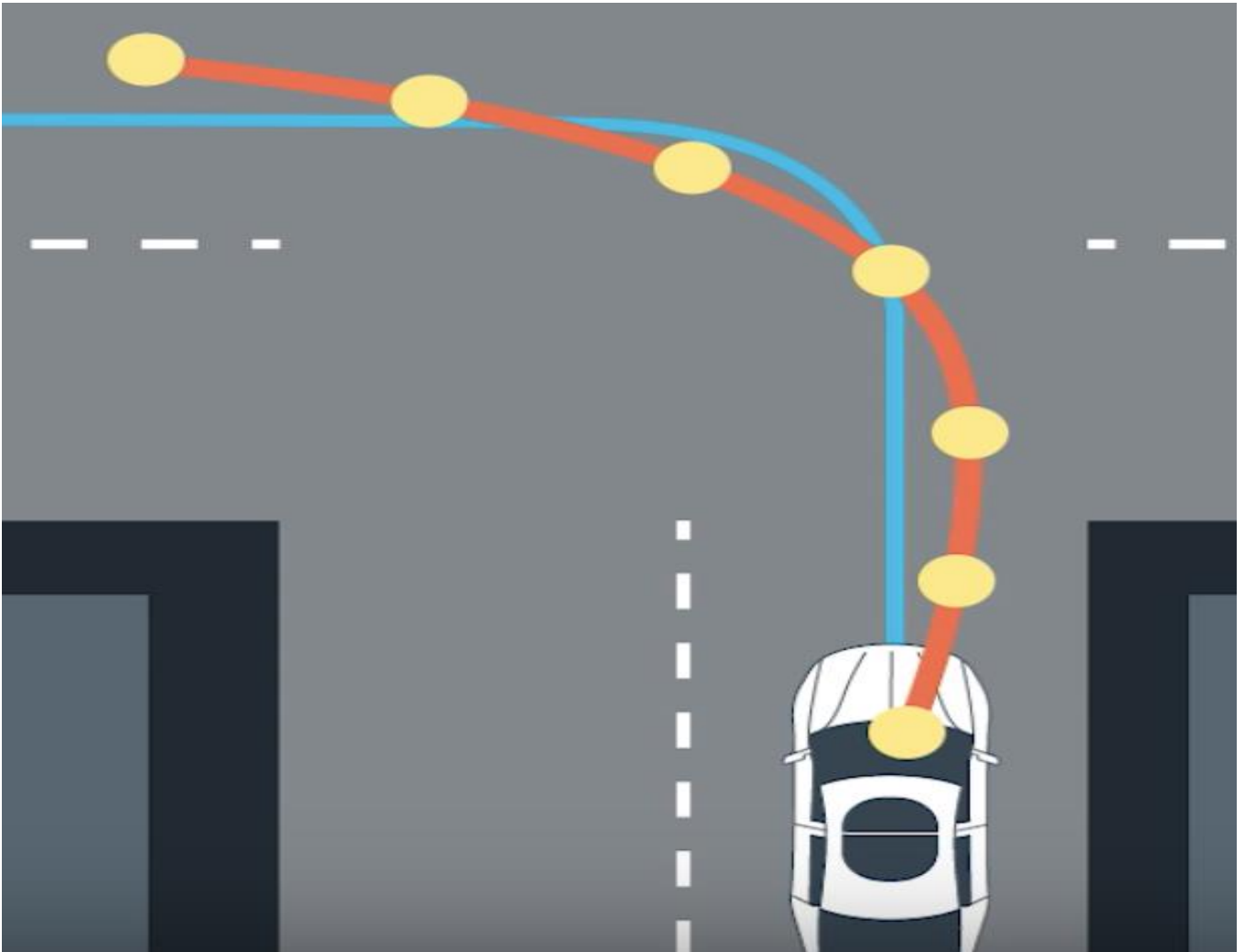

Control



Lab 3 : PID Controllers

Example 2: Cruise Control using PID Controller:

The parameters for the system are as follows:

- (m) vehicle mass = 1000 kg
- (b) damping coefficient = 50 N.s/m

1. Model the Car Cruise open loop system shown in Figure 9 and Equation (8) and get its open loop step response using attached MATLAB PID Simulator attached with the pdf. (if your MATLAB supports it), and by writing a MATLAB code.

