Spis treści

1. Projekt

1.1. Poprawność podanego punktu pracy

Zasymulowano odpowiedź procesu w punkcie pracy dla sterowania upp1=upp2=upp3=upp4=0
Wyjscia obiektu wynoszą ypp1=ypp2=ypp3 =0.
Podany punkt pracy jest poprawny

1.2. Odpowiedzi skokowe 12 torów procesów

Wyznaczono odpowiedzi skokowe dla 12 torów procesu to znaczy zestaw liczb smn dla m równe 1, 2, 3, gdzie m oznacza numer wyjścia y i n równe 1, 2, 3, 4, gdzie n oznacz numer sterowania u przy pojedynczych skokach jednostkowych odpowiednich sygnałów sterujących od chwili k równe 0 włącznie sygnał wymusznia ma wartośc 1, w przeszłości jest zerowy.

1.3. Program do symulacji algorytmu cyfrowego PID i DMC w najprostszej wersji analitycznej

Zaimplementowano cyfrowy algorytm PID oraz algorytm DMC (w najprostszej wersji analitycznej)

Info o PID i DMC z poprzednich

1.3.1. Program do symulacji algorytmu cyfrowego PID

Inicjalizacja

Listing 1.1. "Inicjalizacja"

```
%zadanie 3 i 4 - Skrypt realizujacy algorytm cyfrowego
%                    wielowymiarowego regulatora PID
%inicjalizacja
clear all

E = 0;%wspolczynnik jakosci regualcji
ny = 3;
nu = 4;

Tp = 0.5;%czas probkowania

ster = 4;%odrzucany sygnal sterujacy
```

Nastawy regulatorów eksperymentalnie

Listing 1.2. "Nastawy regulatorów"

```
%nastawy regulatorow

% %eksperymentalnie
% if ster == 1
%     Kr1 = 1.5; Ti1 = 2; Td1 = 0.01;%u2 dla y3
%     Kr2 = 5.5; Ti2 = 0.4; Td2 = 0.2;%u3 dla y2
%     Kr3 = 2; Ti3 = 9; Td3 = 1;%u4 dla y1
% elseif ster == 2
%     Kr1 = 0.7; Ti1 = 0.2; Td1 = 0.3;%u1 dla y3
%     Kr2 = 3.5; Ti2 = 0.2; Td2 = 0.2;%u3 dla y2
%     Kr3 = 3; Ti3 = 7.5; Td3 = 0.8;%u4 dla y1
% elseif ster == 3
%     Kr1 = 0.7; Ti1 = 0.2; Td1 = 0.3;%u1 dla y3
%     Kr2 = 0.7; Ti2 = 1.8; Td2 = 0.6;%u2 dla y2
%     Kr3 = 1.4; Ti3 = 5.5; Td3 = 0.6;%u4 dla y1
% elseif ster == 4
%     Kr1 = 0.7; Ti1 = 0.2; Td1 = 0.3;%u1 dla y3
%     Kr2 = 0.6; Ti2 = 0.3; Td2 = 0.05;%u2 dla y2
%     Kr3 = 0.8; Ti3 = 0.4; Td3 = 0.4;%u3 dla y1
% end
```

Optymalizacja

Listing 1.3. "Optymalizacja"

```
%optymalizacja
if ster == 1
    Kr1 = 2.4380; Ti1 = 3.2542; Td1 = 0;%u2 dla y3
    Kr2 = 8.8647; Ti2 = 0.2623; Td2 = 0;%u3 dla y2
    Kr3 = 3.1042; Ti3 = 16.8144; Td3 = 1.0262;%u4 dla y1
elseif ster == 2
    Kr1 = 2.2901; Ti1 = 0.5102; Td1 = 0.0187;%u1 dla y3
    Kr2 = 0.0219; Ti2 = 0.0006; Td2 = 39.5656;%u3 dla y2
    Kr3 = 4.8545; Ti3 = 17.6086; Td3 = 0.4969;%u4 dla y1
elseif ster == 3
    Kr1 = 2.4231; Ti1 = 0.6438; Td1 = 0;%u1 dla y3
    Kr2 = 1.3759; Ti2 = 1.1730; Td2 = 0;%u2 dla y2
    Kr3 = 6.4637; Ti3 = 13.3023; Td3 = 0.0984;%u4 dla y1
elseif ster == 4
    Kr1 = 2.5885; Ti1 = 0.5918; Td1 = 0;%u1 dla y3
    Kr2 = 1.4396; Ti2 = 0.6631; Td2 = 0;%u2 dla y2
    Kr3 = 13.4882; Ti3 = 4.4987; Td3 = 0.0290;%u3 dla y1
end
```

