Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

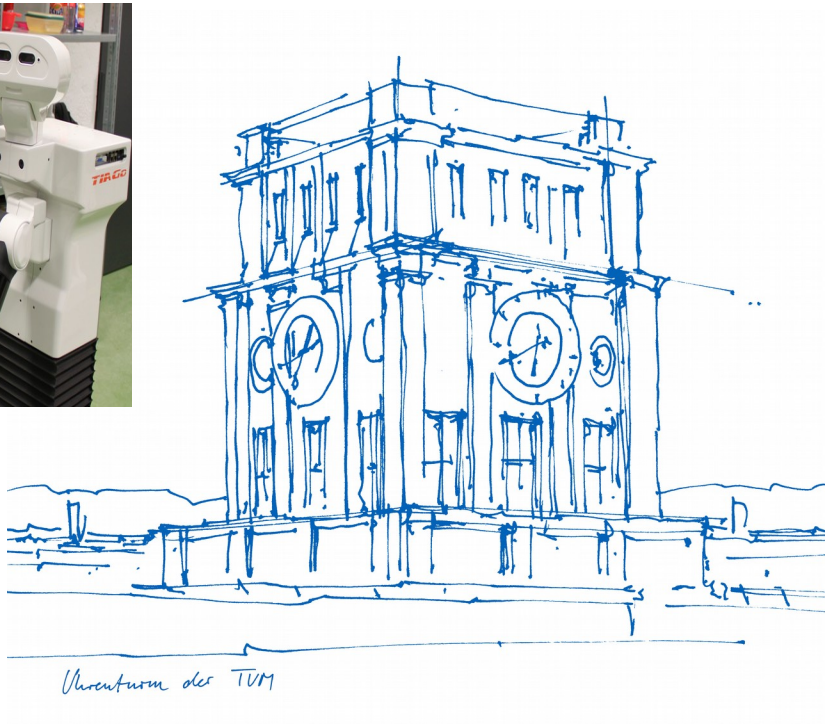
RoboCup@Home Practical course

Tutorials

M.Sc. Rogelio Guadarrama
M.Sc. Constantin Uhde
Dr. Emmanuel Dean
Dr. Gordon Cheng

Technical University of Munich

Chair for Cognitive Systems



Introduction

- Send the tutorials homework here:

