|  |  |
| --- | --- |
| Ex. No.: 5 | **System Identification through Process Reaction Curve** |
| Date: |

**Aim**

To determine the process steady state gain (K), dead time(τD) and effective process time constant τ from process reaction curve and also the determine controller gain (Kc), integral time constant (Ti) and derivative time constant (TD) for P, PI and PID controller.

**Introduction**

Ziegler and Nichols recognized that the open-loop step responses of a large number of processes exhibit a process reaction curve like that shown in Fig. 1 The S-shape of the curve can be approximated by the step response of

K = The process steady-state gain,

τD = The effective process dead time; and

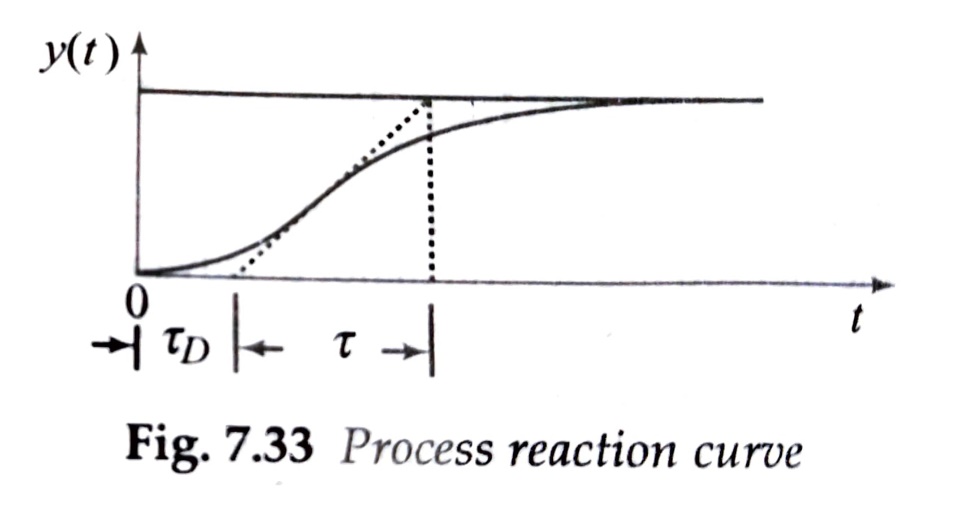
τ = The effective process time constant

Fig.1 Process Reaction Curve

The constants in Equation can be determined from the experimentally obtained step response y(t) to step input u(t) = Aµ(t). The steady-state value of the response yss = KA. This gives the parameter K. If line is KA/τ, and the intersection of the tangent line with the time axis identifies the dead-time τD.

The formulas in Table were developed empirically for the most common range of τD/τ, which is between 0.1 and 0.3.

**Table** QDR tuning formulas based on process reaction curve

|  |  |  |  |
| --- | --- | --- | --- |
| Controller | Gain | Integral time | Derivative time |
| P | Kc = τ /KτD | - | - |
| PI | Kc = 0.9τ /KτD | TI = 3.33τD | - |
| PID | Kc =1.5τ /KτD | TI = 2.5τD | TD = 0.4τD |

**Procedure**

Step 1: Identify the transfer function of system.

Step 2: Get the open loop step response of the system.

Step 3: Calculate the Gain, time constant and dead time from the process reaction curve.

Step4: Calculate the controller gain, integral time constant and derivative time constant

Step5: Display the result.

