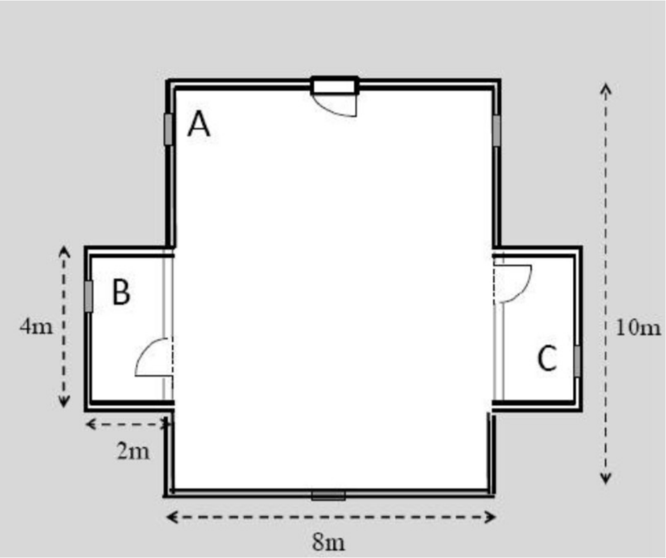
POLITECNICO DI MILANO

Temperature Control of a three-rooms building

Elena Bongiorni
Marina Colombo
Marco Crespi

System's description – I

The system taken into account is illustrated in the picture and described by the following model:



$$
\begin{aligned}
c\rho V_A \dot{T}_A &= s_r u(T_B - T_A) + s_r u(T_C - T_A) + s_A U(T_E - T_A) + q_A \\
c\rho V_{B,C} \dot{T}_B &= s_r u(T_A - T_B) + s_{B,C} U(T_E - T_B) + q_B \\
c\rho V_{B,C} \dot{T}_C &= s_r u(T_A - T_C) + s_{B,C} U(T_E - T_C) + q_C
\end{aligned}
$$

c: specific heat of the air; V_i, $i = A, B, C$: Volume of room i; s_r: wall surface between A and B (and C); s_A: wall surface between A and the environment; $s_{B,C}$: wall surface between B (and C) and the environment; u, U: transmittances; q_i, $i = A, B, C$: inputs (heating power).

System's description – II

At the equilibrium, the values of the variables are:

T_E=0°C, T_A=T_B=T_C=T_{eq}=20°C

q_A=q_{Aeq}=$s_A U T_{eq}$W

qB=qC=q_{Beq}=q_{Ceq}=$s_{B,C} U T_{eq}$W

Define $\delta T_{A/B/C}=T_{A/B/C}-T_{eq}$ and $\delta q_{A/B/C}=q_{A/B/C}-q_{A/B/Ceq}$

Around the given equilibrium, the model can be linearized as:

$$\begin{bmatrix} \delta\dot{T}_B \\ \delta\dot{T}_A \\ \delta\dot{T}_C \end{bmatrix} = \begin{bmatrix} -(\Gamma+\gamma_r) & \Gamma & 0 \\ \gamma & -(2\gamma+\gamma_A) & \gamma \\ 0 & \Gamma & -(\Gamma+\gamma_r) \end{bmatrix} \begin{bmatrix} \delta T_B \\ \delta T_A \\ \delta T_C \end{bmatrix} + \begin{bmatrix} \frac{1}{c\rho V_{B,C}} & 0 & 0 \\ 0 & \frac{1}{c\rho V_A} & 0 \\ 0 & 0 & \frac{1}{c\rho V_{B,C}} \end{bmatrix} \begin{bmatrix} \delta q_B \\ \delta q_A \\ \delta q_C \end{bmatrix}$$

Where:

$\gamma = \frac{s_r U}{c\rho V_A}$, $\Gamma = \frac{s_r U}{c\rho V_{B,C}}$, $\gamma_A = \frac{s_A U}{c\rho V_A}$, and $\gamma_r = \frac{s_{B,C} U}{c\rho V_{B,C}}$

System's description – III

Finally, defining:

$u_1 = \delta q_B$, $u_2 = \delta q_A$, $u_3 = \delta q_C$, $x = [\delta T_B \, \delta T_A \, \delta T_C]^T$

The decomposed model can be written as:

$$\dot{x} = Ax + B_1 u_1 + B_2 u_2 + B_3 u_3$$

$$y_1 = [1 \quad 0 \quad 0]x$$

$$y_2 = [0 \quad 1 \quad 0]x$$

$$y_3 = [0 \quad 0 \quad 1]x$$

Where:

$$A = \begin{bmatrix} -(\Gamma + \gamma_r) & \Gamma & 0 \\ \gamma & -(2\gamma + \gamma_A) & \gamma \\ 0 & \Gamma & -(\Gamma + \gamma_r) \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \; B_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \; B_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Matlab Code: data definition

```
u=2;
U=0.5;
VA=240;
VBC=24;
sr=12;
sA=84;
sBC=24;
c=1.225*1005;

gamma=sr*u/(c*VA);
GAMMA=sr*u/(c*VBC);
gammaA=sA*U/(c*VA);
gammar=sBC*U/(c*VBC);

Atot=[-(GAMMA+gammar) GAMMA              0
      gamma           -(2*gamma+gammaA) gamma
      0               GAMMA              -(GAMMA+gammar)];

Bdec{1}=[1/c/VBC;
         0;
         0];
Bdec{2}=[0;
         1/c/VA;
         0];
Bdec{3}=[0;
         0;
         1/c/VBC];
Cdec{1}=[1 0 0];
Cdec{2}=[0 1 0];
Cdec{3}=[0 0 1];

x_0=rand(3,1);
```