Email: robocup.atHome.ics@gmail.com

Remember:
Individual laboratory assignments:30%

Tutorial 1: ROS Advanced

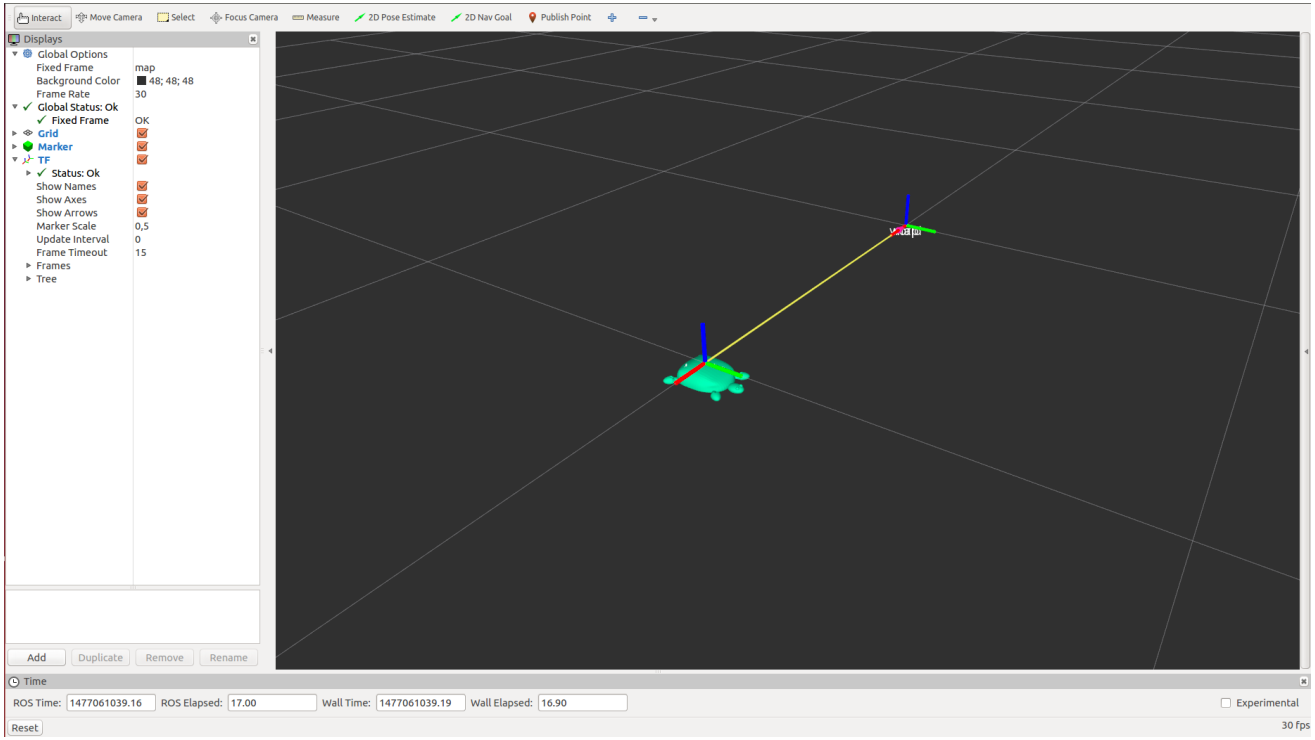
Introduction Questions

Before you start working in the tutorial, please make sure that you can answer these questions.

- What is ROS?
- What is a node?
- What is the ROS core?
- How can nodes exchange information?
- What is a topic?
- What is a service?
- What is an action?
- What is a message?
- Is ROS real-time safe?
- Am I ready to use ROS?

Tutorial 1: ROS intermediate-level

Goal: Move the position and orientation of one turtle to a desired position



You can use: roslaunch turtle_vis TurtleVis.launch

Tutorial 1: ROS intermediate-level

1. Create a new ros workspace

```
$cd ~
$cd ros/workspace
$mkdir roboCupHome_tutorial_YOURNAME
$cd  roboCupHome_tutorial_YOURNAME
$mkdir src
$cd src
$catkin_init_workspace
```

2. Compile the new workspace

```
$cd ~/ros/workspace/roboCupHome_tutorial_YOURNAME
$catkin_make
```

3. Remember to source your new ros workspace

```
$cd ~/ros/workspace/roboCupHome_tutorial_YOURNAME
$source devel/setup.bash
```

Tutorial 1: ROS intermediate-level

1. -- Qtcreator -- Tools to Debug your code.

$cd ~/ros/worspace/roboCupHome_tutorial_YOURNAME
$cd src
$mv CMakeLists.txt CMakeLists.txt.old
$cp CMakeLists.txt.old CMakeLists.txt

2. Open your Code

$cd ~/ros/worspace/roboCupHome_tutorial_YOURNAME/src
$qtcreator CMakeLists.txt

3. Explore Qtcreator

4. Copy the template folder in your ros workspace and compile

$cd ~/ros/worspace/roboCupHome_tutorial_YOURNAME/src/turtle_vis

Tutorial 1: ROS intermediate-level

5. Compile the template folder.

$cd ~/ros/worspace/roboCupHome_tutorial_YOURNAME
$catkin_make

Tutorial 1: ROS intermediate-level

Exercise 1: Fix the CMakeList.txt of the template project

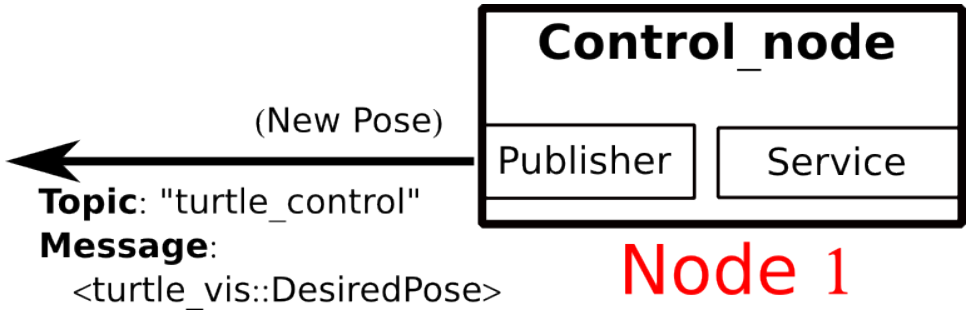
Hint: Look for the #>>>TODO inside the file

