CS 4/6545 Autonomous Robotics:
Simulating the dynamics of a box undergoing compliant contact

Preface: why this assignment? Contact is the key to manipulation, and the only way that Robotics currently knows how to do any manipulation is by modeling that contact (to some degree of fidelity). Contact is difficult to model quickly, accurately, and robustly. Previous assignments have focused on control of robotic systems. *This assignment will focus on modeling a "robotic" (rigid body) system.*

We will use MATLAB to simulate a $1m$ tall by $1m$ wide box contacting a "ground plane" (the line segment $y = 0$) in 2D. While this scenario seems very simple, these assumptions (planar ground, simple rigid body, 2D, compliant contact) are often used in Robotics.

Read all instructions carefully. Your submission will consist of all code and plots.

1. Implement the MATLAB function `drot(.)` (located in the file `drot.m`), which will return the 2×2 time derivative of a 2×2 rotation matrix; differentiate the matrix in `rot.m` with respect to time (don't forget that θ is a function of time!) This problem should give you a fairly gentle introduction to MATLAB syntax. You will need this function in step #3.

   Test your function: Do this by using two nearby values of θ, which I will call θ_1 and θ_2. We can then define $\dot{\theta}$ (again, for testing purposes) as $\theta_2 - \theta_1$. Does $\texttt{drot}(\theta_1, \dot{\theta}) \cong R(\theta_2) - R(\theta_1)$?

   Explain how I arrived at the test above.

2. Fill out the necessary parts of `boxode.m`. Like `springode.m` that we developed in class, `boxode.m` evaluates an ordinary differential equation *and* the 2nd order ODE is converted to a system of first order ODEs:

$$\mathbf{M}\ddot{\boldsymbol{x}} = \boldsymbol{f} \tag{1}$$

   becomes

$$\mathbf{M}\dot{\boldsymbol{v}} = \boldsymbol{f} \tag{2}$$
$$\dot{\boldsymbol{x}} = \boldsymbol{v} \tag{3}$$

   where $\mathbf{M}$ is the 3×3 *generalized inertia* matrix, $\boldsymbol{x}$ is a three dimensional vector describing the position (two components) and orientation (one component) of the box in 2D, and $\boldsymbol{f}$ is the three dimensional vector of all forces on the box. ***NOTE: I have included a file `boxode.m.hints` in the archive to help you along, if you want said help.***

3. Test your simulation of the box by simulating the box. Call the MATLAB function `simulate(.)` as follows from the MATLAB prompt (make sure MATLAB's working directory has your `.m` files in it):

```
>> [t, x] = simulate();
```

   Find reasonable gain constants that make the box sit on the ground. Plot the height of the box using:

```
>> plot(t, x(2))
```

4. Give the box some initial horizontal and vertical velocities and plot the position of the box again (start velocity values near zero). Frictional forces should cause the box to eventually come to a halt. Does it? (You may have to simulate for longer than ten seconds to see this; you can change how long the simulation runs by modifying `simulate.m`). Use:

   ```
   >> plot(t, x(1));
   ```

   to plot the horizontal position of the box and the command from above to plot the vertical position of the box. *Hint:* you likely don't want a corner of the box to leave the ground plane (as it is likely to do under certain conditions).