Question 1

Generate the system matrices (both continuous and discrete time). Perform the following analysis:
1a – Compute the eigenvalues and the spectral abscissa of the continuous time system. Is it open-loop asymptotically stable?
1b – Compute the eigenvalues and the spectral radius of the discrete time system. Is it open-loop asymptotically stable?

1 – Matlab code

```
%1 System matrices in continuous and discrete time
A=Atot;
B=[Bdec{1}, Bdec{2}, Bdec{3}];
C=[Cdec{1}; Cdec{2}; Cdec{3}];
D=0;
systemCT=ss(A, B, C, D);
systemDT=c2d(systemCT, 1);
[F,G,H,L,Ts]=ssdata(systemDT);


for i=1:3
    Gdec{i}=G(:,i);
    Hdec=Cdec;
end

%1a eigenvalues and spectral abscissa of CT system
eigA=eig(A);
rhoCT=max(real(eig(A)))
%1b eigenvalues and spectral abscissa of DT system
eig(F);
rhoDT=max(abs(eig(F)))
%the system is asymptoically stable since in CT all eigenvalues
%have strictly negative real part and in DT all eigenvalues are
%inside the unit circle.
```

Question 2

For different control structures (centralized, decentralized and different distributed schemes), perform the following actions:

2a – Compute the continuous time fixed modes (if necessary, adjust the rounding tolerance in the dedicated function).

2b – Compute the discrete time fixed modes (if necessary, adjust the rounding tolerance in the dedicated function).

2c– Compute, if possible, the corresponding stabilizing continuous time control gain using the LMIs.

2d– Compute, if possible, the corresponding stabilizing discrete time control gain using the LMIs.

2e – Analyse the properties of the so-obtained closed-loop systems (stability and eigenvalues), and compute the closed-loop system trajectories, generated both in continuous and discrete time, of the room temperature starting from a common random initial condition.

2 – Matlab code – I

```
%2 CENTRALIZED
N=3;
ContStrucC=ones(N,N);
rounding=4; %we made some tests and we noticed that with a rounding lower
            %than 4, the result can't be computed because of a too great
            %approximation, while with a greater rounding the result won't
            %change.
%2a continuous time fixed modes
CFMct=di_fixed_modes(A,Bdec,Cdec,N,ContStrucC,rounding);

%2b discrete time fixed modes
CFMdt=di_fixed_modes(F,Gdec,Hdec,N,ContStrucC,rounding);

%2c stabilizing CT control gain
[K_c,rho_c,feas_c]=LMI_CT_DeDicont(A,Bdec,Cdec,N,ContStrucC);

%2d stabilizing DT control gain
[KD_c,rhoD_c,feasD_c]=LMI_DT_DeDicont(F,Gdec,Hdec,N,ContStrucC);
```

METTETE I RISULTATI TIPO I CMFct ECCETERA
CHE A ME NON LI STAMPA MATLAB

2 – Matlab code – II

```
%2e eigenvalues, stability, CL trajectories in CT e DT
An_c=A+B*K_c;

k=0;

for t=0:0.1:30;
    k=k+1;
    yfree_c_ct(:,k)=C*expm(An_c*t)*x_0;
    u_c_ct(:,k)=K_c*yfree_c_ct(:,k);
end

figure(1)
hold on
grid on
plot(0:0.1:30,yfree_c_ct(1,:),0:0.1:30,yfree_c_ct(2,:),0:0.1:30,yfree_c_ct(3,:));
title ('Centralized Control in Continuous Time')
legend('\deltaTB','\deltaTA','\deltaTC')
xlabel('Interval time from 0 to 30[s]')
ylabel('Output free motions for Centralized Control')

figure(2)
hold on
grid on
plot(0:0.1:30,u_c_ct(1,:),0:0.1:30,u_c_ct(2,:),0:0.1:30,u_c_ct(3,:))
title ('Control action for Centralized Control, Continuous Time')
legend('\deltaqB','\deltaqA','\deltaqC')
xlabel('Interval time from 0 to 30[s]')
ylabel('Static Output-feedback Control Law u(t)')


eigen_CT_centralized=eig(An_c);
rho_CT_centralized=max(abs(real(eig(An_c))));
```

2 – Matlab code – III

Figure 1:

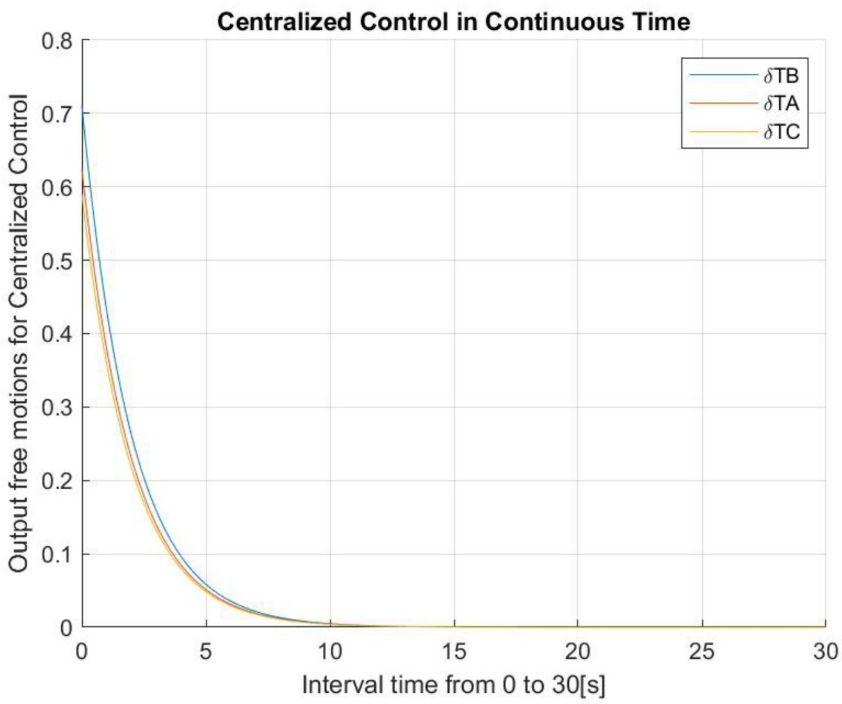
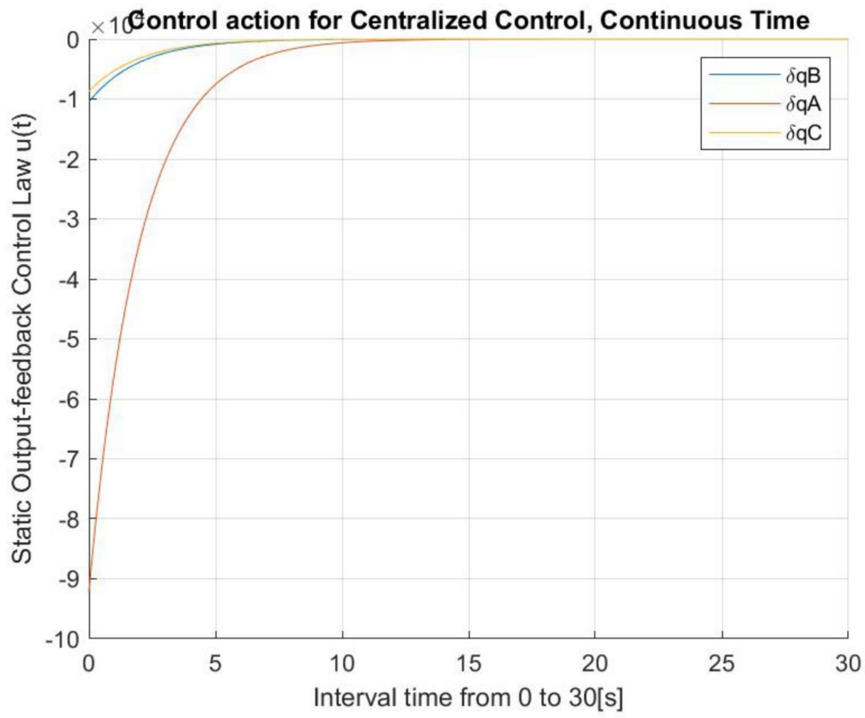


Figure 2:



2 – Matlab code – IV

```
h=1;
i=1;
k=0;
Tfinal=30;
steps=[0:Tfinal/h];
for k=steps
    yfree_c_dt(:,i)=(F+G*KD_c)^(k)*x_0;
    u_c_dt(:,i)=KD_c*yfree_c_dt(:,k+1);
    i=i+1;
end

figure(3)
hold on
grid on
plot(steps(1:30)*h,yfree_c_dt(1,1:30),'ko',steps(1:30)*h,yfree_c_dt(2,1:30),'k*',steps(1:30)*h,yfree_c_dt(3,1:30),'k.');
title ('Centralized Control in Discrete Time')
legend('\deltaTB','\deltaTA','\deltaTC')
xlabel('Time instants from 0 to 30 with sampling time h=1')
ylabel('Output free motions for Centralized Control')

figure(4)
hold on
grid on
plot(steps(1:30)*h,u_c_dt(1,1:30),'ko',steps(1:30)*h,u_c_dt(2,1:30),'k*',steps(1:30)*h,u_c_dt(3,1:30),'k.');
title ('Control action for Centralized Control, Discrete Time')
legend('\deltaqB','\deltaqA','\deltaqC')
xlabel('Time instants from 0 to 30 with sampling time h=1')
ylabel('Static Output-feedback Control Law u(kh)')

eigen_DT_centralized=eig(F+G*KD_c);
rho_DT_centralized=max(abs(real(eigen_DT_centralized)));
```

2 – Matlab code – V

Figure 3:

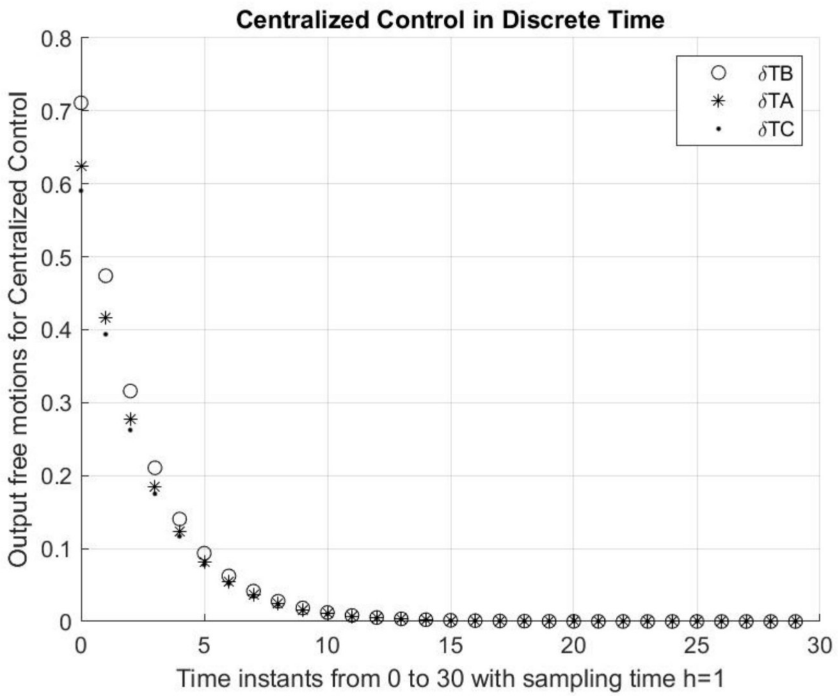
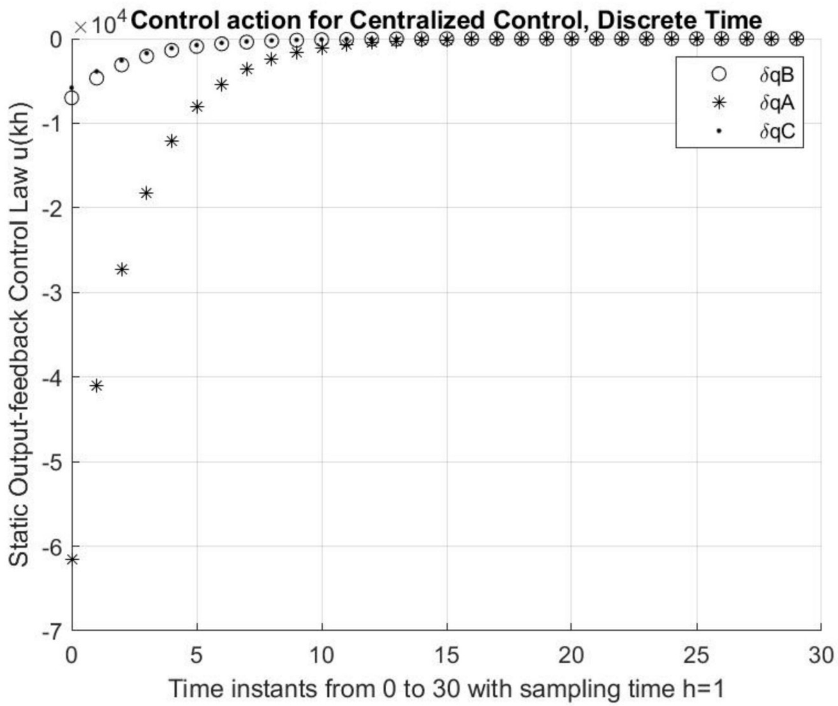


Figure 4:



2 – Matlab code – VI

```
%2 DECENTRALIZED
N=3;
ContStrucD=diag(ones(N,1));;
rounding=4;
%2a continuous time fixed modes
DFMct=di_fixed_modes(A,Bdec,Cdec,N,ContStrucC,rounding)

%2b discrete time fixed modes
DFMdt=di_fixed_modes(F,Gdec,Hdec,N,ContStrucC,rounding);

%2c stabilizing CT control gain
[K_d,rho_d,feas_d]=LMI_CT_DeDicont(A,Bdec,Cdec,N,ContStrucC)

%2d stabilizing DT control gain
[KD_d,rhoD_d,feasD_d]=LMI_DT_DeDicont(F,Gdec,Hdec,N,ContStrucC);
```