- Add the new defined messages
- Include the required service files
- Include the name of the new defined library on catkin_package
- Add the nodes that will be executed and create the proper target link.

Take a look at: http://wiki.ros.org/catkin/CMakeLists.txt

Tutorial 1: ROS intermediate-level

Exercise 2: Create a node to compute the new turtle pose using a simple Kinematic Control.



Node 1: **Turtle_Control_node**

a) This node will provide:

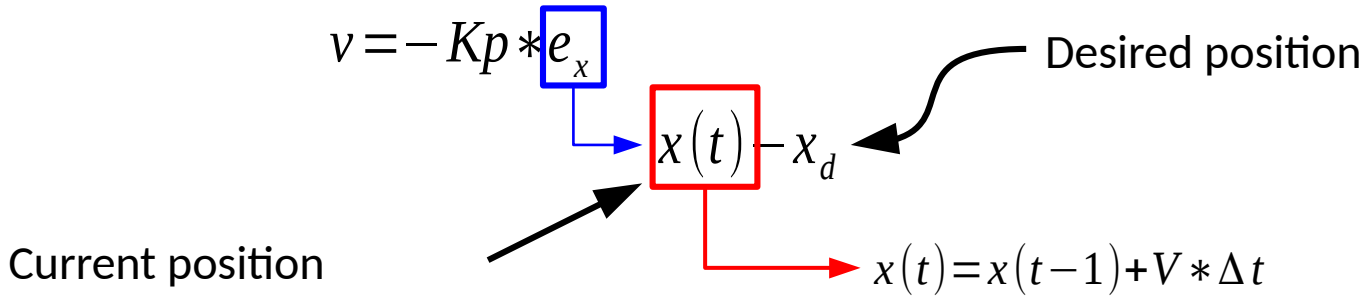
a)Service: this service is used to receive the desired turtle pose (x_d,y_d,theta_d).

b)Publisher: it will publish a topic “turtle_control” with the new turtle pose. This pose will be listened by Node 3.

Tutorial 1: ROS intermediate-level

Inside the main loop of “turtle_control_node.cpp”:

- Obtain the desired pose (x_d) from a class variable (see slide 22, definition of the class).
- Implement a P-control (Kinematic control) to move the turtle to a desired (d) position.



- Publish the obtained turtle position to Node 3.

Tutorial 1: ROS intermediate-level

Use the provided template “turtle_control_node.cpp” to create node 1, look for all #>>>TODO:

- To set the gain values for the controller (Kp), create a *.yaml file, which is usually in a new folder inside your package e.g. turtle_vis/configs.
- Modify the launch file to set the ros parameters from the yaml file (see the template in turtle_vis/launch).
- Define a message type to send the new position of the turtle through a publisher topic to Node 3 (see slide 9 for a message example).

