Homework 3 - Problem 2

Coded by Michael White

```
clear;clc;

% Setup Portion
    % Dynamic equations using Newton-Euler formulation
    syms d_2 dDot_2 dDotDot_2 theta1 thetaDot_1 thetaDotDot_1 theta2 thetaDot_2 thetaDotDot_2 g Ixx_1 Iyy_1 Izz_1 m1 m2;
    linkTable = [0 0 0 theta1; -pi/2 0 d_2 0];

    % Define transforms for each link from table
    T01 = functions.links.Link2Transform(linkTable(1,:));
    T12 = functions.links.Link2Transform(linkTable(2,:));

    % Pull and define rotations from transforms
    R01 = functions.transform.rotationFromTransform(T01);
    R12 = functions.transform.rotationFromTransform(T12);

    % Define position, centroid, inertial, and initial angular velocity/accel vectors
    P_01 = functions.transform.positionFromTransform(T01);
    P_12 = functions.transform.positionFromTransform(T12);
    Pc_11 = [0 0 0].';
    Pc_22 = [0 0 0].';
    Ic_11 = [Ixx_1 0 0; 0 Iyy_1 0; 0 0 Izz_1];
    Ic_22 = 0;
    w_0 = 0;
    wDot_0 = 0;
    v0_dot = [0 0 g].';

% Velocity Propogation:
    % Define velocity conditions at first joint
    w_11 = functions.dynamics.omega_ip1ip1(R01.',w_0,thetaDot_1);
    wDot_11 = functions.dynamics.omegaDot_ip1ip1(R01.',wDot_0,w_0,thetaDot_1,thetaDotDot_1);
    vDot_11 = functions.dynamics.vDot_ip1ip1(R01.',wDot_0,P_01,w_11,v0_dot);
    vcDot_11 = functions.dynamics.vcDot_ip1ip1(wDot_11,Pc_11,w_11,vDot_11);

    % Define force and torque conditions at first joint
    F_11 = functions.dynamics.F_ip1ip1(m1,vcDot_11);
    N_11 = functions.dynamics.N_ip1ip1(wDot_11,w_11,Ic_11);

    % Define velocity conditions at second joint
    w_22 = functions.dynamics.omega_ip1ip1(R12.',w_11,0);
    wDot_22 = functions.dynamics.omegaDot_ip1ip1(R12.',wDot_11,w_11,0,0);
    vDot_22 = functions.dynamics.vDot_ip1ip1_prism(R12.',wDot_11,P_12,w_11,vDot_11,w_22,dDot_2,dDotDot_2);
    vcDot_22 = functions.dynamics.vcDot_ip1ip1(wDot_22,Pc_22,w_22,vDot_22);

    % Define force and torque conditions at second joint
    F_22 = functions.dynamics.F_ip1ip1(m2,vcDot_22);
    N_22 = functions.dynamics.N_ip1ip1(wDot_22,w_22,Ic_22);

% Force Propogation:
    % Summarize force and torque conditions at second joint
    f_22 = F_22;
    n_22 = functions.dynamics.n_ii(N_22,0,0,Pc_22,F_22,Pc_22,0);
    tau_2 = functions.dynamics.tau_i(F_22);

    % Summarize force and torque conditions at first joint
    f_11 = functions.dynamics.f_ii(R12,f_22,F_11);
    n_11 = functions.dynamics.n_ii(N_11,R12,n_22,Pc_11,F_11,P_12,F_22);
    tau_1 = functions.dynamics.tau_i(n_11);

% Cleanup tau_1 and tau_2
    syms c1 c2 s1 s2;
    tau_1 = subs(tau_1,[cos(theta1),cos(theta2),sin(theta1),sin(theta2)],[c1,c2,s1,s2]);
```

```
    tau_2 = subs(tau_2,[cos(theta1),cos(theta2),sin(theta1),sin(theta2)],[c1,c2,s1,s2]);
    display(tau_1);
    display(tau_2);
```

```
tau_1 =

Izz_1*thetaDotDot_1 + d_2*m2*(2*dDot_2*thetaDot_1 + d_2*thetaDotDot_1)


tau_2 =

m2*(- d_2*thetaDot_1^2 + dDotDot_2)
```

Published with MATLAB® R2020b