Parametry symulacji

Listing 1.4. "Parametry symulacji"

```
%parametry symulacji
kk = 1600;
start = 10;
e = zeros(1,kk);
u1 = zeros(1,kk);
u2 = zeros(1,kk);
u3 = zeros(1,kk);
u4 = zeros(1,kk);
y1 = zeros(1,kk);
y2 = zeros(1,kk);
y3 = zeros(1,kk);

Ey = zeros(ny,1);

y1_zad = zeros(1,kk);
y1_zad(start:400) = 1;
y1_zad(400:800) = 1.5;
y1_zad(800:1200) = 0.6;
y1_zad(1200:kk) = 2.5;

y2_zad = zeros(1,kk);
y2_zad(start:400) = 2;
y2_zad(400:800) = 1.2;
y2_zad(800:1200) = 0;
y2_zad(1200:kk) = 1.5;

y3_zad = zeros(1,kk);
y3_zad(start:400) = 1.5;
y3_zad(400:800) = 0.8;
y3_zad(800:1200) = 2;
y3_zad(1200:kk) = 0.2;
```

Główna pętla symulacyjna

Listing 1.5. "Główna pętla symulacyjna"

```
%glowna petla symulacyjna
for k = start:kk
    %symulacja obiektu
   [y1(k),y2(k),y3(k)] = symulacja_obiektu7(...
       u1(k-1),u1(k-2),u1(k-3),u1(k-4),...
       u2(k-1),u2(k-2),u2(k-3),u2(k-4),...
       u3(k-1),u3(k-2),u3(k-3),u3(k-4),...
       u4(k-1),u4(k-2),u4(k-3),u4(k-4),...
       y1(k-1),y1(k-2),y1(k-3),y1(k-4),...
       y2(k-1),y2(k-2),y2(k-3),y2(k-4),...
       y3(k-1),y3(k-2),y3(k-3),y3(k-4));

   %uchyb regulacji
    e(1,k) = y1_zad(k) - y1(k);
    e(2,k) = y2_zad(k) - y2(k);
    e(3,k) = y3_zad(k) - y3(k);

    if ster == 1
        u2(k) = r21*e(3,k-2)+r11*e(3,k-1)+...
                r01*e(3,k)+u2(k-1);%y3 od u2
        u3(k) = r22*e(2,k-2)+r12*e(2,k-1)+...
                r02*e(2,k)+u3(k-1);%y2 od u3
        u4(k) = r23*e(1,k-2)+r13*e(1,k-1)+...
                r03*e(1,k)+u4(k-1);%y1 od u4
    elseif ster == 2
        u1(k) = r21*e(3,k-2)+r11*e(3,k-1)+...
                r01*e(3,k)+u1(k-1);%y3 od u1
        u3(k) = r22*e(2,k-2)+r12*e(2,k-1)+...
                r02*e(2,k)+u2(k-1);%y2 od u3
        u4(k) = r23*e(1,k-2)+r13*e(1,k-1)+...
                r03*e(1,k)+u4(k-1);%y1 od u4
    elseif ster == 3
        u1(k) = r21*e(3,k-2)+r11*e(3,k-1)+...
                r01*e(3,k)+u1(k-1);%y3 od u1
        u2(k) = r22*e(2,k-2)+r12*e(2,k-1)+...
                r02*e(2,k)+u2(k-1);%y2 od u2
        u4(k) = r23*e(1,k-2)+r13*e(1,k-1)+...
                r03*e(1,k)+u4(k-1);%y1 od u4
    elseif ster == 4
        u1(k) = r21*e(3,k-2)+r11*e(3,k-1)+...
                r01*e(3,k)+u1(k-1);%y3 dla u1
        u2(k) = r22*e(2,k-2)+r12*e(2,k-1)+...
                r02*e(2,k)+u2(k-1);%y2 dla u2
        u3(k) = r23*e(1,k-2)+r13*e(1,k-1)+...
                r03*e(1,k)+u3(k-1);%y1 dla u3
    end
```

1.3.2. Program do symulacji algorytmu DMC w najprostszej wersji analitycznej

Nastawy regulatora DMC dobrane eksperymentalnie i optymalizacyjnie

Listing 1.6. "Nastawy regulatora DMC dobrane eksperymentalnie i optymalizacyjnie""

```
%zadanie 3 i 4 - Skrypt relizujacy algorytm DMC regulatora
%                uproszczonego wielowymiarowego
clear all

%nastawy regulatora DMC
D = 350;%horyzont dynamiki

N = 200;%horyzont predykcji
Nu = 10;%horyzont sterowania


%dobrane eksperymentalnie

% lambda1 = 0.15;
% lambda2 = 0.2;
% lambda3 = 0.7;
% lambda4 = 0.1;
% psi1 = 1;
% psi2 = 1;
% psi3 = 1;

%optymalizacja

lambda1 = 0.9004;
lambda2 = -5.5093;
lambda3 = 12.1642;
lambda4 = -0.0569;
psi1 = -0.5147;
psi2 = -9.3173;
psi3 = 3.2215;
```