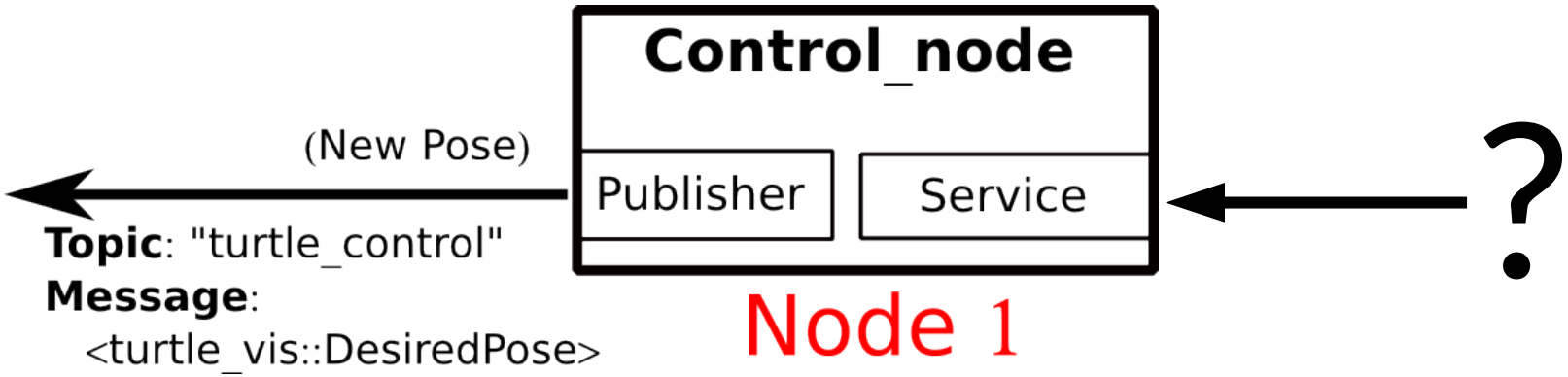
Take a look at: http://wiki.ros.org/rosparam

Tutorial 1: ROS intermediate-level

Exercise 3: Answer the following questions:

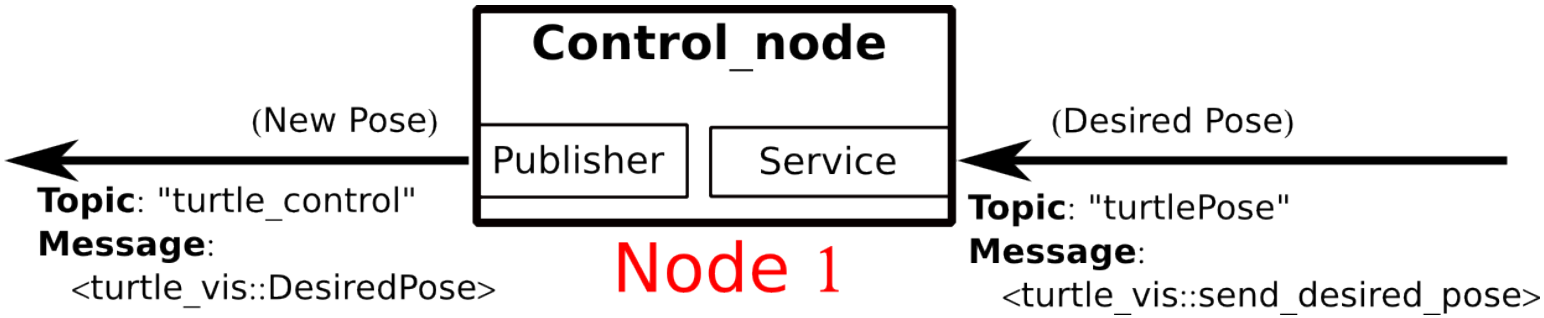
1) How can I send the new desired pose of the turtle (x_d, y_d, theta_d) to Node 1. Please, indicate the command that you will use and explain the reasons.

2)What is the main difference between a Publisher/Subscriber and Service/Client? For example, what would happen if I replace the Service for a Subscriber in Node 1?



Tutorial 1: ROS intermediate-level

- Node 1 also has a service that will receive the desired position and orientation of the turtle.
- Define a new message type for the service of Node 1.
- Replace the "CALLBACKFUNCTION" from the template file (TurtleClass.h and TurtleClass.cpp) with the name of your function for the service, this function should be defined in your class (see slide 22).



