Introduction to Programming with HERB

HERB Lab

2013-09-06

1 Simulation

1.1 Herbpy Console

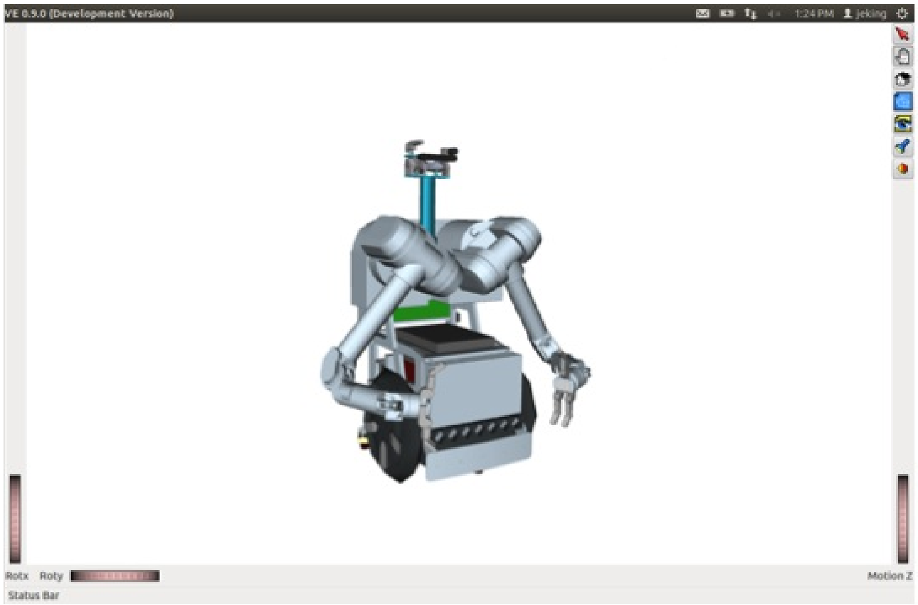
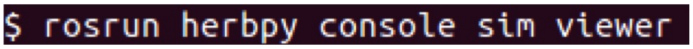


Figure 1: The herbpy viewer

The easiest way to work with HERB in simulation is via the herbpy console. To start up a herbpy console in simulation mode type the following from a command line:



```
$ rosrun herbpy console sim viewer
```

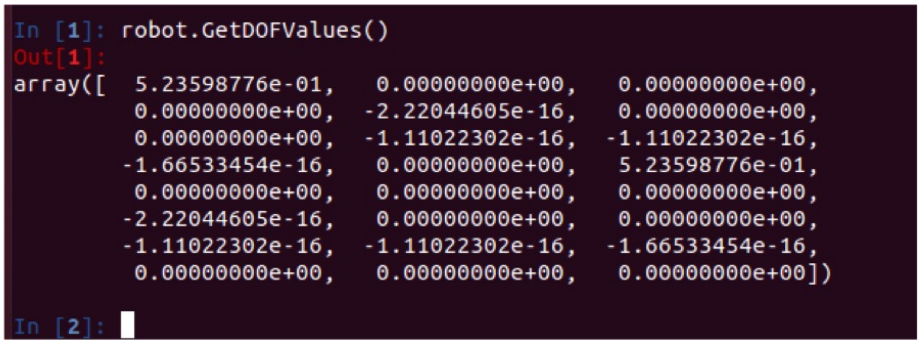
Here the parameter `sim` indicates you would like to start in simulation mode. The parameter `viewer` indicates that a viewer should be attached to the instance. Figure 1 shows an example of the viewer.

An additional parameter that is useful is `debug`. This will set the logging level to DEBUG. By default, the log level is set to INFO.

Starting the herbpy console will drop into an ipython prompt. Two useful objects have been initialized: `robot` and `env`.

1.1.1 `robot`

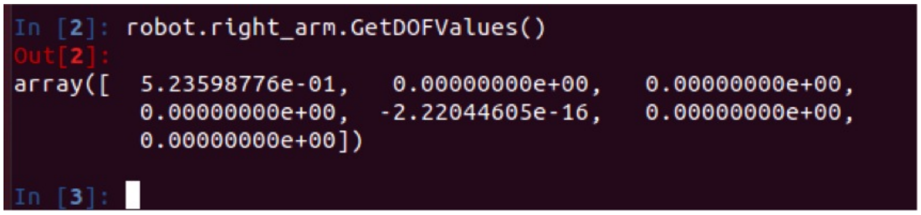
The `robot` objects is a pointer to the model of the robot, in this case HERB. Many operations on the robot are available. For example, the following will give the current value of all 24 DOF values for the robot:



```
In [1]: robot.GetDOFValues()
Out[1]:
array([  5.23598776e-01,   0.00000000e+00,   0.00000000e+00,
         0.00000000e+00,  -2.22044605e-16,   0.00000000e+00,
         0.00000000e+00,  -1.11022302e-16,  -1.11022302e-16,
        -1.66533454e-16,   0.00000000e+00,   5.23598776e-01,
         0.00000000e+00,   0.00000000e+00,   0.00000000e+00,
        -2.22044605e-16,   0.00000000e+00,   0.00000000e+00,
        -1.11022302e-16,  -1.11022302e-16,  -1.66533454e-16,
         0.00000000e+00,   0.00000000e+00,   0.00000000e+00])

In [2]:
```

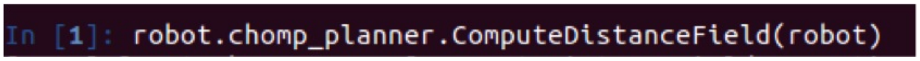
Pointers exist for each arm of the robot via calls to `robot.left_arm` and `robot.right_arm`. The following command gets DOF values for only the right arm:



```
In [2]: robot.right_arm.GetDOFValues()
Out[2]:
array([  5.23598776e-01,   0.00000000e+00,   0.00000000e+00,
         0.00000000e+00,  -2.22044605e-16,   0.00000000e+00,
         0.00000000e+00])

In [3]:
```

Several planners exists for planning motions for the arms. A common one, developed in this lab, is the CHOMP planner. Prior to using the CHOMP planner, the distance field must be initialized. The following command performs this initialization:



```
In [1]: robot.chomp_planner.ComputeDistanceField(robot)
```

2 Real Life

3 Python Scripting