Wyznaczanie macierzy S, Mp i M regulatora DMC

Listing 1.7. "Wyznaczanie macierzy S Mp i M regulatora DMC"

```
%warunki poczatkowe
nu = 4;
ny = 3;
load('s.mat')
%Macierz odopowiedzi skokowych
for i = 1:D
    S(i)={[s11(i) s12(i) s13(i) s14(i);...
           s21(i) s22(i) s23(i) s24(i);...
           s31(i) s32(i) s33(i) s34(i)]};
end

% Macierz predykcji
for i = 1:N
   for j = 1:D-1
      if i+j <= D
         Mp{i,j} = S{i+j}-S{j};
      else
         Mp{i,j} = S{D}-S{j};
      end
   end
end

% Macierz M
for i=1:Nu
    M(i:N,i)=S(1:N-i+1);
    M(1:i-1,i)={[0 0 0 0; 0 0 0 0; 0 0 0 0]};
end
```

Wyznaczanie macierzy Lambda, Psi i K regulatora DMC

Listing 1.8. "Wyznaczanie macierzy Lambda Psi i K regulatora DMC"

```
%Macierz lambda
for i=1:Nu
    for j=1:Nu
        if i==j
            Lambda{i,j}=[lambda1 0 0 0; 0 lambda2 0 0;...
                         0 0 lambda3 0; 0 0 0 lambda4];
        else
            Lambda{i,j}=[0 0 0 0; 0 0 0 0;...
                         0 0 0 0; 0 0 0 0;];
        end
    end
end

%Macierz Psi
for i=1:N
    for j=1:N
        if i==j
            Psi{i,j}=[psi1 0 0; 0 psi2 0; 0 0 psi3];
        else
            Psi{i,j}=[0 0 0; 0 0 0; 0 0 0];
        end
    end
end

%Macierz K
for i = 1:Nu
    size(i) = nu;
end
for i=1:N
    size2(i) = ny;
end

M_temp = cell2mat(M);
M_temp_tr = M_temp';
Psi_temp = cell2mat(Psi);
M_temp_tr = M_temp_tr * Psi_temp;
M_temp = mat2cell(M_temp_tr*M_temp,size,size);
for i=1:Nu
    for j=1:Nu
        L{i,j} = M_temp{i,j}+Lambda{i,j}; %Dodanie Lambdy
    end
end
L_temp = cell2mat(L);
L_temp_rev = mat2cell(L_temp^(-1),size,size);
L_temp_rev = cell2mat(L_temp_rev);

K = mat2cell(L_temp_rev * M_temp_tr,size,size2);
```

Wyznaczanie macierzy Ku oszczędnego regulatora DMC

Listing 1.9. "Wyznaczanie macierzy Ku oszczędnego regulatora DMC"

```
%oszczedny DMC
Mp_tmp = cell2mat(Mp);
K1 = cell2mat(K(1,:));
Ku = K1*Mp_tmp;
for i = 1:nu
    for j = 1:ny
        Ke(i,j) = sum(K1(i,j:3:N*ny));
    end
end
```

Inicjalizacja macierzy do przechowywania danych

Listing 1.10. "Inicjalizacja macierzy do przechowywania danych"

```
%Macierze do przechowywania danych
u_d = zeros(nu,1);
for i = 1:D-1
    u_delta(i,1) = {u_d};
end

y_z = zeros(ny,1);
for i = 1:N
    y_zad_mod(i,1) = {y_z};
end

Y_dmc = zeros(ny,1);
for i=1:N
    y_mod(i,1)={Y_dmc};
end

for i=1:N
    Y0(i,1)={[0;0]};
end

du = zeros(nu,1);
for i=1:Nu
    dU_mod(i,1)={du};
end
```

Parametry symulacji

Listing 1.11. "Parametry symulacji"

```
%parametry symulacji
kk = 1600;
start = 10;
u1 = zeros(1,kk);
u2 = zeros(1,kk);
u3 = zeros(1,kk);
u4 = zeros(1,kk);
y1 = zeros(1,kk);
y2 = zeros(1,kk);
y3 = zeros(1,kk);

Ey = zeros(ny,1);

y1_zad = zeros(1,kk);
y1_zad(start:400) = 1;
y1_zad(400:800) = 1.5;
y1_zad(800:1200) = 0.6;
y1_zad(1200:kk) = 2.5;

y2_zad = zeros(1,kk);
y2_zad(start:400) = 2;
y2_zad(400:800) = 1.2;
y2_zad(800:1200) = 0;
y2_zad(1200:kk) = 1.5;

y3_zad = zeros(1,kk);
y3_zad(start:400) = 1.5;
y3_zad(400:800) = 0.8;
y3_zad(800:1200) = 2;
y3_zad(1200:kk) = 0.2;
```