**Matlab Code**

**% Reaction curve based P I D Controller Tuning**

**clc;**

**clearvars;**

**% Transfer function of system**

**G1 = tf(1, [1 4 6 4 1]);**

**du=1;**

**t0=0;**

**u=1;**

**[y,t]=step(G1);**

**% Calculations of system constants**

**K=(y(end)-y(1))/du; % gain K**

**dy=diff(y);**

**dt=diff(t);**

**[mdy,I]=max(abs(dy)./dt);**

**t\_c=abs(y(end)-y(1))/mdy; % time constant**

**t\_d=t(I)-abs(y(I)-y(1))/mdy-t0;% dead time**

**fprintf("System Gain K: %0.2f\n",K);**

**fprintf("System time constant T: %0.2f\n",t\_c);**

**fprintf("System dead time: %0.2f\n",t\_d);**

**% Plot**

**figure;**

**plot(t,y,[t0+t\_d t0+t\_d+t\_c],[y(1) y(end)]);**

**title('System Response ')**

**grid on;**

**%% Controller Parameters**

**Controller\_P\_Kc = t\_c/(K\*t\_d);**

**Controller\_PI\_Kc= 0.9\*t\_c/(K\*t\_d);**

**Controller\_PID\_Kc= 1.5\*t\_c/(K\*t\_d);**

**ti\_PI = 3.33 \* t\_d;**

**ti\_PID = 2.5\*t\_d;**

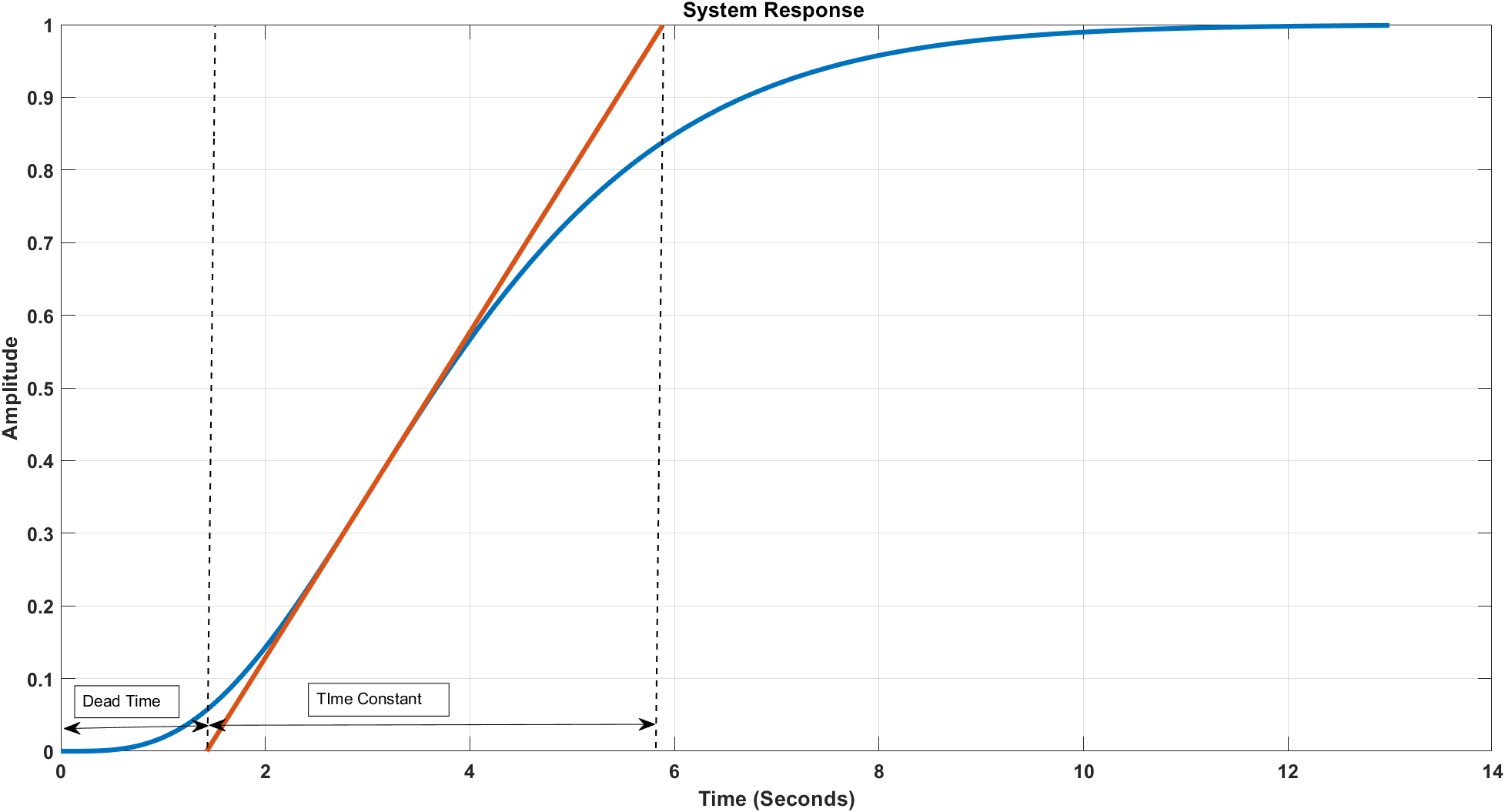
**td\_PID = 0.5\*t\_d;**

**fprintf("P Controller Gain (Kc) : %0.2f \n",Controller\_P\_Kc);**

**fprintf("PI Controller gain (Kc): %0.2f and Integral Time Constant (Ti): %0.2f \n",Controller\_PI\_Kc,ti\_PI);**

**fprintf("PID Controller gain (Kc): %0.2f and Integral Time Constant (Ti): %0.2f ..." + ...**

**"Derivative Time Constant (Td) %0.2f\n",Controller\_PID\_Kc,ti\_PID ,td\_PID);**

**Output Waveform**

**Fig. System Response and Constant Identification**

**System Constants and Controller Parameters**

System Gain K: 1.00

System time constant T: 4.46

System dead time: 1.43

P Controller Gain (Kc): 3.13

PI Controller gain (Kc): 2.82 and Integral Time Constant (Ti): 4.75

PID Controller gain (Kc): 4.70

Integral Time Constant (Ti): 3.56

Derivative Time Constant (Td): 0.71

**Result**

Thus, the system identification and controller parameters identification through process reaction curve was verified using Matlab code