1_4_A_Co-Design_Example

Problem Description

Control System

Consider the following dynamics of suspension system

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -8 & -4 & 8 & 4 \\ 0 & 0 & 0 & 1 \\ 80 & 40 & -160 & -60 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 80 \\ 20 \\ -1120 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\dot{x} = Ax + Bu, y = Cx$$

- There are **4 states**. x_1 and x_2 represents position and velocity of the car and x_3 and x_4 represents position and velocity of the mass of the suspension system.
- Control **input** u represents the force applied to the car by the suspension system.
- Position of the car is the **output**.
- The suspension system has a settling time requirement of 1.5sec.

Tasks

The active suspension controller has to be implemented on an Electronic Control Unit (ECU) where a number of other periodic real-time tasks are also running. The real-time tasks are characterized as follows:

Tasks	Pi (ms)	Di (ms)	ei (ms)
T1	10	10	3
T2	15	15	4
T3	25	25	4

Constraints

- Due to thermal constraint, the maximum processor utilization is $U_{max} = 0.8$

- The sensor-to-actuator delay of the control application must be constant and must not exceed 50% of the chosen sampling period
- Assume that the sensing operation by the sensor task takes negligible time. Also, the actuation takes negligible time. Both sensing and actuating operations are performed periodically on a separate microcontroller in a time-triggered fashion.
- The controller task of the control application has a WCET $e_c = 2ms$.

Requirement

Design

1. sampling period h of the controller such that the utilization limit is not violated
2. the scheduling policy on the control and real-time tasks such that real-time tasks meet theirs deadline and controller task meets its sensor-to-actuator delay constraint
3. the controller such that the suspension system is able to settle in 1.5sec

Solution

```
%%%%%%%%%%%%%%%%
% CoDesign Example in the Slide L3_Controller_Design
%%%%%%%%%%%%%%%%

%% System Parmeter
A = [0 1 0 0;
    -8 -4 8 4;
    0 0 0 1;
    80 40 -160 -60];
B = [0 80 20 -1120]';
C = [1 0 0 0];

nx=4;
nu=1;

%% Utilization Design

res = 0.8 - 3/10 - 4/15 - 4/25; % ans: 0.0733
sp_lb = 2 / res; % ans: 27.2727

% so we choose 30ms as sampling period

%% Decide Scheduling Method

% Rate-Monotonically Method

T = [10, 10, 3, 1;
    15, 15, 4, 2;
    25, 25, 4, 3;
    30, 15, 2, 4];

Time_Response = ResponseTimeAnlaysis(T); % 3 7 14 -1 violate

% Deadline-Monotonically Method
T = [10, 10, 3, 1;
```

```
    15, 15, 4, 2;
    25, 25, 4, 4;
    30, 15, 2, 3];

Time_Response = ResponseTimeAnlaysis(T) % 3 7 20 9 perfect!

%% Controller Design

% Discretize System
sys.A = A;
sys.B = B;
sys.C = C;

sys.h = 30 * 1e-3;
sys.tau = 9 * 1e-3;

sysc = ss(sys.A, sys.B, sys.C, 0);
sysd = c2d(sysc, sys.h);

sys.dA = sysd.a;
sys.dB = sysd.b;
sys.dC = sysd.c;
sys.dD = sysd.D;

sys.phi=expm(sys.A*sys.h);
syms s

sys.gamma_0=inv(A)*(expm(sys.A*(sys.h-sys.tau))-expm(sys.A*0))*B;
sys.gamma_1=inv(A)*(expm(sys.A*sys.h)-expm(sys.A*(sys.h-sys.tau)))*B;

% because tau << h, here we use approximate gamma
% sys.gamma_1 = (sys.h - sys.tau) * sys.B;
% sys.gamma_0 = sys.tau * sys.B;

sys.phi_aug = [sys.phi, sys.gamma_1;
            zeros(nu, nx), zeros(nu, nu)];
sys.gamma_aug = [sys.gamma_0; ones(nu, nu)];
sys.C_aug = [sys.C, zeros(1, nu)];
```