Update to drcsim:

Wyatt Newman
October 19, 2013

There have been quite a few changes to drcsim and to the team code. The following summarizes these changes and additions.

Code reorganization:
The team code now runs on the "groovy" update of ROS. This is necessary to accommodate some of the new and anticipated changes in drcsim. The team code now lives within a subdirectory, "rosbuild." Much of the former code has been reorganized. This includes the location of the joint-names header, joint_names.h, which now resides in #include <atlas_conversions/joint_names.h>. One must also include in the respective package manifest the line:
<depend package="atlas_conversions"/> in order for the compiler to find this header directory.

Although the team code is more deeply nested, you can still use the "roscd" shortcut to find a given package, e.g.: *roscd jacobian_publisher* will take you to the the directory ros_workspace/team_hku.../rosbuild/robot_code/jacobian_publisher.

Drcsim 3.1:
In version 3.1 of drcsim, many improvements and fixes have been introduced. The robot model is more realistic (including an increase in mass from ~100kg to ~150kg). Mounting orientation of the Sandia hands has been corrected. Joint limit ranges for the forearms has been fixed to match Atlas. Changes to the mass model are likely more accurate (though this has not been confirmed, other than the net increase in weight).

The executable code no longer lives in "atlas_utils." Instead, the launch code resides in a new directory, "drcsim_gazebo." To launch the simulator, you can use:
VRC_CHEATS_ENABLED=1 roslaunch drcsim_gazebo atlas_v3_sandia_hands.launch

This will bring up a simulation of Atlas in an empty world (with the Sandia hands attached). You can alternative versions of the launch file in drcsim_gazebo/launch.

With the "cheats" enabled, you can pin the torso (as before) with the command:
rostopic pub --once /atlas/mode std_msgs/String '{data: "pinned_with_gravity"}'

rqt_plot:
With the "groovy" update, you can use an alternative plotting tool, rqt_plot. (See: http://wiki.ros.org/rqt_plot). This tool is an improvement over rxplot. It stalls less often, and it is easier to add or delete topic items to plot. You can launch it by typing: *rqt_plot*. (You can ignore the various warnings and wait while it starts up).

A textbox helps with guiding the user regarding what topics are available. For example (with drcsim running), one can start typing in the "Topic" box: "/atlas" and a drop-down will show the various topics under "atlas" that are currently being published. rqt_plot will also remember what topics were being plotted if you kill rqt_plot and restart it.

New packages/code:
A few new packages will be immediately useful. These include a revised low-level joint controller, a gravity torque publisher and a Jacobian publisher.

Gravity-torque publisher: The package copm_calc (under rosbuild/robot_code) contains a node that subscribes to Atlas' state and computes the gravity loads necessary to maintain static equilibrium of arm poses for any pose. At present, this module does not compute gravity compensation for the leg joints. Further, it does not yet account for the weight of the Sandia hands. Nonetheless, the estimates published are useful in assisting control of the arms.

This node can be run by entering: *rosrun copm_calc fc_com_calc.* It will then publish to the topic "counter_gravity/torques". The values published may be added to the control torques within low-level joint control to augment the PD feedback control, resulting in more precise arm control.

The same node also publishes under /com and /cop, which express center of mass (or center of pressure) for the current robot pose. This will be useful for balance control.

jacobian_publisher: The new package "jacobian_publisher" creates a node "jacobian_publisher". As normal, it can be launched with: *rosrun jacobian_publisher jacobian_publisher*. Within the jacobian_publisher package, there is a README with instructions and reference to example source code that subscribes to the jacobian publications.

low_level_servo_control: Within the package "simple_atlas_interface" (under rosbuild/elec7078_newman/simple_atlas_interface), there is a modified low-level joint controller called "low_level_servo_control". This controller is still under development. It is an attempt to bring closer agreement between drcsim and Atlas. It subscribes to the gravity-torque publications and uses this information to assist in arm control. Different gain options are available, and the user may select from among these. (drcsim 3.1 is the best current choice for drcsim).

This control code is deliberately slowed down to 333Hz, to match the update rate of Atlas. Further, feedback is computed within this routine, instead of by setting gains to be used within Atlas' eBox or within the drcsim/gazebo equivalent. The reason for this is so that nonlinear control options can be exploited, including gravity-torque compensation, Coulomb-friction compensation, modulation of control gains as a function of pose, and anticipated inertial decoupling computations.

Run this code with: *rosrun simple_atlas_interface low_level_joint_servo.* It will accept commands (as before) via a message type: hku_msgs::lowLevelJointController and on the topic: /lowLevelJntCmd. A relatively recent change is that this message now includes acceleration commands (in addition to desired angles and desired velocities).

Changes are anticipated to this low-level controller. Users should avoid making edits in this package, to avoid merge conflicts with future updates.