Part I --- Configuration ---

PACKAGE

• ***One ROS Package*** named **ar*_week5_test*** should be created, within the src folde of your catkin workspace (note: make sure you later modify your Manifest and CMakeLists files depending on the specific requirements of your package...).

• The ar_week5_test package should depend on the following ROS packages: rospy; std_msgs.

• Four additional folders must be created under the ar_week5_test folder: scripts; msg; srv; launch.

MESSAGES

• ***Two ROS messages*** should be created in the msg folder.

• ***1) cubic_traj_params***, which contains 6 real values (float): p0, pf, v0, vf, t0, tf; i.e. initial and final position, initial and final velocity, initial and final time.

• ***2) cubic_traj_coeffs***, which contains 6 real values (float): a0, a1, a2, a3, t0, tf; i.e. the four coefficients of a cubic polynomial trajectory, plus the initial and final time.

SERVICE

• ***One ROS Service*** called ***compute_cubic_traj*** should be created in the srv folder, which takes as input one full set of cubic trajectory parameters (initial and final position, initial and final velocity, initial and final time), and returns the four coefficients of the cubic polynomial trajectory (a0, a1, a2, a3).

LAUNCH FILE

• ***One ROS Launch file*** named ***cubic_traj_gen.launch*** should be created in the launch folder, that will start the four Nodes and the rqt_plot GUI automatically.

README FILE

• A text file named **README.txt** should be created in the package folder, that briefly explains the main steps needed to compile and use the package. The first line must include your name and student ID.

Part II --- The Nodes

• ***Four ROS nodes*** will need to be created in the scripts folder:

• ***1) points_generator.py*** This node should generate different random values (e.g. you can use the Python random.uniform() function for that) for initial and final position, p0, pf, initial and final velocity, v0, vf, initial and final time, t0, tf, every 20 seconds, and publish them on a ROS Topic using the cubic_traj_params message. Positions should not exceed a maximum/minimum value P_MAX= +/- 10; velocities should not exceed a mximum/minimum value V_MAX = +/- 10. For the time, t0 should always be 0, and tf should be tf = t0 + dt, with dt a random real number (i.e. float) between 5 and 10.

• ***2) cubic_traj_planner.py*** This node should subscribe to the ROS Topic created by Node 1, read the desired p0, pf, v0, vf, t0, tf published every 20 seconds, and should compute the a0,a1,a2,a3 coefficients of the cubic polynonial trajectory of the form p(t) = a0 + a1*t + a2*t^2 + a3*t^3 that best fit those requirements. To compute the coefficients, the Node should call a "compute_cubic_traj" Service made available by Node 3. Then, the Node should publish the a0,a1,a2,a3 coefficients and t0,tf time parameters on a ROS Topic, using the cubic_traj_coeffs message.

• ***3) compute_cubic_coeffs.py*** This Node should made the "compute_cubic_traj" Service available to any external node requesting it.

• ***4) plot_cubic_traj.py*** This Node should subscribe to the ROS Topic created by Node 2, read the a0,a1,a2,a3 coefficients and t0,tf time parameters published every 20 seconds, and publish three separate ROS topics: position trajectory, velocity trajectory and acceleration trajectory. These trajectories should be then visualized with the rqt_plot GUI, on the same plot, with different colors. The three trajectories should appear on the GUI at the same time, and last for tf seconds.

Part III --- The Video

• Take a 120 seconds screenshot video of your screen while you run the launch file, visualizing the ROS graph and the real-time plotting of the pos/vel/acc trajectories

• The screen should include:

- the terminal from which you are running the Launch file;
- the ros_graph showing the nodes, topics and their connections;
- the rqt_plot GUI showing the trajectories;
- a window showing your readme.txt file, with your name well visible.