- Plant transfer function $G(S) = \frac{V(s)}{U(s)} = \frac{1}{mS + b} = \frac{1}{1000S + 50}$

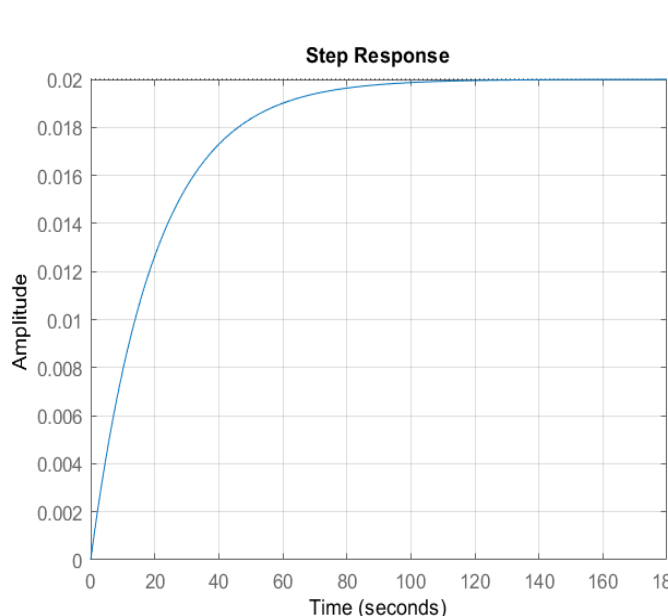
1) Open loop step response:

Code:

```
m = 1000; % Vehicle mass (Kg)
b = 50; % Damping coefficient (N.s/m)

s = tf('s');
plant_tf = 1/(m*s + b)
[y ~] = step(plant_tf);
stepplot(plant_tf);
grid on;
% rlocusplot(plant_tf)
```

```
sysprop = stepinfo(plant_tf, 'RiseTimeThreshold', [0.1 0.9]);
sysprop
steady_state_error = abs(1-y(end))
```

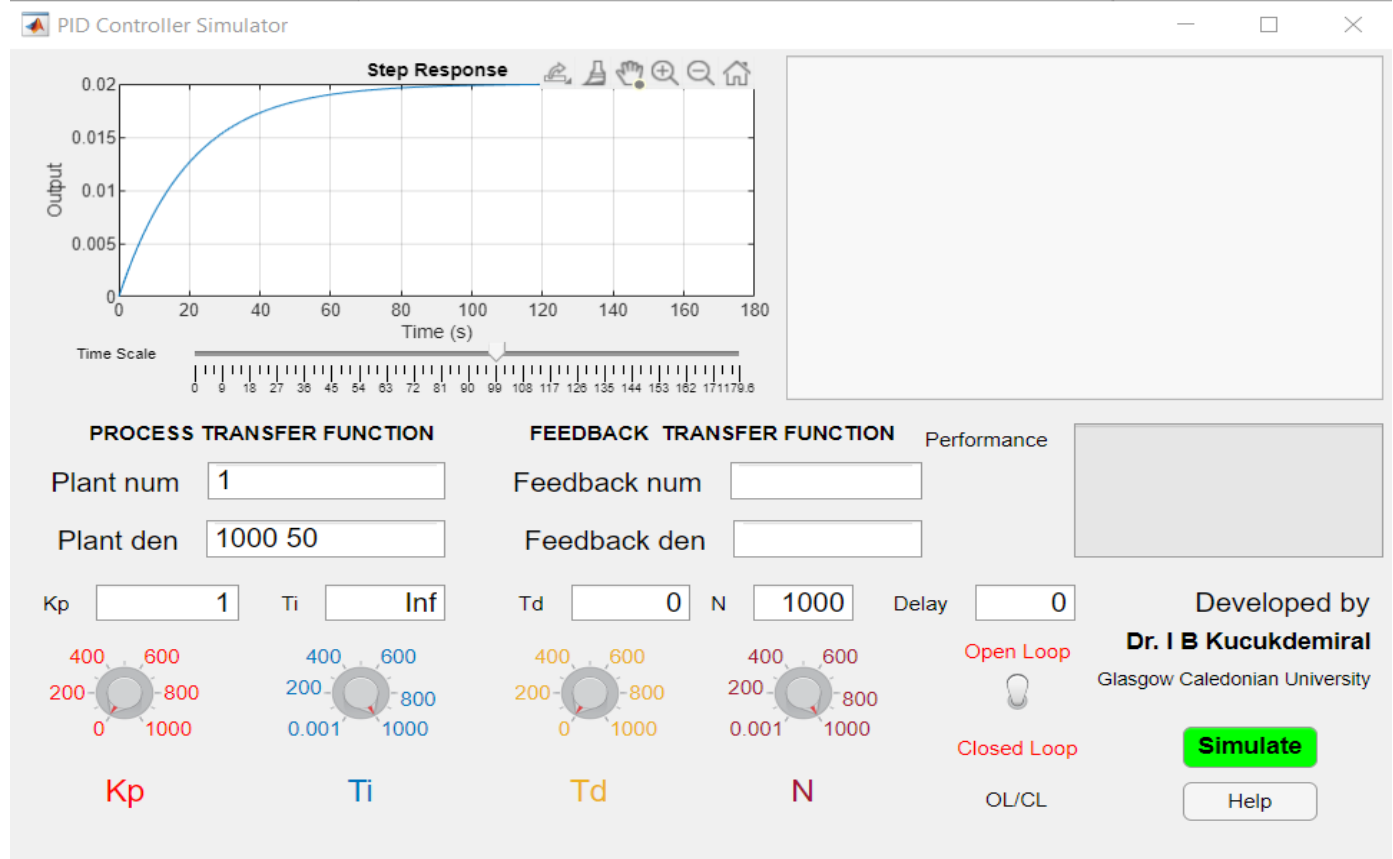


```
0 20 40 60 80 100 120 140 160 180
Time (seconds)
```

```
sysprop = struct with fields:
    RiseTime: 43.9401
    SettlingTime: 78.2415
    SettlingMin: 0.0181
    SettlingMax: 0.0200
    Overshoot: 0
    Undershoot: 0
    Peak: 0.0200
    PeakTime: 210.9168
```

```
steady_state_error = 0.9800
```