Główna pętla symulacyjna

Listing 1.12. "Główna pętla symulacyjna"

```
% Symulacja
for k = start:kk
    %symulacja obiektu
    [y1(k),y2(k),y3(k)] = symulacja_obiektu7(...
       u1(k-1),u1(k-2),u1(k-3),u1(k-4),...
       u2(k-1),u2(k-2),u2(k-3),u2(k-4),...
       u3(k-1),u3(k-2),u3(k-3),u3(k-4),...
       u4(k-1),u4(k-2),u4(k-3),u4(k-4),...
       y1(k-1),y1(k-2),y1(k-3),y1(k-4),...
       y2(k-1),y2(k-2),y2(k-3),y2(k-4),...
       y3(k-1),y3(k-2),y3(k-3),y3(k-4));

    %Regulator
    delta_y(1) = y1_zad(k) - y1(k);
    delta_y(2) = y2_zad(k) - y2(k);
    delta_y(3) = y3_zad(k) - y3(k);

    K1_tmp = Ke*delta_y';

    %obliczanie dU
    u_delta_tmp = cell2mat(u_delta);
    Ku_tmp = Ku*u_delta_tmp;
    du = K1_tmp - Ku_tmp;

    for n = D-1:-1:2
      u_delta(n) = u_delta(n-1);
    end

    u1(k) = u1(k-1) + du(1);
    u2(k) = u2(k-1) + du(2);
    u3(k) = u3(k-1) + du(3);
    u4(k) = u4(k-1) + du(4);
    u_delta(1,1) = {du};

   %bledy
   Ey(1) = Ey(1) + (y1_zad(k) - y1(k))^2;
   Ey(2) = Ey(2) + (y2_zad(k) - y2(k))^2;
   Ey(3) = Ey(3) + (y3_zad(k) - y3(k))^2;

end
```

1.4. Eksperymentalne dobranie konfiguracji i parametrów regulatorów PID i DMC

Dla zaproponowanej trajektorii zmian sygnałów zadanych (kilka skoków o róznej amplitudzie) dobrac nastawy regulatora PID i parametry algorytmu DMC metoda eksperymentalna. Jakosc regulacji oceniac jakosciowo (na podstawie rysunków przebiegów sygnałów) oraz ilosciowo, wyznaczajac wskaznik jakosci regulacji

gdzie kkonc oznacza koniec symulacji (zawsze taki sam). Zamiescic wybrane wyniki symulacji (przebiegi sygnałów wejsciowych i wyjsciowych procesu oraz wartosci wskaznika E).Wprzypadku algorytmu PID rozwazyc kilka mozliwych konfiguracji regulatora, tzn. uchyb e1 pierwszego wyjscia oddziałuje na pierwszy sygnał sterujacy u1, uchyb e2 oddziałuje na u2, uchyb e3 oddziałuje na u3 itd. Zamiescic wybrane wyniki symulacji

1.4.1. Konfiguracja i dobór parametrów regulatorów PID

Pid bez u1

Kr1=1.5 Td1=0.01 Ti1=2 Kr2=5.5 Td2=0.2 Ti2=0.4 Kr3=2 Td3=1 Ti3=9 Wartość wskaźnika E=79.893

Pid bez u2

Kr1=0.7 Td1=0.3 Ti1=0.2 Kr2=3.5 Td2=0.2 Ti2=0.2 Kr3=3 Td3=0.8 Ti3=7.5 Wartość wskaźnika E=64.3445

Pid bez u3

Kr1=0.7 Td1=0.3 Ti1=0.2 Kr2=0.7 Td2=0.6 Ti2=1.8 Kr3=1.4 Td3=0.6 Ti3=5.5 Wartość wskaźnika E=74.1529

Pid bez u4

Kr1=0.7 Td1=0.3 Ti1=0.2 Kr2=0.6 Td2=0.05 Ti2=0.3 Kr3=0.8 Td3=0.4 Ti3=0.4 Wartość wskaźnika E=79.2468

1.4.2. Dobór parametrów regulatorów DMC

DMC1

D=350 N=100 Nu=15 lambda1=0.9 lambda2=0.5 lambda3=0.2 lambda4=0.1 psi1=1 psi2=0.54 psi3=1.3 Wartość wskaźnika E=164.4355

DMC2

D=350 N=30 Nu=5 lambda1=0.9 lambda2=1.2 lambda3=0.5 lambda4=1 psi1=0.89 psi2=1 psi3=1.5 Wartość wskaźnika E=167.2731

DMC BEST

D=350 N=200 Nu=10 lambda1=0.15 lambda2=0.2 lambda3=0.7 lambda4=0.1 psi1=1 psi2=1 psi3=1 Wartość wskaźnika E=103.2769

1.5. Optymalizacja parametrów regulatorów PID i DMC

Dla zaproponowanej trajektorii zmian sygnału zadanego dobrano nastawy regulatora PID w wyniku optymalizacji wskaźnika jakości regulacji E. Optymalizacji dokonano za pomocą wbudowanej funkcji MATLABa fmincon.

