6.881: Problem Set #1a

Due on Monday, September 10, 2018 at 22:00. See course website for submission details. Use Drake release tag `drake-20180906`, i.e. use this notebook via `./docker_run_notebook.sh drake-20180906 .`, or whichever script you need for your platform.

Before attempting the problems in this notebook, please make sure that you've installed docker and drake (https://manipulation.csail.mit.edu/install_drake_docker.html).

A useful guide written by Tomas and Russ for getting up to speed on Drake can be found here (https://docs.google.com/document/d/17KlNJqqcAEqY1DqiezUaEtami4l_v55OV3qagsp9Q5U/edit?usp=sharing). We've enabled comments, so please ask in the document if you have any questions/confusions/suggestions.

There are three files in this handout:

- A jupyter notebook `pset_1a.ipynb`(this file), which contains questions you'll need to answer and code snipptes to run simulations.
- `kuka_multibody_sim.py`. You'll need to modify this file as per instructions in this notebook, and **submit it to Gradescope for grading**.
- `test_pset_1a.py` This is the test script run on Gradescope that determines how many points you score. You can run the same test script by running the cell at the bottom of this notebook. Do not modify the test script.

In [1]:
```
# Import stuff to be used in this notebook. Run this first!
%reload_ext autoreload
%autoreload 2
import numpy as np
from kuka_multibody_sim import (SimulateRobotFreeFall)
import meshcat
```

Open meshcat visualizer

In [2]:

```
# If you interrupt the kernel of this notebook, you'll need to run this cell a
gain to
# restart the meshcat server, and then refresh the visualization window.

# This will open a mesh-cat server in the background, click on the url to disp
lay visualization in a seperate window.
vis = meshcat.Visualizer()

# Duplicate the visualization inside this cell (at a lower resolution).
vis.jupyter_cell()
```

```
You can open the visualizer by visiting the following URL:
http://127.0.0.1:7000/static/
```

Out[2]:

Free-falling robot

Run the cell below to watch a passive robot fall under gravity in the meshcat visualizer window you just opened.

Simulation in drake starts with the creation of a `MultiBodyPlant`, which is a description of the geometry, inertial properties, and kinematic relationships of all rigid bodies to be simulated.

The `MultiBodyPlant` is then added to a Drake `Diagram`, which includes utility systems for plotting and logging. Finally, a `Simulator` system is created from the `Diagram`, and `Simulator.StepTo(simulation_duration)` is called to actually run the simulation.

After watching the simulation starting from the default initial conditions, you may wonder why part of the robot is "inside" the table at the end of the simulation. This is because the geometries used for contact detection (contact geometry) and visualization (visual geometry) are usually different. In order to run simulations faster, the contact geometry of some rigid bodies are usually simplified (from the visual geometry) or even ignored.

In this particular simulation, only the gripper and the table tops (black part of the table) have contact geometry.

In [3]:
```
apple_initial_position_in_world_frame = np.array([-0.2, -0.2, 10.])
robot_initial_joint_angles = np.zeros(7)
robot_initial_joint_angles[0] = np.pi
state_log, mbp = SimulateRobotFreeFall(
    apple_initial_position_in_world_frame,
    robot_initial_joint_angles,
    sim_duration=4.0, real_time_rate=1.0)
```

```
Warning: if you have not yet run meshcat-server in another terminal, this wil
l hang.
You can open the visualizer by visiting the following URL:
http://127.0.0.1:7000/static/
```

Your task

Find `apple_initial_position_in_world_frame` and `robot_initial_joint_angles`, such that the system satisfies the following requirements at $t = 4.0s$ (simulation time):

- both the apple and the robot's end effector are on the table to which the robot is NOT bolted.

You can try out different initial values in the cell above, but for running the test and submission, you will need to **overwrite values of global variable `your_apple_initial_position_in_world_frame` and `your_robot_initial_joint_angles` at the bottom of `kuka_multibody_sim.py` with your answer**.

Run Tests

Running the cell below will print out a score for every test (there's only one this time) and the total score.

In [6]:
```
import os
# Run the tests
os.popen("python test_pset_1a.py test_results.json")

# Print the results json for review
import test_pset_1a
print test_pset_1a.pretty_format_json_results("test_results.json")
```

```
Test apple_and_robot_end_effector_are_on_the_designated_table: 0.00/5.00.
  * Test Failed: Apple is not on the desired table.

TOTAL SCORE (automated tests only): 0.00/5.00
```