Using PID Simulator:



2. Adjust the PID parameters to get the following system specifications for a step response closed loop negative unity feedback system:

- Rise time < 5 sec
- Overshoot $< 10\%$
- Steady-state error $< 2\%$

I can't use the **Zeigler-Nichols** PID tuning method because there is only a pole at -0.05 so the root locus doesn't intersect the imaginary axes so I can't get ultimate gain (K_U) and ultimate period (T_U).

So I will use **Manual PID tuning** (trial and error):

I will use MATLAB live script and use the **numeric slider** for K_P & K_D & K_I

LABEL	VALUES	EXECUTION
Text to display when code is hidden Label: <input type="text" value="KP"/>	Min: <input type="text" value="0"/> Max: <input type="text" value="500"/> Step: <input type="text" value="0.5"/>	Run On: <input type="text" value="Value changing"/> Run: <input type="text" value="Current section"/>
Text to display when code is hidden Label: <input type="text" value="KI"/>	Min: <input type="text" value="0"/> Max: <input type="text" value="500"/> Step: <input type="text" value="0.5"/>	Run On: <input type="text" value="Value changing"/> Run: <input type="text" value="Current section"/>
Text to display when code is hidden Label: <input type="text" value="KD"/>	Min: <input type="text" value="0"/> Max: <input type="text" value="500"/> Step: <input type="text" value="0.5"/>	Run On: <input type="text" value="Value changing"/> Run: <input type="text" value="Current section"/>

1) I want to decrease the **rise time** from **43.9401 secs** in case of the open loop **to be less than 5 secs** so I need to add PID and increase **KP & KI**.

2) But increasing **KP & KI** increases the maximum overshoot so I need to increase **KD** which decrease the maximum overshoot.

3) Adding **KI** makes the steady state error to be almost zero because it increases the system type

I will choose:

KP = 340 & KI = 50 & KD = 100

Code:

```
KP = 340
KI = 50
KD = 100
```

```
control_tf = pid(KP,KI,KD);
```

```
closedLoop_tf = feedback(control_tf*plant_tf,1);
[y ~] = step(closedLoop_tf);
stepplot(closedLoop_tf);
grid on;
sysprop = stepinfo(closedLoop_tf, 'RiseTimeThreshold', [0.1 0.9]);
rise_time = sysprop.RiseTime
max_overshoot = sysprop.Overshoot
steady_state_err = abs(1-y(end)) *100
```