Tutorial 1: ROS intermediate-level

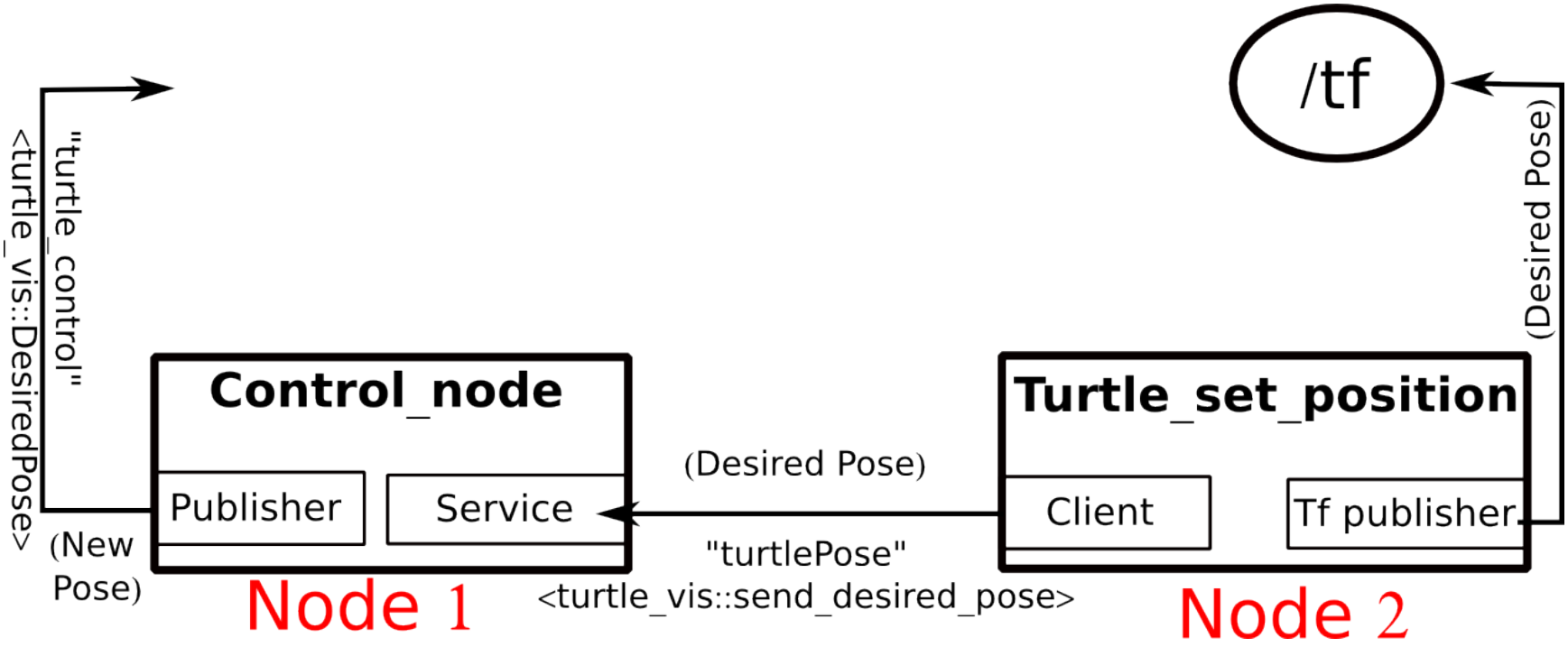
Hint: You can use the following command to debug the correct behavior of node 1, i.e. send the desired position of the turtle from the ros service terminal command, e.g.

```
rosservice call /TurtlePose "p:
    x: 0.0
    y: 0.0
    theta: 0.0"
```

Take a look at: http://wiki.ros.org/rosservice

Tutorial 1: ROS intermediate-level

Exercise 4: Create node 2 to set the new desired pose of the turtle (x_d, y_d, theta_d) from a client to your defined service from node 1 .



Tutorial 1: ROS intermediate-level

Node 2 has the name: **Turtle_set_position_node** and it contains:

- **Client:** will connect to the service provided by Node 1 and will send the desired turtle pose
- The desired pose should be acquired from the terminal (x_d, y_d and theta_d) in a continuous loop (this is a different terminal command than the one used in slide 14).
- **TF Publisher:** will publish the coordinate frame of the desired turtle pose to the /tf node.

You must define a custom message. This message will be used for the topic for the Service/Client definition (communication between Node 1 and Node 2), e.g. turtle_vis::send_desired_pose.

