



Politechnika Łódzka
Instytut Automatyki

Laboratorium

Robotów Usługowych

Instrukcja 3
Processing - ROS

Wstęp

Celem instrukcji jest połączenie utworzonego interfejsu dla robota z wykorzystaniem języka Processing z systemem ROS. Zapoznać się z załączonym do instrukcji przykładem *ros_processing*.

Zadania

Zadanie na 3

Rozbudowa interfejsu (instrukcja 1 zadanie na 3) do obsługi 1 robota o możliwość wysłania po ROS w zależności od wybranego robota na topicu */nazwa_robota/move_base_simple/goal* współrzędnych punktu (co najmniej 1 z 3), do którego dojechać ma robot.

Zadanie na 4

Rozbudowa interfejsu (instrukcja 1 zadanie na 4) o możliwość wysłania do 1 z 3 robotów po ROS informacji o celu, do którego ma dojechać robot oraz podłączenie do ROS kontrolki sterujących prędkościami postępowymi i obrotowymi dla robota. Prędkości należy wysłać na topicu */nazwa_robota/cmd_vel*. Prędkość postępową należy wysłać w wiadomości w polu **linear.x**, a obrotową na **angular.z**.

Zadanie na 5

Do tego zadania należy uruchomić dodatkowo skrypt "task_result" dołączony do instrukcji 3. Biblioteka ROSprocessing obsługuje tylko niektóre typy wiadomości ROS, dlatego do przekazywania informacji o statusach będzie wykorzystywana wiadomość typu Vector3 z biblioteki geometry_msgs.

Należy rozbudować interfejs (instrukcja 1 zadanie na 5) o kontrolki odbierające informację na topicu */robots_task_status*. W polach x, y, z kolejno przekazane są statusy po dojechaniu do punktu dla robotów tb3_0, tb3_1, tb3_2. Należy wyświetlać odebrany status w zależności od stanu robota.

Znaczenie statusów:

```
uint8 PENDING=0
uint8 ACTIVE=1
uint8 PREEMPTED=2
uint8 SUCCEEDED=3
uint8 ABORTED=4
uint8 REJECTED=5
uint8 PREEMPTING=6
uint8 RECALLING=7
uint8 RECALLED=8
uint8 LOST=9
```

Do skonwertowania odebranej wiadomości i wyświetlenia tekstu po stronie interfejsu można użyć funkcji:

Double.toString(zmienna)