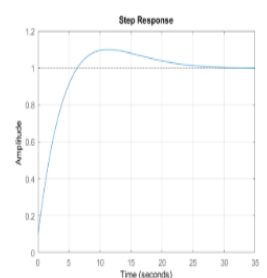
```
16 KP = 340
17 KI = 50
18 KD = 100
19
20 control_tf = pid(KP,KI,KD);
21
22 closedLoop_tf = feedback(control_tf*plant_tf,1);
23 [y ~] = step(closedLoop_tf);
24 stepplot(closedLoop_tf);
25 grid on;
26 sysprop = stepinfo(closedLoop_tf, 'RiseTimeThreshold', [0.1 0.9]);
27 rise_time = sysprop.RiseTime
28 max_overshoot = sysprop.Overshoot
29 steady_state_err = abs(1-y(end)) *100
30
```

steady_state_err = 0.0000

KP = 340

KI = 50

KD = 100

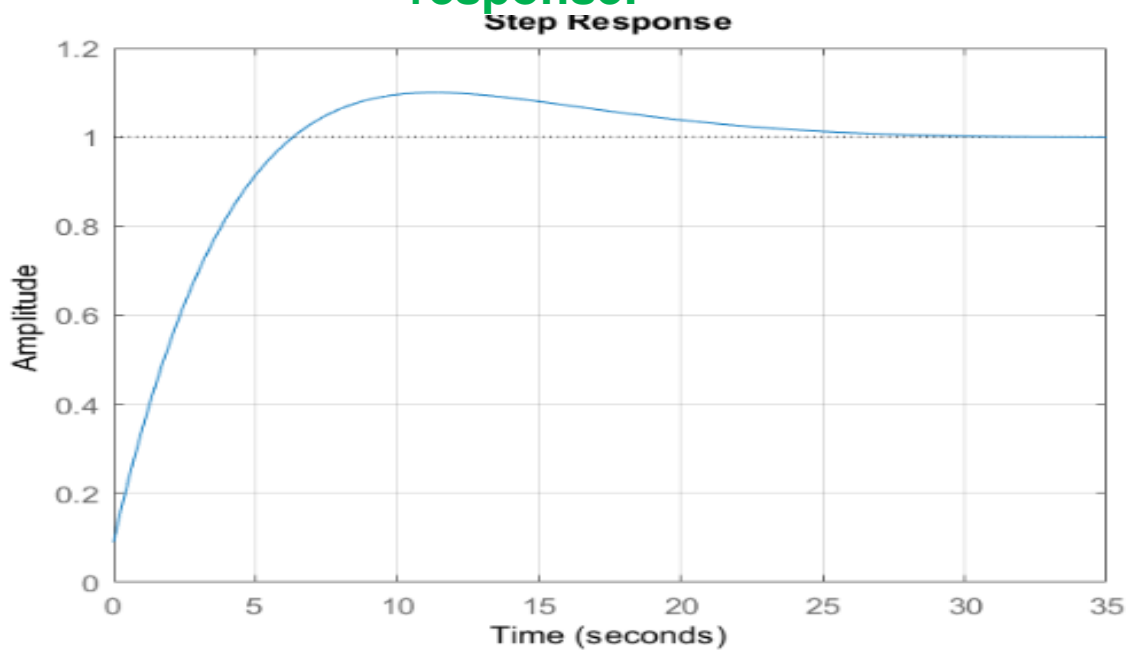


rise_time = 4.5992

max_overshoot = 9.9960

steady_state_err = 0.0057

Proportional Integral Derivative Controller step response:



```

rise_time = 4.5992
max_overshoot = 9.9960
steady_state_err = 0.0057

```

Using PID Simulator:

PID Controller Simulator

Step Response

Time (s)

with $K_p = 340$, $T_i = 6.8$, $T_d = 0.294$, $N = 1e+03$

Continuous-time PIDF controller in standard form

Process =

$$\frac{1}{1000s + 50}$$

Continuous-time transfer function.

H =

$$1$$

Static gain.

ClosedLoopTF =

$$\frac{3.403e05 s^2 + 1.156e06 s + 1.7e05}{1000 s^3 + 3.74e06 s^2 + 1.326e06 s + 1.7e05}$$

Continuous-time transfer function.

PROCESS TRANSFER FUNCTION

Plant num

Plant den

FEEDBACK TRANSFER FUNCTION

Feedback num

Feedback den

Performance

Gm = Inf

Pm = 80.8810

Wog =

Delay

Controller Parameters

K_p T_i

T_d N

Open Loop ☐

Closed Loop ☒

K_p T_i

T_d N

OL/CL ☐

Developed by

Dr. I B Kucukdemir

Glasgow Caledonian University

Simulate

Help