Tutorial 1: ROS intermediate-level

Exercise 5: Create Node 3 that receives the new computed pose of the turtle from Node 1 and visualize the turtle with its new pose.

Node 3: **Turtle_vis_node**

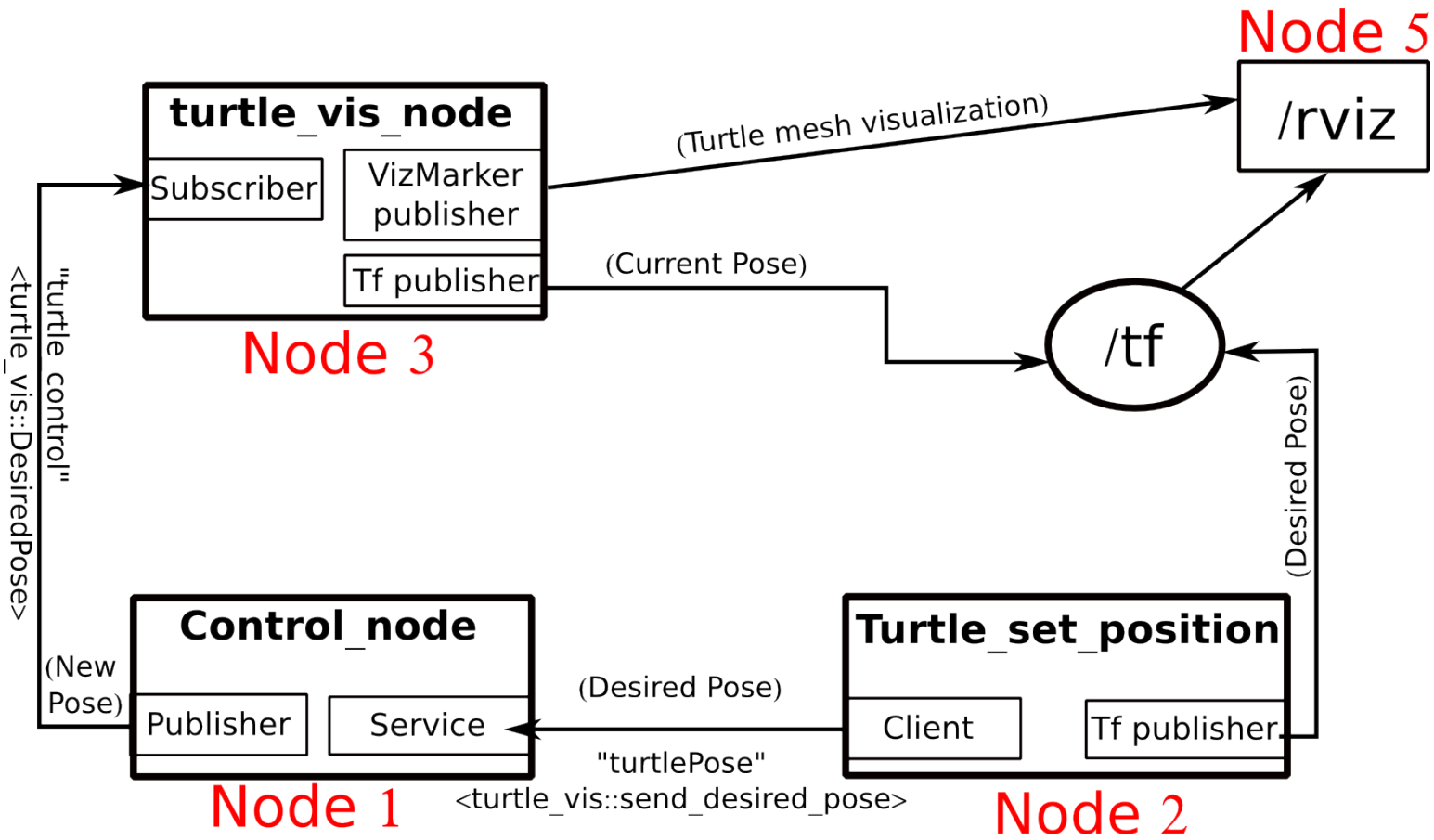
This node will provide:

a) **Subscriber:** will connect to the topic generated by Node 1 and will generate the coordinate frame for the current turtle position and the visualization of the turtle.

