# Anatomia robotów

Laboratorium 1

Marcin Hanas

Radosław Tuzimek

## Cel zadania:

*Za pomocą dowolnego z dwóch języków (C++, Python) tworzymy program, który będzie służył do sterowania żółwiem*

* *Program powinien zapewnić sterowanie za pomocą innych klawiszy niż oryginalnie w przewodniku, np.:****g d p l****.*
* *Klawisze, którymi się steruje, powinny być wczytywane z serwera parametrów. Serwer parametrów startuje automatycznie po uruchomieniu roscore.*

## Pliki programu:

##### move.py

#!/usr/bin/env python

# import potrzebnych pakietow

**import** click

**import** rospy

**from** geometry\_msgs**.**msg **import** Twist

# funkcja odpowiadajaca za sterowanie

**def** move**():**

# Stworzenie nowego publishera

pub **=** rospy**.**Publisher**(**'/turtle1/cmd\_vel'**,** Twist**,** queue\_size**=**10**)**

# Zainicjowanie wezla

rospy**.**init\_node**(**'move'**)**

# Pobranie z serwera parametrow odpowiednich liter klawiszy

UP **=** rospy**.**get\_param**(**"up"**)**

LEFT **=** rospy**.**get\_param**(**"left"**)**

DOWN **=** rospy**.**get\_param**(**"down"**)**

RIGHT **=** rospy**.**get\_param**(**"right"**)**

# Petla wykonywana, dopoki program dziala

**while** **not** rospy**.**is\_shutdown**():**

# Stworzenie obiektu typu Twist - wiadomosci wysylanej przez wezel

# zawierajacej wspolrzedne przesuniecia i obrotu

twist **=** Twist**()**

# Uzyskanie informacji o wcisnietym przycisku

key\_pressed **=** click**.**getchar**()**

# Sprawdzenie, ktory przycisk zostal wcisniety i ustawianie

# parametrow ruchu

**if** key\_pressed **==** UP**:**

twist**.**linear**.**x **=** 2

**elif** key\_pressed **==** LEFT**:**

twist**.**angular**.**z **=** 1.04719755119

**elif** key\_pressed **==** DOWN**:**

twist**.**linear**.**x **=** **-** 2

**elif** key\_pressed **==** RIGHT**:**

twist**.**angular**.**z **=** **-**1.04719755119

# Publikowanie wiadomosci o przemieszceniu na topic

pub**.**publish**(**twist**)**

**if** \_\_name\_\_ **==** '\_\_main\_\_'**:**

**try:**

move**()**

**except** rospy**.**ROSInterruptException**:**

**pass**

##### turtle\_move.launch

<launch>

# Podawanie do serwera paramterow liter odpowiadajacym

# klawiszom odpowiedzialnym za sterowanie

<rosparam param="left">'g'</rosparam>

<rosparam param="right">'j'</rosparam>

<rosparam param="up">'y'</rosparam>

<rosparam param="down">'h'</rosparam>

# Stworzenie wezla - zolwia Zdzisia

<node pkg="turtlesim" name="Zdzisiu" type="turtlesim\_node"/>

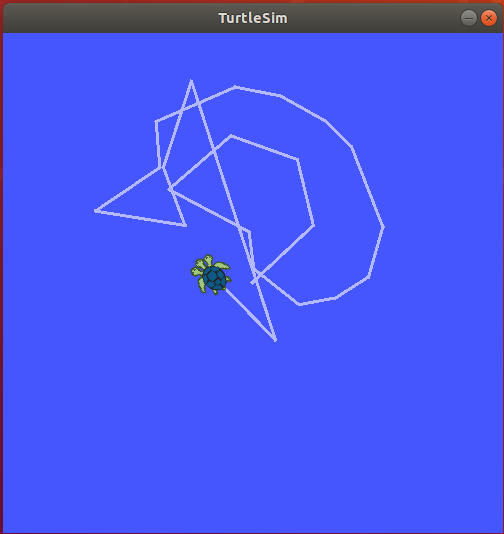
# Stworzenie wezla wysylajacego wiadomosci sterujace

<node pkg="lab1" name="move" type="move.py" output="screen"/>

</launch>

## Opis działania:

Program uruchamiany jest za pomocą pliku turtle\_move.launch, który tworzy nowe węzły – żółwia Zdzisia i obiekt sterujący, oraz dodaje wartości do serwera parametrów, używając komendy roslaunch lab1 turtle\_move.launch. W naszym przypadku, żółw sterowany jest klawiszami: g (lewo), h (tył), j (prawo), y (przód). Następnie żółw jest sterowany za pomocą zdefiniowanych w tym pliku klawiszy. Dokładne działanie programu sterującego zawarte jest w komentarzach.



## Schemat węzłów:



Węzeł „move” przesyła do węzła „Zdzisiu” wiadomości (typu Twist message).