1.5.1. Optymalizacja PID

PID bez u1

Kr1=2.438 Td1=0 Ti1=3.2542 Kr2=8.8647 Td2=0 Ti2=0.2623 Kr3=3.1042 Td3=1.0262 Ti3=16.8144 Wartość wskaźnika E=64.2991

PID bez u2

Kr1=2.2901 Td1=0.0187 Ti1=0.5102 Kr2=0.0219 Td2=39.5656 Ti2=0.0006 Kr3=4.8545 Td3=0.4969 Ti3=17.6086 Wartość wskaźnika E=51.5533

PID bez u3

Kr1=2.4231 Td1=0 Ti1=0.6438 Kr2=1.3759 Td2=0 Ti2=1.173 Kr3=6.4637 Td3=0.0984 Ti3=13.3023 Wartość wskaźnika E=42.5008

PID bez u4

Kr1=2.5885 Td1=0 Ti1=0.5918 Kr2=1.4396 Td2=0 Ti2=0.6631 Kr3=13.4882 Td3=0.029 Ti3=4.4987 Wartość wskaźnika E=43.3297

1.5.2. Optymalizacja DMC

W regulatorze DMC dobierano współczynniki u1, u2, u3, lambda1, lambda2, lambda3, lambda4, natomiast horyzonty D, N, Nu przyjęto stałe.

D=350 N=200 Nu=10 lambda1=0.9004 lambda2=-5.5093 lambda3=12.1642 lambda4=-0.0569 psi1=-0.5147 psi2=-9.3173 psi3=3.2215 Wartość wskaźnika E=77.9097

1.5.3. Wnioski

Biorąc pod uwagę ocenę regulacji jakościową oraz ilościową najlepiej prezentuje się algorytm PID z dobranymi parametrami za pomocą optymalizacji funkcją fmincon, pokazuje to że prostota regulatora PID w takiej sytuacji dała możliwość lepszego dostrojenia oraz lepszych wyników.

Porównanie najlepszego DMC i PID

PID – PID bez u3
Kr1=2.4231 Td1=0 Ti1=0.6438 Kr2=1.3759
Td2=0 Ti2=1.173 Kr3=6.4637
Td3=0.0984 Ti3=13.3023
Wartość wskaźnika E=42.5008

DMC – D=350 N=200 Nu=10
lambda1=0.9004 lambda2=-5.5093 lambda3=12.1642 lambda4=-0.0569
psi1=-0.5147 psi2=-9.3173psi3=3.2215
Wartość wskaźnika E=77.9097
Wskaźnik jakości jest prawie dwa razy większy w przypadku regulatora DMC

1.6. Implementacja algorytmu DMC w wersji klasycznej

1.6.1. Program do symulacji algorytmu DMC w wersji klasycznej

Nastawy regulatora DMC dobrane eksperymentalnie i optymalizacyjnie

Listing 1.13. "Nastawy regulatora DMC dobrane eksperymentalnie i optymalizacyjnie""

```
%zadanie 6 - Skrypt relizujacy algorytm DMC
%            regulatora klasycznego wielowymiarowego
clear all

%nastawy regulatora DMC
D = 350;%horyzont dynamiki

N = 200;%horyzont predykcji
Nu = 10;%horyzont sterowania

eks = 0;
%dobrane eksperymentalnie
if eks == 1
    lambda1 = 0.15;
    lambda2 = 0.2;
    lambda3 = 0.7;
    lambda4 = 0.1;
    psi1 = 1;
    psi2 = 1;
    psi3 = 1;
else
    lambda1 = 0.9;
    lambda2 = 1.2;
    lambda3 = 0.5;
    lambda4 = 1;
    psi1 = 0.89;
    psi2 = 1;
    psi3 = 1.5;
    N = 30;%horyzont predykcji
    Nu = 5;%horyzont sterowania
end
```

Wyznaczanie macierzy S, Mp i M regulatora DMC

Listing 1.14. "Wyznaczanie macierzy S Mp i M regulatora DMC"

```
%warunki poczatkowe
nu = 4;
ny = 3;
load('s.mat')
%Macierz odopowiedzi skokowych
for i = 1:D
    S(i)={[s11(i) s12(i) s13(i) s14(i);...
           s21(i) s22(i) s23(i) s24(i);...
           s31(i) s32(i) s33(i) s34(i)]};
end

% Macierz predykcji
for i = 1:N
   for j = 1:D-1
      if i+j <= D
         Mp{i,j} = S{i+j}-S{j};
      else
         Mp{i,j} = S{D}-S{j};
      end
   end
end

% Macierz M
for i=1:Nu
    M(i:N,i)=S(1:N-i+1);
    M(1:i-1,i)={[0 0 0 0; 0 0 0 0; 0 0 0 0]};
end
```

