Antonio Rafael V. Umali
CS 525 HW 1

I. Book Questions

A. Ch.4, #5:

Assuming that the cylindrical rod can translate and rotate freely in $\mathbb{R}^3$, then its C-space is:

$$\mathbb{R}^3 \times \mathbb{RP}^3$$

as seen in equation 4.31.

However, the problem states that rotation about the central axis does not induce any change in the cylinder's position and orientation, hence one degree of freedom and one dimension of the C-space is lost. This dimension is taken from the rotational space, so the C-space becomes:

$$\mathbb{R}^3 \times \mathbb{RP}^2$$

which is homeomorphic to

$$\mathbb{R}^3 \times S^2$$

by the argument in section 4.1.2, with the antipodal points identified.

B. Ch.4, #8:

The spacecraft can translate and rotate in 2D space, so assuming that there are no loops in translation in the task space, then the C-space would be:

$$\mathbb{R}^2 \times S^1$$

However, the problem defines the top and bottom parts of the screen to be identified, hence causing a loop in position when the craft goes past the screen limits. The C-space would then be

$$T^2 \times S^1$$

which is just:

$$S^1 \times S^1 \times S^1 = T^3$$

C. Ch.4, #14:

a.

h_1 express a rotation of $-\pi/2$ around the axis given by the vector $[1/\sqrt{3},\ 1/\sqrt{3},\ 1/\sqrt{3}]$. Using angle-axis to quaternion equations:

$$q_x = a_x * \sin(\theta/2) = 1/\sqrt{3} * -\sqrt{2}/2 = -\sqrt{6}/6,$$
$$q_y = a_y * \sin(\theta/2) = 1/\sqrt{3} * -\sqrt{2}/2 = -\sqrt{6}/6,$$
$$q_z = a_z * \sin(\theta/2) = 1/\sqrt{3} * -\sqrt{2}/2 = -\sqrt{6}/6,$$
$$q_w = \cos(\theta/2) = \sqrt{2}/2$$

and it can be verified that

$$q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1$$

b.

h_2 express a rotation of π around the axis given by the vector $[0, 1, 0]$. Using angle-axis to quaternion equations:

$$q_x = a_x * \sin(\theta/2) = 0 ,$$
$$q_y = a_y * \sin(\theta/2) = 1 * 1 = 1 ,$$
$$q_z = a_z * \sin(\theta/2) = 0 ,$$
$$q_w = \cos(\theta/2) = 0$$

and it can be verified that

$$q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1$$

c.

Rotation by h_1 is followed by the rotation of h_2, this would produce the quaternion $h_2 h_1$, which, by quaternion multiplication, is

$$Let\ h_i = [h_{is}, h_{iv}],\ then\ h_i h_j\ is:$$
$$[h_{is}h_{js} - dot(h_{iv}, h_{jv}),\ h_{is}h_{jv} + h_{js}h_{iv} + cross(h_{iv}, h_{jv})$$

thus for $h_2 h1$,

$$h_{2s}h_{1s} = 0$$
$$dot(h_{2v}, h_{1v}) = -\sqrt{6}/6$$
$$h_{2s}h_{1v} = (0, 0, 0)$$
$$h_{1s}h_{2v} = (0, \sqrt{2}/2, 0)$$
$$cross(h_{2v}, h_{1v}) = (\sqrt{6}/6, 0, -\sqrt{6}/6)$$

$h_2 h1 = [q_w = \sqrt{6}/6, q_x = \sqrt{6}/6 , q_y = \sqrt{2}/2, q_z = -\sqrt{6}/6]$,

which has a norm of 1, constructing the angle-axis representation yields

$$\theta = 2 * \cos^{-1}(q_w) = 2.3005$$
$$x = q_x/\sqrt{(1 - q_w^2} = 0.4472 = \sqrt{5}/5$$
$$y = q_y/\sqrt{(1 - q_w^2} = 0.7746 = \sqrt{15}/5$$
$$z = q_z/\sqrt{(1 - q_w^2} = -0.4472 = -\sqrt{5}/5$$
$$x^2 + y^2 + z^2 = 1$$