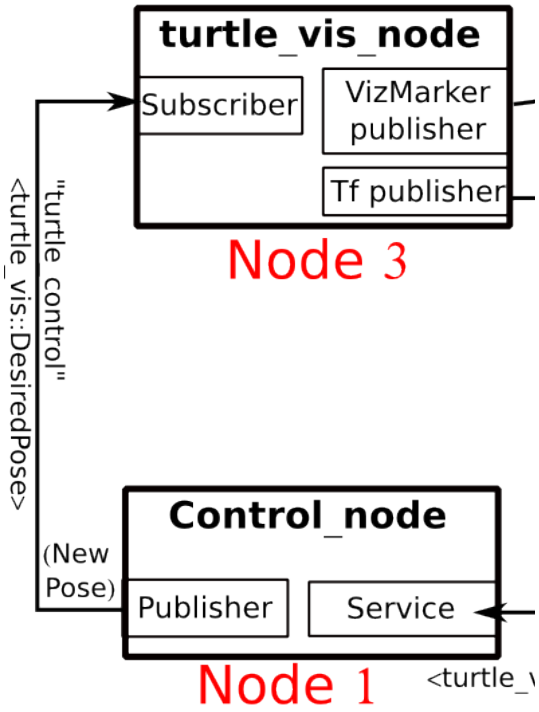
b) **TF Publisher:** to publish the coordinate frame for the current turtle pose.

c) **Visualization_Marker Publisher:** To visualize the turtle mesh in rviz.

Tutorial 1: ROS intermediate-level



Tutorial 1: ROS intermediate-level



You must define a custom message. This message will be used for the topic publisher/subscriber (communication between Node 1 and Node 3), e.g:

turtle_vis::DesiredPose

Tutorial 1: ROS intermediate-level

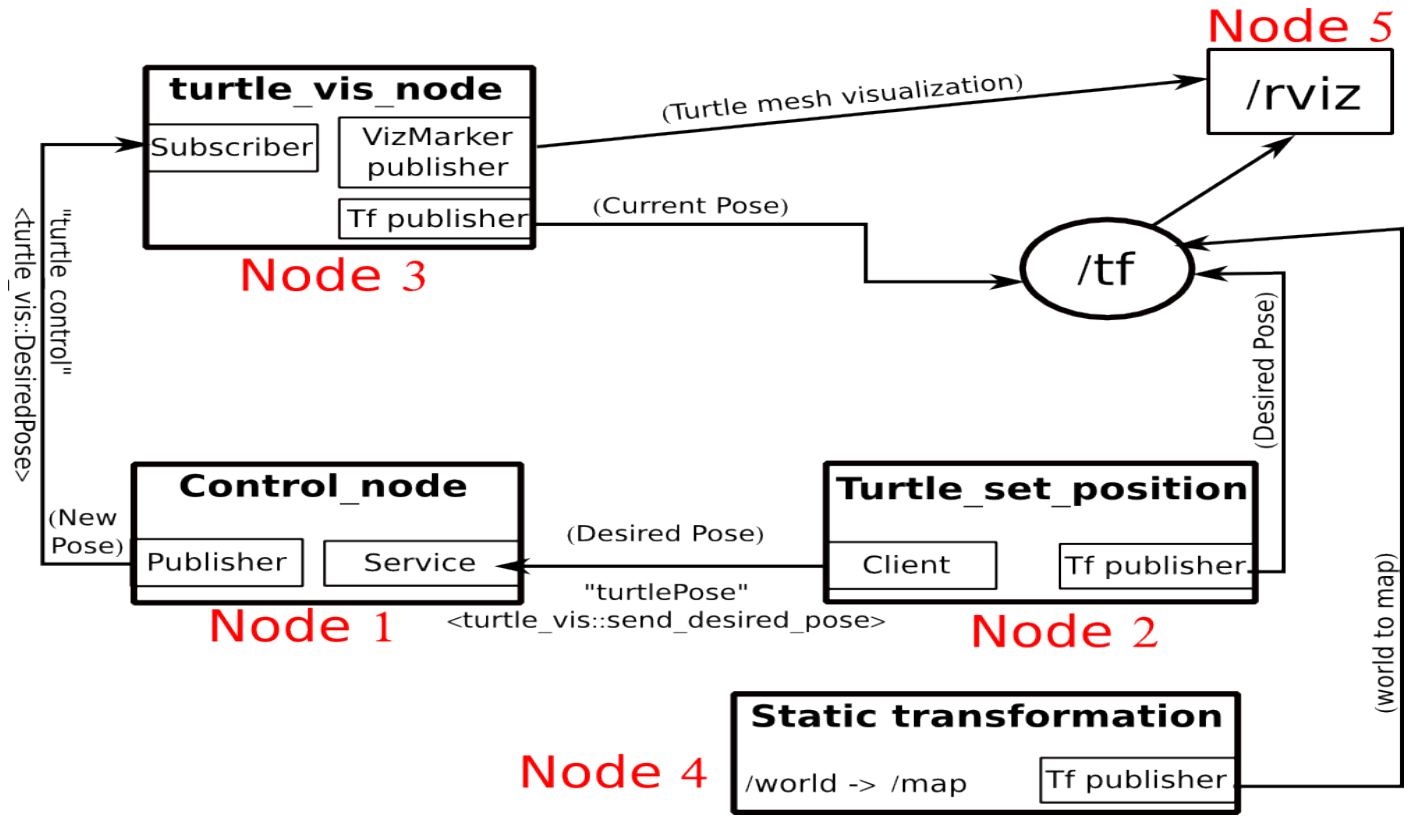
As additional nodes, we have static transformations as well as rviz transformations using tf.

