprawnie oddaje pozę robota.
- W terminalu sprawdź jakie tematy i jakich typów utworzył sterownik `UR3`.
- Ustaw ramię w pozycji `Home` wykorzystując panel `UR3`.
- Spróbuj uruchomić przykładową trajektorię wykorzystując polecenie: `ros2 launch ur_robot_driver test_scaled_joint_trajectory_controller.launch.py`. Czy robot porusza się?
- Wykorzystaj panel `UR3` do utworzenia nowego, pustego programu. Dodaj do niego komendę `External Control` (ze `Structure` -> `URCaps`). Uruchom program oraz przykładową trajektorię z poprzedniego zadania. Czy robotot porusza się w panelu oraz `rviz`?
- Przetestuj powyższe na innym robocie `UR`, na przykład `ur5e`.