2 – Matlab code – VII

```
%2e eigenvalues, stability, CL trajectories in CT e DT
An_d=A+B*K_d;

k=0;

for t=0:0.1:30;
    k=k+1;
    yfree_de_ct(:,k)=C*expm(An_d*t)*x_0;
    u_de_ct(:,k)=K_d*yfree_de_ct(:,k);
end

figure(5)
hold on
grid on
plot(0:0.1:30,yfree_de_ct(1,:),0:0.1:30,yfree_de_ct(2,:),0:0.1:30,yfree_de_ct(3,:));
title ('Decentralized Control in Continuous Time')
legend('\deltaTB','\deltaTA','\deltaTC')
xlabel('Interval time from 0 to 30[s]')
ylabel('Output free motions for Decentralized Control')

figure(6)
hold on
grid on
plot(0:0.1:30,u_de_ct(1,:),0:0.1:30,u_de_ct(2,:),0:0.1:30,u_de_ct(3,:))
title ('Control action for Decentralized Control, Continuous Time')
legend('\deltaqB','\deltaqA','\deltaqC')
xlabel('Interval time from 0 to 30[s]')
ylabel('Static Output-feedback Control Law u(t)')


eigen_CT_decentralized=eig(An_d);
rhoCT_decentralized=max(abs(real(eig(An_d))));
```

2 – Matlab code – VIII

Figure 5:

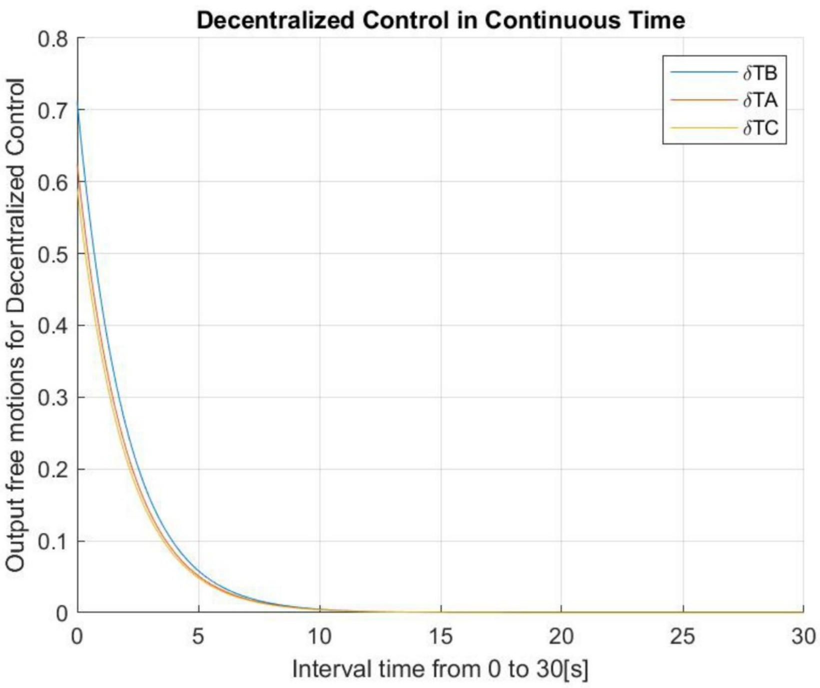
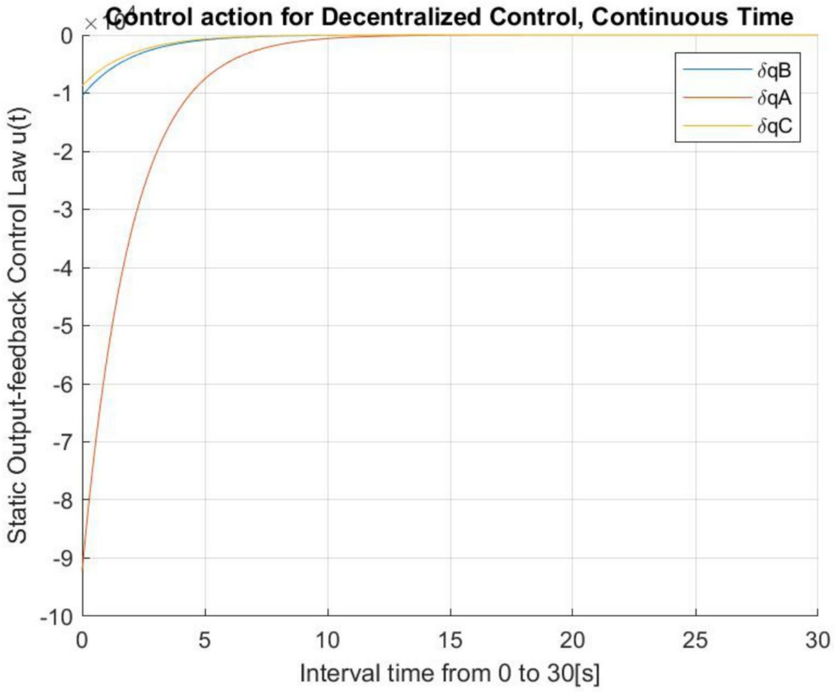


Figure 6:



2 – Matlab code - IX

```
i=1;
k=0;
Tfinal=30;
steps=[0:Tfinal/h];
for k=steps
    yfree_de_dt(:,i)=(F+G*KD_d)^(k)*x_0;
    u_de_dt(:,i)=KD_d*yfree_de_dt(:,k+1);
    i=i+1;
end

figure(7)
hold on
grid on
plot(steps(1:30)*h,yfree_de_dt(1,1:30),'ko',steps(1:30)*h,yfree_de_dt(2,1:30),'k*',steps(1:30)*h,yfree_de_dt(3,1:30),'k.');
title ('Decentralized Control in Discrete Time')
legend('\deltaTB','\deltaTA','\deltaTC')
xlabel('Time instants from 0 to 30 with sampling time h=1')
ylabel('Output free motions for Decentralized Control')

figure(8)
hold on
grid on
plot(steps(1:30)*h,u_de_dt(1,1:30),'ko',steps(1:30)*h,u_de_dt(2,1:30),'k*',steps(1:30)*h,u_de_dt(3,1:30),'k.');
title ('Control action in Decentralized Control, Discrete Time')
legend('\deltaqB','\deltaqA','\deltaqC')
xlabel('Time instants from 0 to 30 with sampling time h=1')
ylabel('Static Output-feedback Control Law u(kh)')

eigen_DT_decentralized=eig(F+G*KD_d);
rho_DT_decentralized=max(abs(real(eigen_DT_decentralized)));
```

2 – Matlab code - X

Figure 7:

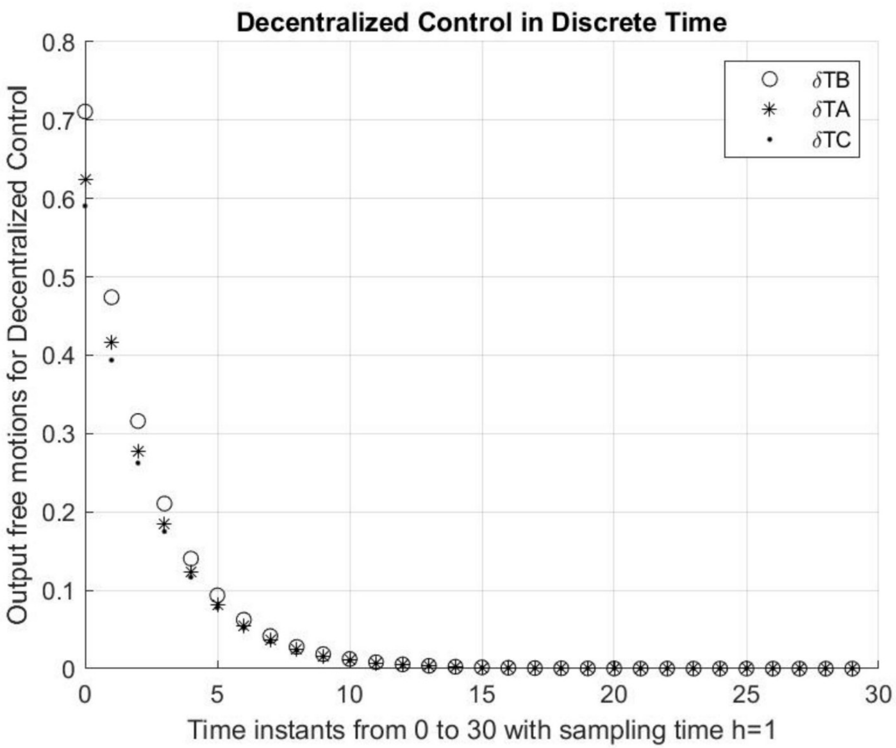
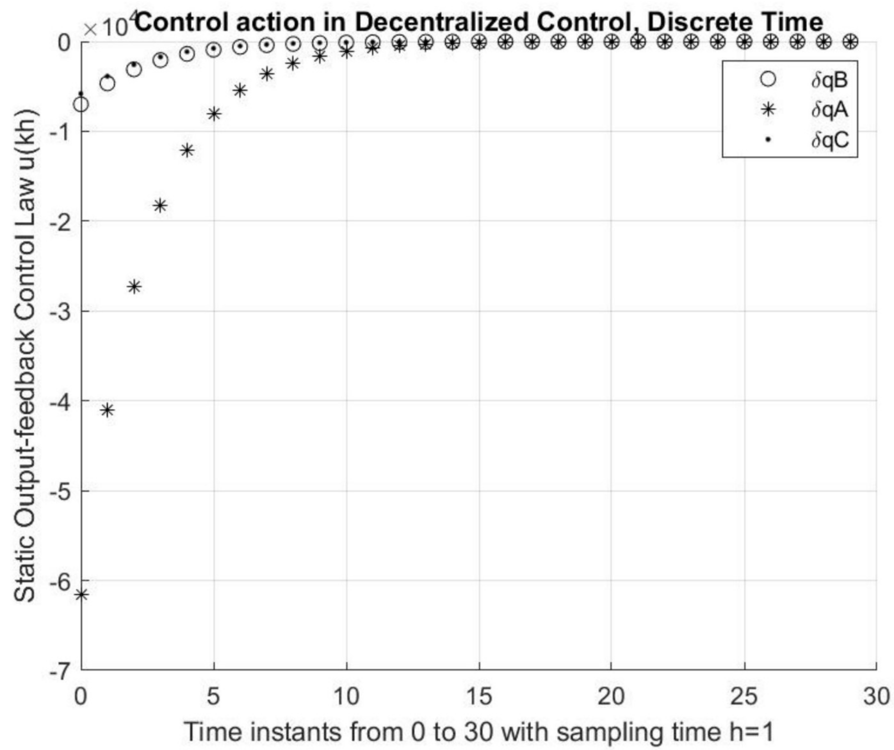


