```
function out = RK4_RigidBody_MRP(x0, u0, I, t0, dt, tf)
% Function that implements the Runga-Kutta 4 algorithm to integrate the
% rigid body EOM in terms of MRPs.
%   Inputs:
%       - x0: Initial state vector, with w written in body coordinates
%               [sig1_0; sig2_0; sig3_0; w1_0; w2_0; w3_0]
%       - u0: Initial control vector, with L written in body coordinates
%               [L1_0; L2_0; L3_0]
%       - I: Inertia matrix of the rigid body
%       - t0: Time that integration will start, in seconds
%       - dt: Time step for integration, in seconds
%       - tf: Time that integration will stop, in seconds
%
%   Outputs:
%       - out: Integration output matrix, each column is a vector with the
%              same number of elements n as there were timesteps
%               [t (nx1), sigma (nx3), w (nx3), L (nx3)]
%
    X = x0;
    t = t0;

    out = zeros(length(t0:dt:tf), 10);
    out(1,:) = [t0, x0', u0']; % t, sig(1:3), w(1:3), L(1:3)
    k = 1;

    while t < tf - dt
        L = zeros(3,1);

        k1 = dt*rigidBodyEOM_MRP(X,L,I);
        k2 = dt*rigidBodyEOM_MRP(X+k1/2,L,I);
        k3 = dt*rigidBodyEOM_MRP(X+k2/2,L,I);
        k4 = dt*rigidBodyEOM_MRP(X+k3,L,I);

        X = X + (1/6)*(k1 + 2*k2 + 2*k3 + k4);

        if norm(X(1:3)) >= 1
            X(1:3) = -X(1:3)/norm(X(1:3))^2;
        end

        t = t + dt;
        k = k + 1;

        out(k,:) = [t, X', L'];

    end

end
```

Published with MATLAB® R2023b