Wyznaczanie macierzy Lambda, Psi i K regulatora DMC

Listing 1.15. "Wyznaczanie macierzy Lambda Psi i K regulatora DMC"

```
%Macierz lambda
for i=1:Nu
    for j=1:Nu
        if i==j
            Lambda{i,j}=[lambda1 0 0 0; 0 lambda2 0 0;...
                         0 0 lambda3 0; 0 0 0 lambda4];
        else
            Lambda{i,j}=[0 0 0 0; 0 0 0 0;...
                         0 0 0 0; 0 0 0 0;];
        end
    end
end

%Macierz Psi
for i=1:N
    for j=1:N
        if i==j
            Psi{i,j}=[psi1 0 0; 0 psi2 0; 0 0 psi3];
        else
            Psi{i,j}=[0 0 0; 0 0 0; 0 0 0];
        end
    end
end

%Macierz K
for i = 1:Nu
    size(i) = nu;
end
for i=1:N
    size2(i) = ny;
end

M_temp = cell2mat(M);
M_temp_tr = M_temp';
Psi_temp = cell2mat(Psi);
M_temp_tr = M_temp_tr * Psi_temp;
M_temp = mat2cell(M_temp_tr*M_temp,size,size);
for i=1:Nu
    for j=1:Nu
        L{i,j} = M_temp{i,j}+Lambda{i,j}; %Dodanie Lambdy
    end
end
L_temp = cell2mat(L);
L_temp_rev = mat2cell(L_temp^(-1),size,size);
L_temp_rev = cell2mat(L_temp_rev);
```

Inicjalizacja macierzy do przechowywania danych

Listing 1.16. "Inicjalizacja macierzy do przechowywania danych"

```
%Macierze do przechowywania danych
u_d = zeros(nu,1);
for i = 1:D-1
    u_delta(i,1) = {u_d};
end

y_z = zeros(ny,1);
for i = 1:N
    y_zad_mod(i,1) = {y_z};
end

Y_dmc = zeros(ny,1);
for i=1:N
    y_mod(i,1)={Y_dmc};
end

for i=1:N
    Y0(i,1)={[0;0]};
end

du = zeros(nu,1);
for i=1:Nu
    dU_mod(i,1)={du};
end
```

Parametry symulacji

Listing 1.17. "Parametry symulacji"

```
%parametry symulacji
kk = 1600;
start = 10;
u1 = zeros(1,kk);
u2 = zeros(1,kk);
u3 = zeros(1,kk);
u4 = zeros(1,kk);
y1 = zeros(1,kk);
y2 = zeros(1,kk);
y3 = zeros(1,kk);

Ey = zeros(ny,1);

y1_zad = zeros(1,kk);
y1_zad(start:400) = 1;
y1_zad(400:800) = 1.5;
y1_zad(800:1200) = 0.6;
y1_zad(1200:kk) = 2.5;

y2_zad = zeros(1,kk);
y2_zad(start:400) = 2;
y2_zad(400:800) = 1.2;
y2_zad(800:1200) = 0;
y2_zad(1200:kk) = 1.5;

y3_zad = zeros(1,kk);
y3_zad(start:400) = 1.5;
y3_zad(400:800) = 0.8;
y3_zad(800:1200) = 2;
y3_zad(1200:kk) = 0.2;
```

Główna pętla symulacyjna

Listing 1.18. "Główna pętla symulacyjna"

```
for k = start:kk
   % Równanie róznicowe
   [y1(k),y2(k),y3(k)] = symulacja_obiektu7(u1(k-1),u1(k-2),u1(k-3),u1(k-4),...
       u2(k-1),u2(k-2),u2(k-3),u2(k-4),u3(k-1),u3(k-2),u3(k-3),u3(k-4),...
       u4(k-1),u4(k-2),u4(k-3),u4(k-4),y1(k-1),y1(k-2),y1(k-3),y1(k-4),...
       y2(k-1),y2(k-2),y2(k-3),y2(k-4),y3(k-1),y3(k-2),y3(k-3),y3(k-4));

   % Regulator
    Y_dmc(1) = y1(k);
    Y_dmc(2) = y2(k);
    Y_dmc(3) = y3(k);
    for i=1:N
        y_mod(i,1)={Y_dmc};
    end

    Y_zad(1) = y1_zad(k);
    Y_zad(2) = y2_zad(k);
    Y_zad(3) = y3_zad(k);

    for i=1:N
        y_zad_mod(i,1) = {Y_zad'};
    end

    %obliczanie Y0
    Mp_tmp = cell2mat(Mp);
    u_delta_tmp = cell2mat(u_delta);
    Y0_tmp = mat2cell(Mp_tmp * u_delta_tmp,size2,[1]);
    for i = 1:N
        Y0{i,1} = y_mod{i,1} + Y0_tmp{i,1};
    end

    %obliczanie dU
    for i = 1:N
        uchyb{i,1} = y_zad_mod{i,1} - Y0{i,1};
    end
    K_tmp = cell2mat(K);
    uchyb_tmp = cell2mat(uchyb);
    dU_mod = mat2cell(K_tmp*uchyb_tmp,size,[1]);
    du = dU_mod{1};

    for n = D-1:-1:2
      u_delta(n) = u_delta(n-1);
    end

    u1(k) = u1(k-1) + du(1);
    u2(k) = u2(k-1) + du(2);
    u3(k) = u3(k-1) + du(3);
    u4(k) = u4(k-1) + du(4);
    u_delta(1,1) = {du};

   %bledy
   Ey(1) = Ey(1) + (y1_zad(k) - y1(k))^2;
   Ey(2) = Ey(2) + (y2_zad(k) - y2(k))^2;
   Ey(3) = Ey(3) + (y3_zad(k) - y3(k))^2;

end
```

