Contents

Simulation: This is the simulation second draft. Physics engine will run inside this

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%% Version 1_1 attempts to give the cylinder    %%%%%%%%%
%%%%%%%%% actual dimensions and orient it around the   %%%%%%%%%
%%%%%%%%% center of mass so that position tracking     %%%%%%%%%
%%%%%%%%% and calculation of inertia all use the same  %%%%%%%%%
%%%%%%%%% frame                                        %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc
clear all
close all

% Define frames: need to make these globals
global Nframe
global dt
global T

%define simulation space frame
Nframe = [1 0 0; 0 1 0; 0 0 1];

T = 7.25;                       % duration of simulation
dt = 0.01;
t = 0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% STUFF THAT WILL HAPPEN IN THE GUI %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%define initial conditions
phi_0 = 0;  %initial spin orientation
psi_0 = 0;  %initial precession orientation
theta_0 = 20;  % The newtation angle, it is between the spin axis and the angular momentum ve
ctor
phi_dot = 10; % rotational rate in radians per second
psi_dot = 10.66; %rad/sec
theta_dot = 0;
%define object and load initial conditions into its properties
object1 = Cylinder;
object1.mass = 1; object1.radius = 4; object1.height = 20;
object1.orientation = [phi_0; psi_0; theta_0];
[X,Y,Z] = cylinder([object1.radius object1.radius object1.radius],40);
points = pointcloud(X,Y,Z*object1.height); points(3,:) = points(3,:) - object1.height/2;

%define point on the object we want to track as a vector from the origin to
%that point
          R_na = [cosd(object1.orientation(2)) -sind(object1.orientation(2)) 0;        %Prece
ssion rotation matrix
              sind(object1.orientation(2)) cosd(object1.orientation(2)) 0;
```

```
              0 0 1];
          R_ag = [1 0 0;                                                    %Newtation rot
ation matrix
                  0 cosd(object1.orientation(3)) -sind(object1.orientation(3));
                  0 sind(object1.orientation(3)) cosd(object1.orientation(3))];
          R_gb = [cosd(object1.orientation(1)) -sind(object1.orientation(1)) 0;        %Spin
rotation matrix
                  sind(object1.orientation(1)) cosd(object1.orientation(1)) 0;
                  0 0 1];
           R_nb = R_na*R_ag*R_gb;                %Matrix from N-->B
object1.position = R_nb*points;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% BEHIND THE SCENES %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Conversions of spin rates and entered into class phelp roperties
phi_dot = (180/pi)*phi_dot; %deg per second
% psi_dot = 0; %rad/sec
psi_dot = (180/pi)*psi_dot; %deg per sec
theta_dot = (180/pi)*theta_dot; %deg per sec
object1.spin_rates = [phi_dot; psi_dot; theta_dot];
```

Loop

```
%this for loop will eventually be "while Applied_Force or
%Applied_Moment == 0" and will live inside another for loop called "while sim = true"
%which can perhaps be canceled from the GUI with an off switch or something
Position = zeros(3,123,T*100+1);        %preallocate position vector as a function of the tot
al simulation time
n = 1;
nn = 1;
Position(:,:,n) = object1.position;   %and define initial position
fprintf('Runtime for a simulation length of %f seconds\n',T)
tic
while t <= T        %want this to eventually be "While objectX exists, obey the laws of physi
cs"
    t = t + dt;     %define the time stamp at which we will calculate
    n = n + 1;
    %disp('time incremented')
    [object1.orientation, Position(:,:,n)] = object1.freerotate(dt,Nframe);
```

ADD THE BELOW IN FOR A NICE MOVIE, BUT PROGRAM RUNTIME WILL INCREASE %%%

```
%     plot3(Position(1,:,nn),Position(2,:,nn),Position(3,:,nn),'b.'); axis equal; hold on;
%     plot3(Position(1,101,nn),Position(2,101,nn),Position(3,101,nn),'r-o'); hold off
%     M(nn) = getframe(gcf);
%     nn = nn + 1;

end
toc
```

```
Runtime for a simulation length of 7.250000 seconds
Elapsed time is 0.059111 seconds.
```

Published with MATLAB® R2016a