SOFAR lab 1 – Launch files and topics

In this lab, we will build a simulation of the motions of swarms of birds. The swarm is merely a pretext for the lab, and the result is anything but a correct rendering of the actual motions of a swarm.

In some cases the motions look like all birds are following a leader, when in fact, it seems there is no identified leader in a swarm. In this simulation, the individuals in the swarm move to a close proximity of a goal point set by a « virtual leader ». The virtual leader **is not a swarm member**. We will build the simulation using three types of ROS nodes: on representing a member of a swarm, one to randomly assign a new position to the virtual leader and one to trigger the modification of the position of the swarm leader by pressing a predefined key.

Our aim is to simulate two independent swarms, which we will call the « blue swarm » and the « red swarm ».

The available packages

You have two packages:

- The « `capture_key` » package. It contains a single type of node. Such a node reads the keyboard and publishes the ASCII code of the characters that are typed. We will use this node to trigger random motions of swarm leaders.
- The « `swarm` » package, which contains two types of nodes:
  - « `swarm_node` » to simulate a bird of the swarm.
  - « `swarm_behavior` » to randomly set a new position for the virtual leader.

Your tasks

The instructions below are given assuming you are using the ECN Virtual Machine, where the username is `ecn` and the folder for the catkin workspace is `/home/ecn/ros`. If not, replace `/home/ecn` by `/home/your_user_name`. I suggest you establish your catkin workspace in `/home/your_user_name/ros` to minimize differences with the “standard” setup.

If you are using Ubuntu, and not the Xubuntu virtual machine, I strongly recommend you install catkin_tools. It will allow you to use the command `catkin build` for compilation. Beware: this command cannot be used concurrently to `catkin_make`. In a given catkin workspace, you either use one or the other, and you stick to your choice all the time. The instructions to install catkin_tools are available here (four commands to copy and paste to a terminal, takes a minute): https://catkin-tools.readthedocs.io/en/latest/installing.html

1. Preparation

- Initialize your work environment. There is a video to help you organize your workspaces (link on Hippocampus).
- Make your `ecn/ros` folder a catkin workspace. In a terminal, go to the folder (`cd ros` if you are in your home folder, or `cd /home/ecn/ros` from anywhere). Then type `catkin_init_workspace`.
- Create a src folder in your catkin workspace (`mkdir src`). Make sure you are in `ecn/ros`. If unsure, type `pwd`. The command should answer `/home/ecn/ros`.
- Run the command `catkin build` to pre-build your catkin workspace. This should create

three new folders under `/home/ecn/ros`, named `build`, `devel` and `logs`. Remember that deleting these folders forces catkin to recompile all your packages.

- Copy the two provided packages to your `ros/src` folder.
- Compile the packages (command: `catkin build`). You should have no errors if you have followed all previous steps carefully. You should see the following message:
  [build] Note: Workspace packages have changed, please re-source setup files to use them.
- Execute the commande `source devel/setup.bash`. This is necessary for the system to find your newly created ROS packages. You will need to execute this command, not after each compilation, but **each time your compilation creates a new package**, which is the case here. If you need to run it from a different location, give the full path to the file, like this: `source /home/ecn/ros/devel/setup.bash`.
- Check that you can run the nodes (they won't do anything visible, though). Run each of these commands in a separate tab of a terminal (use multiple tabs rather than multiple terminals, otherwise the screen becomes messy.
  - `roscore`
  - `rosrun capture_key capture_key_node`
  - `rosrun swarm swarm_node`
  - `rosrun swarm swarm_behavior`

    Remember to use the “Tab” key to auto-complete the package name and node type. If auto-completion does not work (and your compilation did not fail) it means that ROS does not find your packages: run the “source” command above.
- Create a fifth tab to execute commands and stay in this fifth tab. Leave all nodes running.
- Inform the teacher when you are done with the preparation.

2. Answer the online questionnaire

On Hippocampus, you will find an online questionnaire entitled “lab_questionnaire_1”. Answer it and submit your answers before proceeding.
The questions are designed to help you complete the next task. When you have submitted the questionnaire, you have access to the correct answers.

3. Creation of two swarms using the unmodified provided software

Complementary preparation:

- Kill all the nodes you had started in different tabs. Leave only roscore running. Use Ctrl-C in each tab to kill the nodes.
- Initialize rviz. For the tasks of this evaluation, it is convenient to have rviz running in the same workspace where you will launch your application. Do not put it in full screen mode. Run rviz using the « `rviz` » command in a tab. In the « fixed frame » drop down menu (top left region of rviz' screen), select the frame « map » and hit the return key. Then click on the « add » button (bottom left region of rviz' screen). A list of displayable items appears. Choose « marker » and click the « OK » button.

In this first task, you must use the software « as is ». No modification of the source is allowed. But you can read the relevant part of the source files to understand how they handle topic and node names. Refer to the table on slide 68.

- Draw the connection graph you want to obtain to run this application. Draw this more or less how rqg_graph does: all the nodes you want, with their names, and how they should connect via topics (with topic names). You can draw on paper and take a photograph or use your favorite drawing software. **Before proceeding, you must ask the teacher to validate your graph**. If you continue with an incorrect graph, you are loosing your time.

- Create a launch file which allows to create and run two independent swarms, the « blue swarm » and the « red swarm ». Use the launch file template in the « launch » sub-folder of the « swarm » package.
- Each swarm will contain two « birds » (I know, that's a pretty small swarm!).
- The blue (resp. red) swarm should move when the 'b', ascii code 98, (resp. 'r', ascii code 114) key is hit.
- The world size shall be set to 2.0 (it's a global parameter and, thus, must be set outside of any `<node>...</node>` section, and its name should start with a '`/`' to set it).
- In a separate workspace, display the rqt_graph (command `rqt_graph`).
- When the task is completed:
  - **Call the teacher for validation**.
  - Create a copy of the launch file, of the form <name>_task1.launch (*e.g.* Salvador_Dali_task1.launch) and submit it on Hippocampus.

4. Creation of two swarms after modifying the provided software

Now we want to create a « blue swarm » and a « red swarm » with 5 birds each. It's still a pretty small swarm, but big enough for our purpose.

For this second task, you can modify the source code of any nodes in the provided packages. Your goal is to find a solution which will minimize the remappings in the launch file, so it's easy to create larger swarms.

The validation steps are the same as for task 2. Submission will consist of a single zipped file of the form <name>_task2.zip, which must contain:

- The « capture_key » and « swarm » packages, whether modified or not.
- Both launch files of tasks 1 and 2.

It is your responsibility to check that the submission is complete. As for the first task, the teacher must have seen the state of your submitted work so he can take note of the achieved result. No submission will be taken into account unless this step has been performed.