Contents


```
function [ dx ] = ode1linkTracking_robust( t,x,param )
```

Implementation

```
        %Extracting the coefficients of the trajectory
        a1=param(1,:);

        %Approximated dynamics
        I0=100;
        mgd0=200;
        fv0=100;

        %Create the actual trajectory
        vec_t = [1; t; t^2; t^3];
        theta_d= [a1*vec_t];

        % compute the velocity and acceleration in both theta 1 and theta2.
        a1_vel = [a1(2), 2*a1(3), 3*a1(4), 0];
        a1_acc = [2*a1(3), 6*a1(4),0,0 ];


        % compute the desired trajectory (assuming 3rd order polynomials for trajectories)
        dtheta_d =[a1_vel*vec_t];
        ddtheta_d =[a1_acc*vec_t];
        theta= x(1);
        dtheta= x(2);
```

```
Not enough input arguments.

Error in ode1linkTracking_robust (line 5)
        a1=param(1,:);
```

Robust controller

True dynamics of the system

```
        I=7.2;
        mgd=1;
        fv=1;


        % Assign the gain values
        KP=10;
        KD=2;
```

```
        K=[0 1;-KP -KD];

        x=[theta-theta_d;dtheta-dtheta_d];
        aq=ddtheta_d-KP*x(1)-KD*x(2);
        n=(1/I)*(I0*aq + (fv0-fv)*dtheta+ (mgd0-mgd));
        p=[1 0;0 1];
        beta=1;
        p=1;


        v=-transpose(x)*p*[0;1]*p/abs(transpose(x)*p*[0;1]);
```

Calculate dx based on the dynamics of the robot and the error

```
        dx=K*x+[0;1]*v + n;
```

```
end
```

Published with MATLAB® R2018b