D. Ch.4, #16:

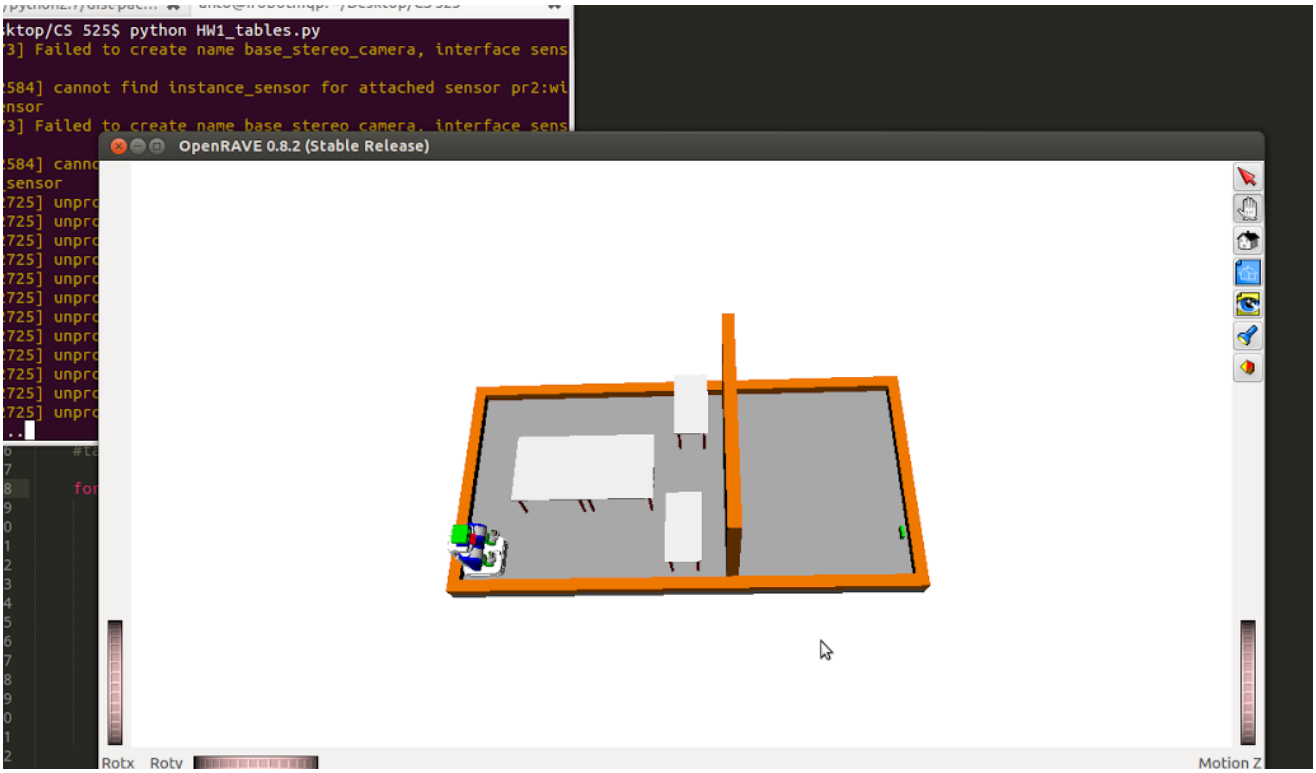
The five polyhedral bodies float freely in a 3D world, and capable of both translation and rotation. They are disjoint but form one composite robot. The C-space would be:

$$(\mathbb{R}^3 \times \mathbb{RP}^3) \times (\mathbb{R}^3 \times \mathbb{RP}^3) \times (\mathbb{R}^3 \times \mathbb{RP}^3) \times (\mathbb{R}^3 \times \mathbb{RP}^3) \times (\mathbb{R}^3 \times \mathbb{RP}^3)$$