1.6.2. Porównanie implementacji regulatorów DMC

DMC1

D=350 N=100 Nu=15 lambda1=0.9 lambda2=0.5 lambda3=0.2 lambda4=0.1 psi1=1 psi2=0.54psi3=1.3 Wartość wskaźnika E=164.4355

DMC2

D=350 N=30 Nu=5 lambda1=0.9 lambda2=1.2 lambda3=0.5 lambda4=1 psi1=0.89 psi2=1 psi3=1.5 Wartość wskaźnika E=167.2731

DMC BEST

D=350 N=200 Nu=10 lambda1=0.15 lambda2=0.2 lambda3=0.7 lambda4=0.1 psi1=1 psi2=1 psi3=1 Wartość wskaźnika E=103.2769

1.6.3. Wnioski

Otrzymane wyniki symulacji dla wybranego zestawu parametrów są takie same jak w wersji klasycznej.
Algorytm DMC w najprostszej wersji uzyskał taki sam wskaźnik regulacji co DMC w klasycznej wersji, a dzięki uproszczeniu obliczeń jest szybszy.

2. Laboratorium

2.1. Stanowisko grzejąco-chłodzące

2.1.1. Sprawdzenie poprawności punktu pracy

Sygnały sterujące ustawione został na wskazane w poleceniu wartości: G1 = 32, G2 = 37, W1=W2=50. Sprawdzona została możliwość sterowania i pomiaru w komunikacji ze stanowiskiem. Wartości temperatur w punkcie pracy wyniosły: T1 = 36,3 T2 = 38,3

2.1.2. Zabezpieczenia stanowiska

W celu zabezpieczenia stanowiska przed uszkodzeniem na sterowniku został zaimplementowany mechanizm, który przy przekroczeniu temperatury 150 ◦ C wyłącza grzałkę sąsiadującą z czujnikiem, który zmierzył niebezpieczną temperaturę. Implementacja takiego mechanizmu jest prosta, ale niezwykle istotna w tego typu procesach. Zadeklarowano wartość krytyczną temperatury oraz zaimplementowano funkcję sprawdzającą czy wskazanie czujnika nie przekracza tej wartości. W przypadku jej przekroczenia grzałka sąsiadująca z danym czujnikiem zostaje wyłączona, sterowanie G zostaje ustawione na 0.

Implementacja mecahnizmu zabezpieczeń stanowiska na PLC

Listing 2.1. "Implementacja mechanizmu zabezpieczeń stanowiska na PLC"

```
// Program: Modbus; Typ: Scan
// Jesli czujnik T1 wskazuje ponad 150 stopni
IF(T_1 >= K15000) THEN
        SET(TRUE, T_1_alarm);
END_IF;

// Jesli czujnik T1 wskazuje ponad 150 stopni
IF(T_3 >= K15000) THEN
        SET(TRUE, T_3_alarm);
END_IF;

// Jesli aktywny alarm T1 wylacz grzalke G1
IF(T_1_alarm) THEN
        MOV(TRUE, K0, G_1);
END_IF;

// Jesli aktywny alarm T3 wylacz grzalke G2
IF(T_3_alarm) THEN
        MOV(TRUE, K0, G_2);
END_IF;
```

2.1.3. Implementacja oraz dobór nastaw dwupętlowego regulatora PID

Na sterowniku zaimplementowano uwzględniając ograniczenia dwupętlowy regulator PID. Metodą eksperymentalną dobrano nastawy regulatora.