- **Node 4:** Static transformation
  This node will perform:
  a) A static transformation between the coordinate frames /map and /world. This tf will publish the rviz visualization.

- **Node 5**: rviz visualization
  a) Visualize the turtle mesh (marker), the world coordinate frame (tf), the turtle tf and the desired pose tf.

Tutorial 1: ROS intermediate-level

The nodes should be connected as shown in the Figure:



Tutorial 1: ROS intermediate-level

Exercise 6: The following sub-tasks should be considered.

Make your system modular, i.e. use a common object class for the subscriber (Node 3) and the client (Node 2) callback functions, i.e.
Look at the file: turtle_vis/src/solutions/myClass

- Create a callback function for the service in Node 1
- Create two methods for obtaining the turtle pose needed in Node 1
- Create a callback function for the subscriber of Node 3

Important: Create the header for the class in a separate file and place it in turtle_vis/include/turtle_vis/myClass

Tutorial 1: ROS intermediate-level

Use the given template to complete the exercises.

You should deliver a compress file with your implementations using the instructions from the slides and the given template. Please include a readMe.txt file to indicate how to run your nodes and to answer the questions.

Please, name the compress file as follows:

“Name_lastName_roboCupHome_tutorial1”

Tutorial 1: ROS intermediate-level

Example of the readMe.txt file:

1) roslaunch turtle_vis TurtleVis.launch

2) rosservice call /TurtlePose "p:
x: 2.0
y: 2.0
theta: 1.57"

3) rosrun turtle_vis turtle_set_position_node (cm, cm, rad)

Tutorial 1: ROS intermediate-level

Important links:

http://wiki.ros.org/ROS/Tutorials

http://wiki.ros.org/Services

http://wiki.ros.org/roscpp/Overview/Publishers%20and%20Subscribers

Q & A about this homework Wednesday and Friday @15:00 hrs. Tuesday 10:00 hrs.

Deadline to deliver this homework: Tuesday 29th Oct @23:59 hrs

Q&A for the Tutorials

Assistant Tutor: Chenghao Wang

Availability for Q & A

Tuesday	10:00 – 12:00
Wednesday	15:00 – 17:00
Friday	15:00 – 17:00