UVA@HOME

LABBOOK

UVA INTELLIGENT ROBOTICS LAB

Introduction

This is the labbook for the development and research done at the UvA Intelligent Robotics Lab.

May 2018

May 8, Janne Spijkervet Tim MÃijller joined the team. All packages needed to develop for the robots were installed on his computer.

May 16, Janne Spijkervet I wrote a set_network.sh script for a multi-machine setup with ROS. Usage: source set_network.sh <IP_OF_ROS_CORE_MACHINE>

May 16, Janne Spijkervet Chris Al Gerges, Mirka Schouten and Charlotte Kaandorp joined the team. I ran through all the installation instructions with them in the robotlab to make sure their computers were ready for developing in our framework (ROS) and the Pepper robot. I also gave them a quick introduction into publisher/subscriber node's topics, data structures and writing a first Hello World program for the Pepper robot (including an example with head/hand sensors).

May 16, Janne Spijkervet A new script has been added to the uva-robotics repository to run ROS with multiple machines. This involved setting the ROS_MASTER_URI, ROS_IP environment variables to that of the ROS core machine and the child machine.

May 18, Janne Spijkervet ROS Hydro has been installed on one of the MBots (DHenk). Most sensors work again. There are still some problems with the voltage warning (and shutdown of the machine), but hopefully this will be solved when it gets a full charge.

May 18, Janne Spijkervet ROS has been compiled for the Pepper robot. It took a lot of time to make cross-compilation work. I used a docker implementation found on Github and tweaked the settings so it would work on our current Pepper/Naoqi/libqi version (2.5.5). I am planning to do the same for the Nao robots. I also wrote the pepper_bringup.launch scripts so all nodes/topics will be available at startup.

May 18, Janne Spijkervet I created the slack_ros package so we can easily communicate with our ROS robots through Slack. This can be found at `https://github.com/uva-robotics/slack_ros`.

April 2018

ROS

Janne Spijkervet While starting to program a new framework for the development of algorithms and programs for the Pepper robot, I stumbled upon ROS (Robot Operating System). Much of the design I wanted to make for my framework (publisher/subscriber programs, nodes) were already tested and tried in the ROS framework. I decided to port the work I had done to ROS (face recognition, text-to-speech, speech-to-text).

ROS2

Janne Spijkervet While looking through ROS' documentation, I was more tended to choose ROS2 for our framework. But, as its documentation is very scarce compared to ROS, I figured it would be better to go with ROS for now until I am familiar with the framework. ROS2 is preffered for us, because we often work with low-quality, unstable (wireless) networks. Also, we prefer real-time processing over anything else. ROS2 is built on newer protocols (DDS). **In the future, moving forward to ROS2 is very recommended!**

Repositories

Janne Spijkervet A new repository group was made that holds all ROS packages developed for the Intelligent Robotics Lab. This was done to make implementations of algorithms and other code more re-usable for future work. It is important to implement standards for robotics programming. ROS is one of such platforms that envisions this for future research and practical implementations of robotics software. The repositories can be found at: `https://gitlab.com/uva-robotics`.

Workstations

Janne Spijkervet ROS was installed on both workstations at the lab. This was done to speed up the work that involves the use of a GPU, since most students only have a laptop to work on.

Installation

Janne Spijkervet Installation README: https://gitlab.com/uva-robotics/uva-robotics

Uva Robotics Repository for the development of intelligent systems for the UvA Intelligent Robotics Lab. Most programs use either ROS or our own Redis/Websockets stack as middleware. Below, you will find a basic installation tutorial to install all required dependencies. ROS runs on Python 2.7.

Boilerplate package In the https://gitlab.com/uva-robotics/_boilerplate respository, you will find a template for building your own ROS packages. It contains one meta-package folder and the package folder itself. We use meta-packages so that we don't get a lot of sub-repositories in our git directory. The boilerplate can be used for C++ and Python (2.7) code.

ROS installation

Add keys and update package manager

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key 421C365BD9FF1F717815A3895523BA
sudo apt-get update
```

Full install:

```
sudo apt-get install ros-kinetic-desktop-full
```

Minimum install:

```
sudo apt-get install ros-kinetic-ros-base
```

Initialize rosdep

```
sudo rosdep init
rosdep update
```

Environment setup

```
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Dependencies for building packages

```
sudo apt-get install python-rosinstall python-rosinstall-generator python-wstool build-essential
```

Installing ROS Pepper

```
sudo apt-get install ros-kinetic-pepper-.*
```

Go into your workspace (e.g. ~/projects/uva-robotics).
Navigate to src directory (e.g. ~/projects/uva-robotics/src).

```
git clone https://github.com/ros-naoqi/naoqi_driver.git
rosdep install -i -y --from-paths ./naoqi_driver
source /opt/ros/kinetic/setup.sh
cd ../ && catkin_make
```

Running ROS

(Pepper example) Run the following command to run roscore:

```
roscore
```

C++ version:

```
roslaunch pepper_bringup pepper_full.launch nao_ip:=<yourRobotIP> roscore_ip:=<roscore_ip> [network_interf
```

Python version:

```
roslaunch pepper_bringup pepper_full_py.launch nao_ip:=<yourRobotIP> roscore_ip:=<roscore_ip>
```

Using ROS

First, make sure you are running roscore in a seperate terminal.

```
roscore
```

List all running nodes:

```
rosnode list
```

List all topics:

```
rostopic list
```

View data from topic:

```
rostopic echo <topic>
```

Run a ROS package script:

```
rosrun <package_name> <script_name>
```

Run a ROS launch file:

```
roslaunch <package_name> <launch_file_name>
```

Bibliography