Implementacja regulatora PID toru T1-G1

Listing 2.2. "Inicjacja parametrów regulatora PID toru T1-G1"

```
//Program: Init; Typ: Initial
//init regulatora PID_G1
PID_G1.K_gain := 30.1;
PID_G1.TI := 25.0;
PID_G1.TD := 1.0;
PID_G1.Ep0 := 0.0;
PID_G1.Ep1 := 0.0;
PID_G1.Ep2 := 0.0;
PID_G1.Rp0 := 0.0;
PID_G1.Rp1 := 0.0;
PID_G1.Rp2 := 0.0;
PID_G1.sampling_time := 4.0;
PID_G1.SV := 36.68;
PID_G1.MV := 32.0;
```

Listing 2.3. "Program regulatora PID toru T1-G1"

```
//Program: PID_R; Typ: FixedScan 4000ms
IF PID_G1.Control_ON THEN
//      //Ustawienie wartosci PV
        PID_G1.PV := INT_TO_REAL(T_1)/100.0;


        //Wyliczenie parametrow
        //r0 = K*( 1+(Tp/(2*Ti))+Td/Tp );
        PID_G1.Rp0 := PID_G1.K_gain*(
                1.0
                +(PID_G1.sampling_time/(2.0*PID_G1.TI))
                +PID_G1.TD/PID_G1.sampling_time
                );
        //r1 = K*( (Tp/(2*Ti))-(2*Td/Tp)-1 );
        PID_G1.Rp1 := PID_G1.K_gain*(
                (PID_G1.sampling_time/(2.0*PID_G1.TI))
                -(2.0*PID_G1.TD/PID_G1.sampling_time)
                -1.0
                );
        //K*Td/Tp;
        PID_G1.Rp2 := PID_G1.K_gain*PID_G1.TD/PID_G1.sampling_time;

        //Wyliczenie uchybu regulacji i przesuniecie historii
        PID_G1.Ep2 := PID_G1.Ep1;
        PID_G1.Ep1 := PID_G1.Ep0;
        PID_G1.Ep0 := PID_G1.SV - PID_G1.PV;

        //Obliczenie sterowania
        //u = R2*E2 + R1*E1 + R0*E0 + u;
        PID_G1.MV := PID_G1.Rp2*PID_G1.Ep2
                +PID_G1.Rp1*PID_G1.Ep1
                +PID_G1.Rp0*PID_G1.Ep0
                +PID_G1.MV;

        //Ograniczenia sterowania
        IF (PID_G1.MV > 100.0) THEN
                PID_G1.MV := 100.0;
        END_IF;

        IF (PID_G1.MV < 0.0) THEN
                PID_G1.MV := 0.0;
        END_IF;

        G_1 := REAL_TO_INT(PID_G1.MV*10.0);
END_IF;
```

Wyznaczenie modelu obiektu

Dobranie nastaw regulatora

2.1.4. Implementacja i dobór parametrów regulatora DMC 2x2

Na sterowniku zaimplementowano uwzględniając ograniczenia regulator DMC 2x2 w wersji oszczędnej obliczeniowo(analitycznej). Pozyskano odpowiedzi skokowe obiektu. Dobierając parametry regulatora uwzględniono: Liczbę wykorzystanych rejestrów pamięci, czas obliczeń pojedynczej iteracji algorytmu oraz jakość regulacji Implementacja Wykresy

Implementacja

Odpowiedzi skokowe

Dobór parametrów regulatora

2.1.5. Panel operatora

Panel operatora Wartości mierzone,zadane oraz sterowanie

2.1.6. Automat stanów

Zaimplementowac automat stanów, na podstawie którego modyfikowane beda wartosci zadane. Opisac implementacje.

2.2. Stanowisko INTECO - zbiorniki wodne

2.2.1. Konfiguracja sterownika stanowiska Inteco

Skonfigurowac sterownik w celu obsługi stanowiska Inteco. Opisac zastosowana konfiguracje.

2.2.2. Zabezpieczenia stanowiska

Zaimplementowac na sterowniku mechanizm zabezpieczajacy przed uszkodzeniem stanowiska. Omówic zastosowane podejscie.

2.2.3. Charakterystyka statyczna

Spróbowac wyznaczyc charakterystyke statyczna. Omówic wyniki.

2.2.4. Dostosowanie i dobieranie parametrów regulatorów PID

Dostosowac implementacje regulatora PID (wielopetlowego) do współpracy ze stanowiskiem Inteco. Regulator(y) dostroic. Omówic proces dobierania nastaw regulatorów. Uwzglednic ograniczenia jesli istnieja. Zamiescic wykresy w sprawozdaniu.

2.2.5. Automat stanów

Dostosowac automat stanów, na podstawie którego modyfikowane beda wartosci zadane.

2.2.6. Wizualizacja procesu

Przygotowac wizualizacje procesu: — jego szczegółowa reprezentacje graficzna, — wykres sygnałów wyjsciowych, wartosci zadanych oraz sterowania, — graf przejsc automatu stanów.

2.3. Porównanie regulatorów PID

Porównac działanie własnej implementacji regulatora PID z działaniem wbudowanej w sterownik funkcji PID. Sprawdzic wpływ ograniczen na działanie obu wersji regulatora. Omówic parametry zastosowane w funkcji PID.