Robot Practical Course

Assignment #1

Due: 30.04.2021, 13.00

This assignment is supposed to set up your working environment, familiarize yourself with ROS and the `itr_rpc` package.

Task 1.1 Join a group: Before you start with the actual tasks, join a group in moodle (`https://lernen.min.uni-hamburg.de/mod/groupselect/view.php?id=48084`). You can connect with your group via BigBlueButton before the first appointment and work together. You will find a link to the BigBlueButton room in Moodle.
All assignments should be solved and handed in in groups of three.

Task 1.2 Set up your OS and installing ROS: For this practical course, it is necessary to work with Ubuntu, we will use version 18.04. In the following sections you will learn how to set up a Ubuntu operating system on your computer or alternatively in a virtual machine. The easiest way is to use the virtual machine image we prepared for this lecture described later.

1.2.1: Native Ubuntu
To install a native Ubuntu on your computer, follow this tutorial: `https://ubuntu.com/tutorials/tutorial-install-ubuntu-desktop#1-overview`

1.2.2: Virtual Machine
Download and install a VM, we recommend VirtualBox (`https://www.virtualbox.org/`). To enable USB 2.0+ support, it is necessary to install the extension pack (`https://download.virtualbox.org/virtualbox/6.1.18/Oracle_VM_VirtualBox_Extension_Pack-6.1.18.vbox-extpack`).
We recommend you to download the prepared image from our tams website with a ready to start system (`https://tams.informatik.uni-hamburg.de/lectures/2021ss/vorlesung/itr/doc/Ubuntu_ITR_SS21.ova`). To import the image go to `File` → `Import Appliance...` and follow the instructions. The username is `"user"` and the password is `"password"`.

If you want to set up your own virtual machine, you will find several tutorials for Windows, Mac, and Linux. Make sure, that you use Ubuntu 18.04.4 LTS.

1.2.3: Installing ROS
If you use the provided VM image, you can skip this section.

Otherwise, you need to install ROS by yourself. You find a tutorial for the installation on `https://wiki.ros.org/melodic/Installation/Ubuntu`. Follow the instructions and choose "Desktop-Full Install" in 1.4. More documentation can be found at `https://wiki.ros.org`.

After setting up ROS, you need to install some additional packages, use the following terminal command: `sudo apt install -y git python-catkin-tools ros-melodic-map-server ros-melodic-trac-ik`

Task 1.3 Set up your own workspace: You will set up your workspace and checkout all necessary repositories for this course now.

1.3.1: First of all, you need to set up your environment and workspace for ROS. This only has to be done once for your user account.

- Open a command shell (`Ctrl+Alt+T`)
- Load the ROS environment `source /opt/ros/melodic/setup.bash`
- Create a folder for your workspace `mkdir -p ~/ros_ws/src` (The ~ is an alias for your homefolder in Linux, `mkdir -p` creates a new folder with all required intermediate folders.)
- Change into the workspace `cd ~/ros_ws/` (cd stands for change directory.)
- Initialize the workspace `catkin build` (This is the command for building your workspace. It has to be called inside your ROS workspace. You will need it again later on.)

Your personal workspace is now set up. Nevertheless, you would have to tell each shell which you open about your workspace and the ROS environment. We will now make this permanent, so you do not forget to load it.

- In your shell, open your shell profile either in a graphical or a command line editor
  `vim ~/.bashrc` (command line)
  `gedit ~/.bashrc` (graphical)
- At the end of the file, add `source ~/ros_ws/devel/setup.bash`. Save and close.
- In all open shells, enter `exec bash`

1.3.2: Check out the project from the git repository. There will be updates to the project during the practical course. In order to avoid conflicts, only edit the files which you are instructed to edit. If you are unfamiliar with git and would like to know more, please read a guide on how to use git (e.g., `https://githowto.com/`).

- Navigate to the `src` folder in your workspace.
- Check out the git repository `git clone https://git.mafiasi.de/jonetzko/itr_rpc.git` using your mafiasi account (with a two-digit number).
- Build your workspace to generate all required message and service definitions.
- Update your package index with `rospack profile` and `exec bash` in each shell

Now you are all set with your ROS environment and the workspace. For the future, there are some ROS specific commands which might be useful:
`roscd`, `rosrun`, `roslaunch`, `roscore`, `rostopic`, `rosservice`, `rosclean`, `rosmsg`, `rosnode`.
All these commands support `tab completion`, be sure to use it as it will avoid typing mistakes.

Task 1.4 Launching ROS nodes: In this task, you will learn how to launch ROS nodes and how to interact with ROS.

1.4.1: Launch the graphical display `roslaunch itr_rpc task_1.launch`. Inspect the available messages, topics, and services.

1.4.2: Run the default forward kinematics script `rosrun itr_rpc dummy_fk.py`. Discuss the behavior.

1.4.3: ROS supports dynamically reconfiguring parameters of running nodes. Run the configuration GUI `rosrun rqt_reconfigure rqt_reconfigure` and switch off the velocity limits. Explain the difference in the behavior.

Task 1.5 Write your first node: In this task, you will write your first own node, which performs a circular motion with the TCP. **Ensure, that the velocity limits are off**. After relaunching the GUI, you will have to change the parameter again.

When writing code in Python, ensure that you are using the correct amount of spaces for indentation. In case you use `gedit` as editor, on the bottom bar, set the tab width to `4` and check `Use Spaces` (Check box must be checked).

1.5.1: The circular motion must be performed around $\left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$ with a radius of 1. Inspect the GUI to find out the orientation of the coordinate system. What does the rotation look like. Write your script in the `nodes/script.py` file and inspect `nodes/dummy_fk.py` for assistance.

1.5.2: Turn on the velocity limits. Explain what is happening and why. Can you fix your code to work with the velocity limits?

1.5.3: The circular motion is supposed to be only around $\left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$. Verify that your circular motion does not start before reaching that position by relaunching the GUI.