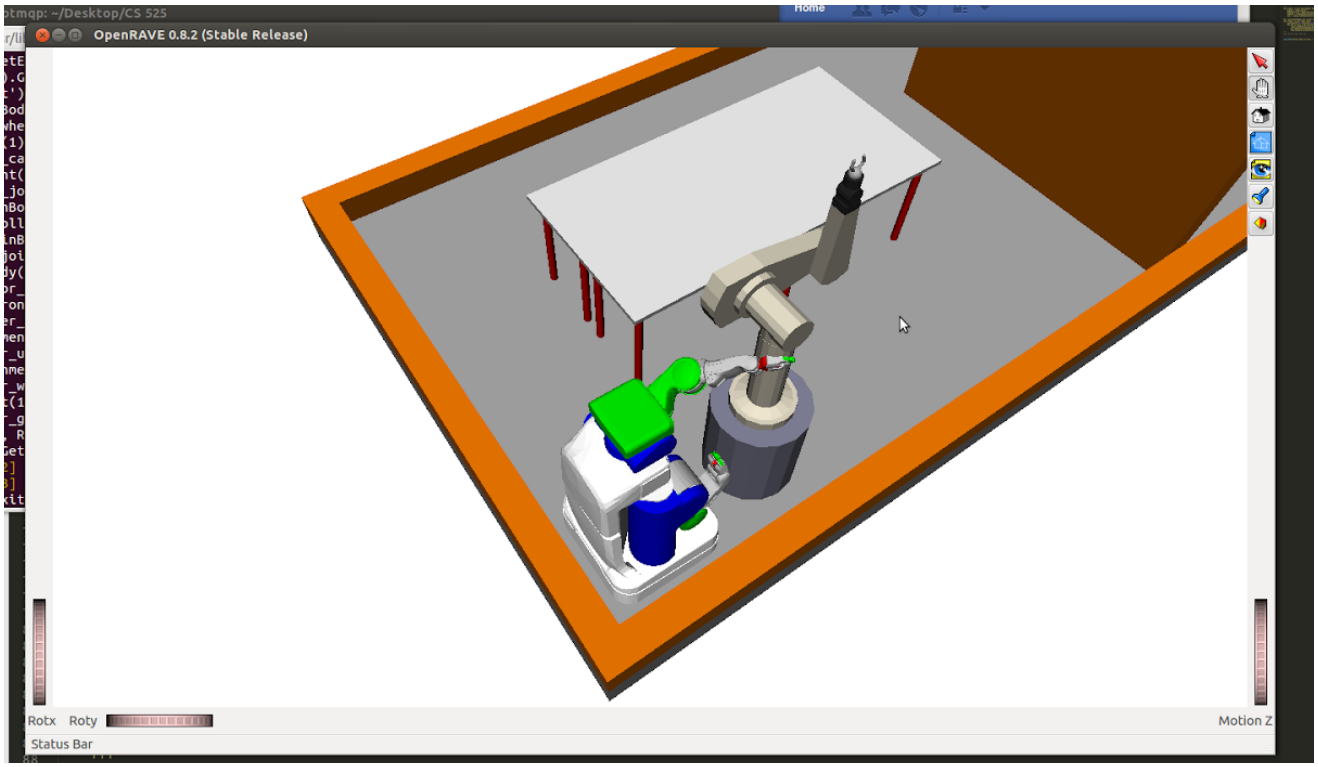
The Cartesian product adds up the dimensions of the individual spaces. With $\mathbb{R}^3 \times \mathbb{RP}^3$ having a dimension of 6, then the dimension of the final C-space is $5 * 6 = 30$