Figure 8:



2 – Matlab code - XI

```
%2 DISTRIBUTED
N=3;
ContStrucD=diag(ones(N,1));;
rounding=4;
%2a continuous time fixed modes
DiFMct=di_fixed_modes(A,Bdec,Cdec,N,ContStrucC,rounding)

%2b discrete time fixed modes
DiFMdt=di_fixed_modes(F,Gdec,Hdec,N,ContStrucC,rounding);

%2c stabilizing CT control gain
[K_di,rho_di,feas_di]=LMI_CT_DeDicont(A,Bdec,Cdec,N,ContStrucC)

%2d stabilizing DT control gain
[KD_di,rhoD_di,feasD_di]=LMI_DT_DeDicont(F,Gdec,Hdec,N,ContStrucC);
```

2 – Matlab code - XII

```
%2e eigenvalues, stability, CL trajectories in CT e DT
An_di=A+B*K_di;

k=0;

for t=0:0.1:30;
    k=k+1;
    yfree_di_ct(:,k)=C*expm(An_di*t)*x_0;
    u_di_ct(:,k)=K_di*yfree_di_ct(:,k);
end

figure(9)
hold on
grid on
plot(0:0.1:30,yfree_di_ct(1,:),0:0.1:30,yfree_di_ct(2,:),0:0.1:30,yfree_di_ct(3,:));
title ('Distributed Control in Continuous Time')
legend('\deltaTB','\deltaTA','\deltaTC')
xlabel('Interval time from 0 to 30[s]')
ylabel('Output free motions for Distributed Control')


figure(10)
hold on
grid on
plot(0:0.1:30,u_di_ct(1,:),0:0.1:30,u_di_ct(2,:),0:0.1:30,u_di_ct(3,:))
title ('Control action for Distributed Control, Continuous Time')
legend('\deltaqB','\deltaqA','\deltaqC')
xlabel('Interval time from 0 to 30[s]')
ylabel('Static Output-feedback Control Law u(t)')

eigen_CT_distributed=eig(An_di);
rhoCT_distributed=max(abs(real(eig(An_di))));
```

2 – Matlab code - XIII

Figure 9:

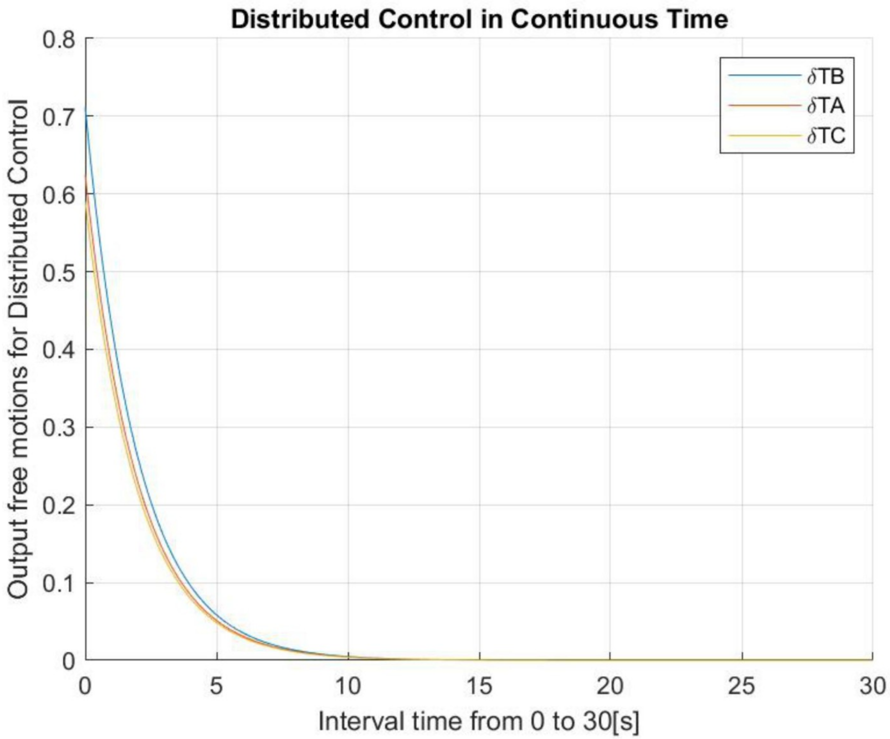
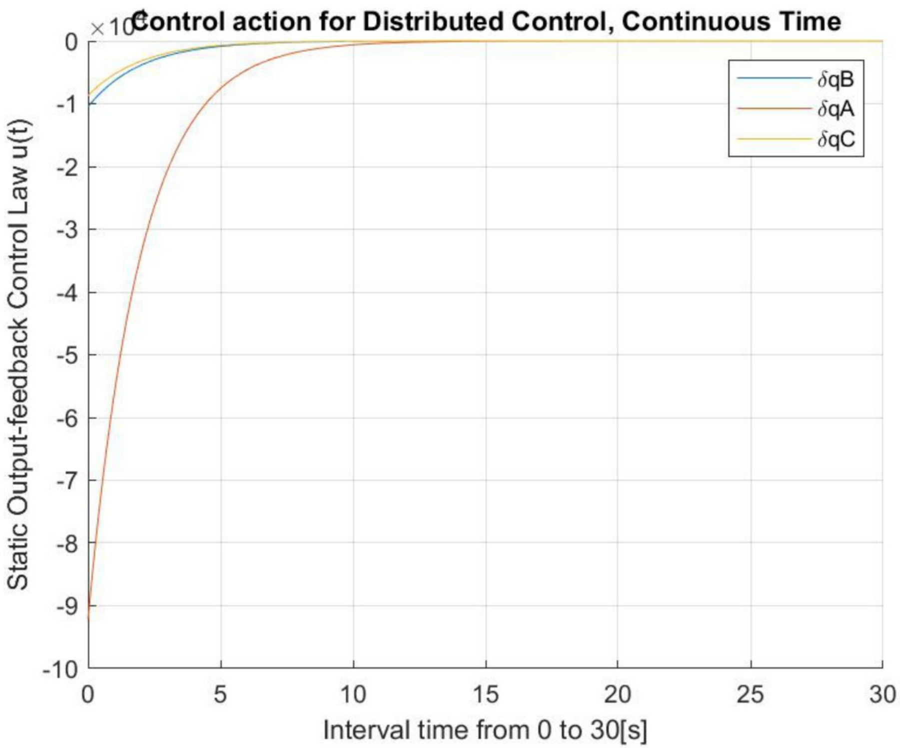