II. Implementation screenshots and writeup

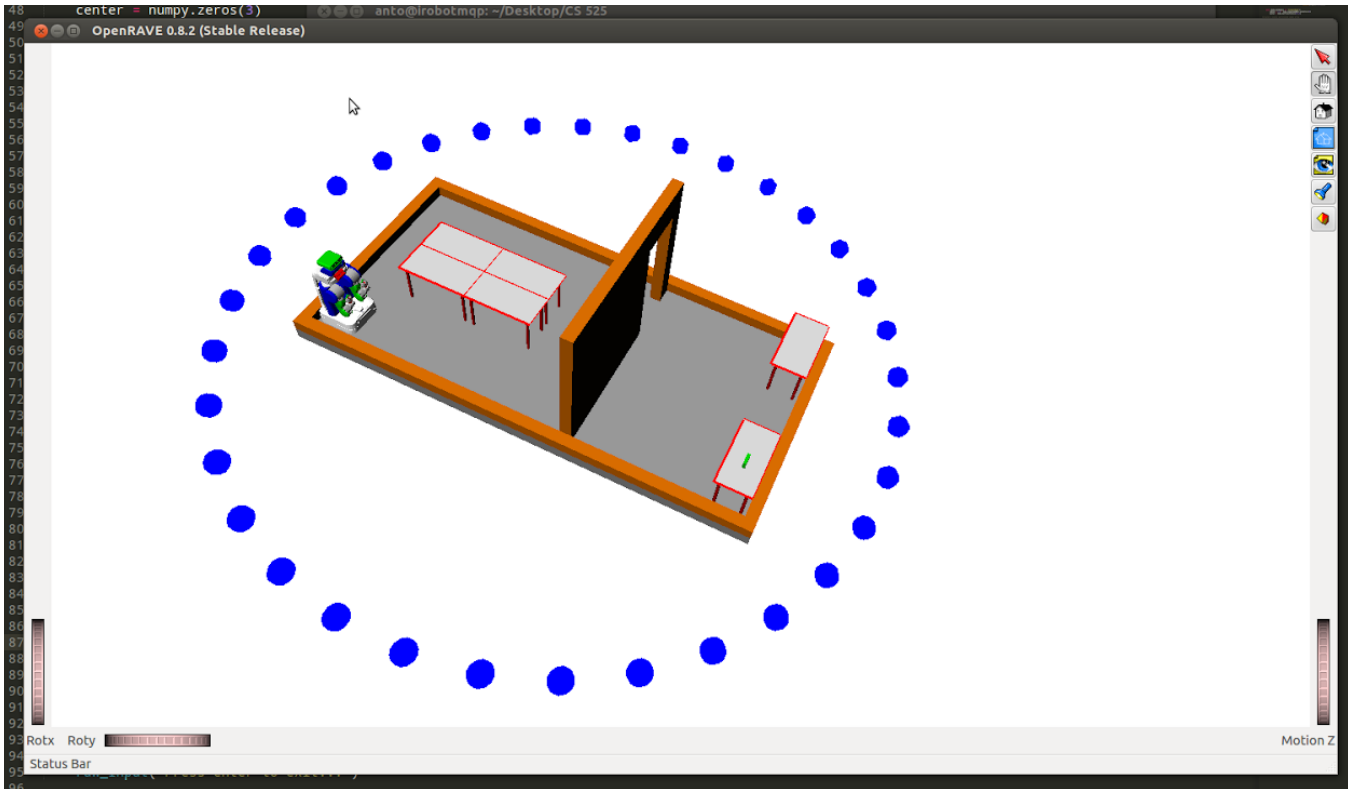
A.


ktop/CS 525$ python HW1_tables.py
3] Failed to create name base_stereo_camera, interface sens
584] cannot find instance_sensor for attached sensor pr2:wi
nsor
3] Failed to create name base_stereo_camera, interface sens
584] canno
_sensor
725] unpro
OpenRAVE 0.8.2 (Stable Release)
Rotx
Roty
Motion Z

B.


tmqp: ~/Desktop/CS 525
Home
OpenRAVE 0.8.2 (Stable Release)
Rotx
Roty
Motion Z
Status Bar

C.


center = numpy.zeros(3)
anto@irobotmqp: ~/Desktop/CS 525
OpenRAVE 0.8.2 (Stable Release)
Rotx
Roty
Motion Z
Status Bar

D.

I placed waitrobot() outside the with env: block. This is because waitrobot() simply waits for the controller to finish moving the robot. Placing this inside the with env: block would not move the robot, as all KinBody() objects are locked at that point, robot included. Thus its state will not change. Placing waitrobot() after processes which the user wants to change after the robot has moved will also cause problems, as these processes will occur before the robot finishes moving.