Figure 10:



2 – Matlab code - XII

```
i=0;
k=0;
Tfinal=30;
steps=[0:Tfinal/h];
for k=steps
    i=i+1;
    yfree_di_dt(:,k+1)=(F+G*KD_di)^(k)*x_0;
    u_di_dt(:,k+1)=KD_di*yfree_di_dt(:,k+1);
end

figure(11)
hold on
grid on
plot(steps(1:30)*h,yfree_di_dt(1,1:30),'ko',steps(1:30)*h,yfree_di_dt(2,1:30),'k*',steps(1:30)*h,yfree_di_dt(3,1:30),'k.');
title ('Distributed Control in Discrete Time')
legend('\deltaTB','\deltaTA','\deltaTC')
xlabel('Time instants from 0 to 30 with sampling time h=1')
ylabel('Output free motions for Distributed Control')

figure(12)
hold on
grid on
plot(steps(1:30)*h,u_di_dt(1,1:30),'ko',steps(1:30)*h,u_di_dt(2,1:30),'k*',steps(1:30)*h,u_di_dt(3,1:30),'k.');
title ('Control action in Distributed Control, Discrete Time')
legend('\deltaqB','\deltaqA','\deltaqC')
xlabel('Time instants from 0 to 30 with sampling time h=1')
ylabel('Static Output-feedback Control Law u(kh)')

eigen_DT_distributed=eig(F+G*KD_di);
rho_DT_distributed=max(abs(real(eigen_DT_distributed)));
```

2 – Matlab code - XIII

Figure 11:

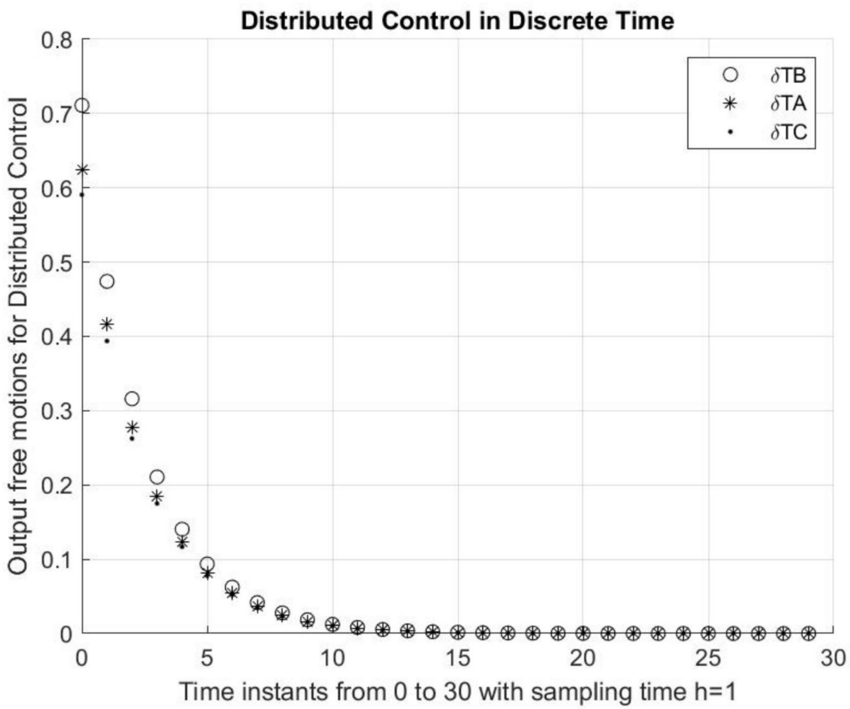


Figure 12: