Spis treści

1. Projekt

1.1. Poprawność podanego punktu pracy

Zasymulowano odpowiedź procesu w punkcie pracy dla sterowania $u_1 = u_2 = u_3 = u_4 = 0$
Wyjścia obiektu wynoszą $y_1 = y_2 = y_3 = y_4 = 0$.
Podany punkt pracy jest poprawny.

1.2. Odpowiedzi skokowe 12 torów procesów

Wyznaczono odpowiedzi skokowe dla 12 torów procesu to znaczy zestaw liczb $s_1^{m,n}, s_2^{m,n}, ...$ dla $m = 1, 2, 3$, gdzie m oznacza numer wyjścia y i $n = 1, 2, 3, 4$, gdzie n oznacza numer sterowania u przy pojedynczych skokach jednostkowych odpowiednich sygnałów sterujących od chwili $k = 0$ włącznie sygnał wymusznia ma wartość 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)

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 różnej amplitudzie) dobrano nastawy regulatora PID i parametry algorytmu DMC metodą eksperymentalną. Jakość regulacji oceniano jakościowo (na podstawie rysunków przebiegów sygnałów) oraz ilościowo, wyznaczając wskaźnik jakości regulacji

$$E = \sum_{k=1}^{k_{konc}} \sum_{m=1}^{3} (y_{zad}(k) - y(k))^2$$

gdzie k_{konc} oznacza koniec symulacji (zawsze taki sam). Poniżej zamieszczono wybrane wyniki symulacji (przebiegi sygnałów wejściowych i wyjściowych procesu oraz wartości wskaźnika E). W przypadku algorytmu PID przedstawiono kilka możliwych konfiguracji regulatora.

1.4.1. Konfiguracja i dobór parametrów regulatorów PID

Konfiguracja regulatorów PID niesterująca wejściem u_1

	Kr	Ti	Td
PID1	—	—	—
PID2	1.5	2	0.01
PID3	5.5	0.4	0.2
PID4	2.0	9.0	1.0

Tab. 1.1. Nastawy konfiguracji regulatorów bez PID1

Wartość wskaźnika $E = 79{,}893$.

Konfiguracja regulatorów PID niesterująca wejściem u_2

	Kr	Ti	Td
PID1	0.7	0.2	0.3
PID2	—	—	—
PID3	3.5	0.2	0.2
PID4	3.0	7.5	0.8

Tab. 1.2. Nastawy konfiguracji regulatorów bez PID2

Wartość wskaźnika $E = 64{,}3445$.

Konfiguracja regulatorów PID niesterująca wejściem u_3

	Kr	Ti	Td
PID1	0.7	0.2	0.3
PID2	0.7	1.8	0.6
PID3	—	—	—
PID4	1.4	5.5	0.6

Tab. 1.3. Nastawy konfiguracji regulatorów bez PID3

Wartość wskaźnika $E = 74{,}1529$.

Konfiguracja regulatorów PID niesterująca wejściem u_4

	Kr	Ti	Td
PID1	0.7	0.2	0.3
PID2	0.6	0.3	0.05
PID3	0.8	0.4	0.4
PID4	—	—	—

Tab. 1.4. Nastawy konfiguracji regulatorów bez PID4

Wartość wskaźnika $E = 79{,}2468$.

1.4.2. Dobór parametrów regulatorów DMC

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

Pierwszy eksperyment

$$D = 350 \quad N = 100 \quad Nu = 15$$
$$\psi_1 = 1 \quad \psi_2 = 0{,}54 \quad \psi_3 = 1{,}3$$
$$\lambda_1 = 0{,}9 \quad \lambda_2 = 0{,}5 \quad \lambda_3 = 0{,}2 \quad \lambda_4 = 0{,}1$$

Wartość wskaźnika $E = 164{,}4355$

Drugi eksperyment

$$D = 350 \quad N = 30 \quad Nu = 5$$
$$\psi_1 = 0{,}89 \quad \psi_2 = 1{,}0 \quad \psi_3 = 1{,}5$$
$$\lambda_1 = 0{,}9 \quad \lambda_2 = 1{,}2 \quad \lambda_3 = 0{,}5 \quad \lambda_4 = 1{,}0$$

Wartość wskaźnika $E = 167{,}2731$

Trzeci ekperyment

$$D = 350 \quad N = 200 \quad Nu = 10$$
$$\psi_1 = 1{,}0 \quad \psi_2 = 1{,}0 \quad \psi_3 = 1{,}0$$
$$\lambda_1 = 0{,}15 \quad \lambda_2 = 0{,}2 \quad \lambda_3 = 0{,}7 \quad \lambda_4 = 0{,}1$$

Wartość wskaźnika $E = 103{,}2769$
Nastawy regulatorów DMC z trzeciego eksperymentu dały najlepszą jakość regualcji.

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

Konfiguracja regulatorów PID niesterująca wejściem u_1

	Kr	Ti	Td
PID1	—	—	—
PID2	2.438	3.2542	0.0
PID3	8.8647	0.2623	0.0
PID4	3.1042	16.8144	1.0262

Tab. 1.5. Nastawy konfiguracji regulatorów bez PID1

Wartość wskaźnika $E = 64{,}2991$.

Konfiguracja regulatorów PID niesterująca wejściem u_2

	Kr	Ti	Td
PID1	2.2901	0.5102	0.0187
PID2	—	—	—
PID3	0.0219	0.0006	39.5656
PID4	4.8545	17.6086	0.4969

Tab. 1.6. Nastawy konfiguracji regulatorów bez PID2

Wartość wskaźnika $E = 51{,}5533$.

Konfiguracja regulatorów PID niesterująca wejściem u_3

	Kr	Ti	Td
PID1	2.4231	0.6438	0.0
PID2	1.3759	1.173	0.0
PID3	—	—	—
PID4	6.4637	13.3023	0.0984

Tab. 1.7. Nastawy konfiguracji regulatorów bez PID3

Wartość wskaźnika $E = 42{,}5008$.

Konfiguracja regulatorów PID niesterująca wejściem u_4

	Kr	Ti	Td
PID1	2.5885	0.5918	0.0
PID2	1.4396	0.6631	0.0
PID3	13.4882	4.4987	0.029
PID4	—	—	—

Tab. 1.8. Nastawy konfiguracji regulatorów bez PID4

Wartość wskaźnika $E = 43{,}3297$.

1.5.2. Optymalizacja DMC

W regulatorze DMC dobierano współczynniki $\psi_1, \psi_2, \psi_3, \lambda_1, \lambda_2, \lambda_3, \lambda_4$, natomiast horyzonty D, N, N_u przyjęto stałe.

$$D = 350 \quad N = 200 \quad Nu = 10$$

$$\psi_1 = -0{,}5147 \quad \psi_2 = -9{,}3173 \quad \psi_3 = 3{,}2215$$

$$\lambda_1 = 0{,}9004 \quad \lambda_2 = -5{,}5093 \quad \lambda_3 = 12{,}1642 \quad \lambda_4 = -0{,}0569$$

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 osiągnięcia lepszych wyników.

Porównanie najlepszego DMC i PID

Konfiguracja regulatorów PID niesterująca wejściem u_3

	Kr	Ti	Td
PID1	2.4231	0.6438	0.0
PID2	1.3759	1.173	0.0
PID3	—	—	—
PID4	6.4637	13.3023	0.0984

Tab. 1.9. Nastawy konfiguracji regulatorów bez PID3

Wartość wskaźnika $E = 42{,}5008$.

Regulator DMC z optymalizacji

$$D = 350 \quad N = 200 \quad Nu = 10$$

$$\psi_1 = -0{,}5147 \quad \psi_2 = -9{,}3173 \quad \psi_3 = 3{,}2215$$

$$\lambda_1 = 0{,}9004 \quad \lambda_2 = -5{,}5093 \quad \lambda_3 = 12{,}1642 \quad \lambda_4 = -0{,}0569$$

Wartość wskaźnika $E = 77{,}9097$.

Wskaźnik jakości jest prawie dwa razy większy w przypadku regulatora DMC. Regulator PID okazał się lepiej sterującym, jednak wymagał większego nakładu pracy na etapie strojenia.

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

Pierwszy eksperyment

$$D = 350 \quad N = 100 \quad Nu = 15$$
$$\psi_1 = 1 \quad \psi_2 = 0{,}54 \quad \psi_3 = 1{,}3$$
$$\lambda_1 = 0{,}9 \quad \lambda_2 = 0{,}5 \quad \lambda_3 = 0{,}2 \quad \lambda_4 = 0{,}1$$

Wartość wskaźnika $E = 164{,}4355$

Drugi eksperyment

$$D = 350 \quad N = 30 \quad Nu = 5$$
$$\psi_1 = 0{,}89 \quad \psi_2 = 1{,}0 \quad \psi_3 = 1{,}5$$
$$\lambda_1 = 0{,}9 \quad \lambda_2 = 1{,}2 \quad \lambda_3 = 0{,}5 \quad \lambda_4 = 1{,}0$$

Wartość wskaźnika $E = 167{,}2731$

Trzeci ekperyment

$$D = 350 \quad N = 200 \quad Nu = 10$$
$$\psi_1 = 1{,}0 \quad \psi_2 = 1{,}0 \quad \psi_3 = 1{,}0$$
$$\lambda_1 = 0{,}15 \quad \lambda_2 = 0{,}2 \quad \lambda_3 = 0{,}7 \quad \lambda_4 = 0{,}1$$

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. Wyznaczając model dobrano nastawy regulatora.

Implementacja regulatora PID

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

```
//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. "Inicjacja parametrów regulatora PID toru G2-T3"

```
//Program: Init; Typ: Initial
//init regulatora PID_G2
PID_G2.K_gain := 30.7;
PID_G2.TI := 25.5;
PID_G2.TD := 1.0;
PID_G2.Ep0 := 0.0;
PID_G2.Ep1 := 0.0;
PID_G2.Ep2 := 0.0;
PID_G2.Rp0 := 0.0;
PID_G2.Rp1 := 0.0;
PID_G2.Rp2 := 0.0;
PID_G2.sampling_time := 4.0;
PID_G2.SV := 38.68;
PID_G2.MV := 37.0;
```

Listing 2.4. "Program regulatora PID toru G1-T1"

```
//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;
```

Listing 2.5. "Program regulatora PID toru G2-T3"

```
//Program: PID_R; Typ: FixedScan 4000ms
IF PID_G2.Control_ON THEN
        //Ustawienie wartosci PV
        PID_G2.PV := INT_TO_REAL(T_3)/100.0;

        //Wyliczenie parametrow
        //r0 = K*( 1+(Tp/(2*Ti))+Td/Tp );
        PID_G2.Rp0 := PID_G2.K_gain*(
                1.0
                +(PID_G2.sampling_time/(2.0*PID_G2.TI))
                +PID_G2.TD/PID_G2.sampling_time
                );
        //r1 = K*( (Tp/(2*Ti))-(2*Td/Tp)-1 );
        PID_G2.Rp1 := PID_G2.K_gain*(
                (PID_G2.sampling_time/(2.0*PID_G2.TI))
                -(2.0*PID_G2.TD/PID_G2.sampling_time)
                -1.0
                );
        //K*Td/Tp;
        PID_G2.Rp2 := PID_G2.K_gain*PID_G2.TD/PID_G2.sampling_time;

        //Wyliczenie uchybu regulacji i przesuniecie historii
        PID_G2.Ep2 := PID_G2.Ep1;
        PID_G2.Ep1 := PID_G2.Ep0;
        PID_G2.Ep0 := PID_G2.SV - PID_G2.PV;

        //Obliczenie sterowania
        //u = R2*E2 + R1*E1 + R0*E0 + u;
        PID_G2.MV := PID_G2.Rp2*PID_G2.Ep2
                + PID_G2.Rp1*PID_G2.Ep1
                + PID_G2.Rp0*PID_G2.Ep0
                + PID_G2.MV;

        //ANTI WIND UP
        IF (PID_G2.MV > 100.0) THEN
                PID_G2.MV := 100.0;
        END_IF;

        IF (PID_G2.MV < 0.0) THEN
                PID_G2.MV := 0.0;
        END_IF;

        G_2 := REAL_TO_INT(PID_G2.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

Listing 2.6. "Skrypt generujący parametry regulatora DMC"

```
% Opis: Skrypt wyliczajacy parametry regulatora DMC
% przeznaczonego do uruchomienia na PLC

% Zalozone parametry regulatora
lam = 1
load('../zad1/esy/s1.mat')
s = s1aprox;
D1 = length(s); % horyzont dynamiki
N1=D1;
Nu1=N1;
lambda1 = lam;
D = D1; % horyzont dynamiki
N=N1;
Nu=Nu1;
lambda = lambda1;
run('DMC_init.m');
Ke1 = sum(K(1,:));
Ku1 = K(1,:)*Mp;

load('../zad1/esy/s2.mat')
s = s2aprox;
D2 = length(s); % horyzont dynamiki
N2=D2;
Nu2=N2;
lambda2 = lam;
D = D2; % horyzont dynamiki
N=N2;
Nu=Nu2;
lambda = lambda2;
run('DMC_init.m');
Ke2 = sum(K(1,:));
Ku2 = K(1,:)*Mp;

% wyeksportowanie wyznaczonych parametrow do
%pliku naglowkowego zgodnego ze standardem jezyka ST
run('exporter.m');
```

Listing 2.7. "Skrypt wyliczający parametry regulatora DMC"

```
% Opis: Skrypt wyliczajacy parametry regulatora DMC

% Wyznaczenie macierzy M
M = zeros(D,D);
for kNu=1:Nu
    M(kNu:N,kNu) = s(1:(N+1-kNu));
end

% Wyznaczenie macierzy Mp
Mp = ones(D,D-1)*s(end);
for kD=1:D-1
    Mp(1:(N-kD),kD) = s((kD+1):(N))';
end
Mp = Mp - ones(D,1)*s(1:end-1);

fi = eye(D);
LAMBDA = lambda*eye(D);
% Wyznaczenie macierzy K
K = inv((M')*M+LAMBDA)*(M');
```

Listing 2.8. "Skrypt eksportujący parametry regulatora DMC do pliku"

```
% Opis: Skrypt eksportujacy wyliczone parametry regulatora DMC do postaci
% zgodnej ze standardem jezyka ST

% powstanie plik "DMC_data.h" w folderze Inc
fileID = fopen('DMC_data.st','w');

fprintf(fileID,'//Parametry regulatora DMC1\n');
fprintf(fileID,'DMC_G1.D_ := %d;\n', D1);
fprintf(fileID,'DMC_G1.N_ := %d;\n', N1);
fprintf(fileID,'DMC_G1.Nu := %d;\n', Nu1);
fprintf(fileID,'DMC_G1.lambda := %f;\n\n', lambda1);

fprintf(fileID,'//Parametry regulatora DMC2\n');
fprintf(fileID,'DMC_G2.D_ := %d;\n', D2);
fprintf(fileID,'DMC_G2.N_ := %d;\n', N2);
fprintf(fileID,'DMC_G2.Nu := %d;\n', Nu2);
fprintf(fileID,'DMC_G2.lambda := %f;\n\n', lambda2);

fprintf(fileID,'// Przeliczone wartosci do sterowania DMC1\n');
fprintf(fileID,'DMC_G1.Ke := %f;\n\n', Ke1);
fprintf(fileID,'// Przeliczone wartosci do sterowania DMC2\n');
fprintf(fileID,'DMC_G2.Ke := %f;\n\n', Ke2);

for n=1:length(Ku1)
    fprintf(fileID,'DMC_G1.Ku[%d] := %f;\n', n-1, Ku1(n));
end
fprintf(fileID,'\n');

for n=1:length(Ku2)
    fprintf(fileID,'DMC_G2.Ku[%d] := %f;\n', n-1, Ku2(n));
end
fprintf(fileID,'\n');

for n=1:length(Ku1)
    fprintf(fileID,'DMC_G1.delta_u_past[%d] := %f;\n', n-1, 0.0);
end
fprintf(fileID,'\n');

for n=1:length(Ku2)
    fprintf(fileID,'DMC_G2.delta_u_past[%d] := %f;\n', n-1, 0.0);
end
fprintf(fileID,'\n');

fclose(fileID);
```

Listing 2.9. "Wygenerowany program ST inicjujący regulatory DMC"

```
//W celu pokazania w sprawozdaniu kod zostal skrocony
//W miejscach obciecia pozostawiono trzy kropki

//Parametry regulatora DMC1
DMC_G1.D_ := 75;
DMC_G1.N_ := 75;
DMC_G1.Nu := 75;
DMC_G1.lambda := 0.200000;
//Parametry regulatora DMC2
DMC_G2.D_ := 75;
DMC_G2.N_ := 75;
DMC_G2.Nu := 75;
DMC_G2.lambda := 0.200000;
// Przeliczone wartosci do sterowania DMC1
DMC_G1.Ke := 1.650272;
// Przeliczone wartosci do sterowania DMC2
DMC_G2.Ke := 1.686157;

DMC_G1.Ku[0] := 0.597954;
DMC_G1.Ku[1] := 0.749775;
DMC_G1.Ku[2] := 0.907605;
DMC_G1.Ku[3] := 1.026848;
...
DMC_G1.Ku[71] := 0.009323;
DMC_G1.Ku[72] := 0.005970;
DMC_G1.Ku[73] := 0.002869;

DMC_G2.Ku[0] := 0.555568;
DMC_G2.Ku[1] := 0.684826;
DMC_G2.Ku[2] := 0.816335;
DMC_G2.Ku[3] := 0.947948;
...
DMC_G2.Ku[71] := 0.013145;
DMC_G2.Ku[72] := 0.008487;
DMC_G2.Ku[73] := 0.004111;

DMC_G1.delta_u_past[0] := 0.000000;
DMC_G1.delta_u_past[1] := 0.000000;
DMC_G1.delta_u_past[2] := 0.000000;
...
DMC_G1.delta_u_past[71] := 0.000000;
DMC_G1.delta_u_past[72] := 0.000000;
DMC_G1.delta_u_past[73] := 0.000000;

DMC_G2.delta_u_past[0] := 0.000000;
DMC_G2.delta_u_past[1] := 0.000000;
DMC_G2.delta_u_past[2] := 0.000000;
...
DMC_G2.delta_u_past[71] := 0.000000;
DMC_G2.delta_u_past[72] := 0.000000;
DMC_G2.delta_u_past[73] := 0.000000;
```

Listing 2.10. "Program ST implementujący algorytm DMC toru G1-T1"

```
IF DMC_G1.Control_ON THEN
        //Ustawienie wartosci PV
        DMC_G1.PV := INT_TO_REAL(T_1)/100.0;

        // iloczyn wektorow wspolczynnikow Ku
        // i przeszlych zmian sterowania delta_u_past
        // u(k|k) = u(k-1) + Ke*e(k) - Ku*deltaUp(k)
        DMC_G1.tmp := 0.0;
        FOR n_ := 0 TO DMC_G1.D_-1 BY 1 DO
                DMC_G1.tmp := DMC_G1.tmp
                        + DMC_G1.Ku[n_]*DMC_G1.delta_u_past[n_];
        END_FOR;

        // wyznaczenie nowej zmiany sterowania
        DMC_G1.delta_u := DMC_G1.Ke*(DMC_G1.SV - DMC_G1.PV)
                - DMC_G1.tmp;
        // wyznaczenie nowej wartosci sterowania
        DMC_G1.tmp := DMC_G1.MV + DMC_G1.delta_u;
        // nalozenie ograniczen na sterowanie
        IF(DMC_G1.tmp >  100.0) THEN
                DMC_G1.tmp :=  100.0;
        END_IF;
        IF(DMC_G1.tmp < 0.0) THEN
                DMC_G1.tmp := 0.0;
        END_IF;

        // przekazanie do regulatora osiagnietej zmiany sterowania
        DMC_G1.delta_u := DMC_G1.tmp - DMC_G1.MV;
        DMC_G1.MV := DMC_G1.tmp;
        // przesuniecie wektora przeszlych zmian sterowania
        // o jeden krok w tyl i wstawienie biezacej
        // zmiany sterowania na poczatek
        FOR n_ := DMC_G1.D_-2 TO 0 BY -1 DO
                DMC_G1.delta_u_past[n_+1] := DMC_G1.delta_u_past[n_];
        END_FOR;
        DMC_G1.delta_u_past[0] := DMC_G1.delta_u;

        G_1 := REAL_TO_INT(DMC_G1.MV*10.0);
END_IF;
```

Listing 2.11. "Program ST implementujący algorytm DMC toru G2-T3"

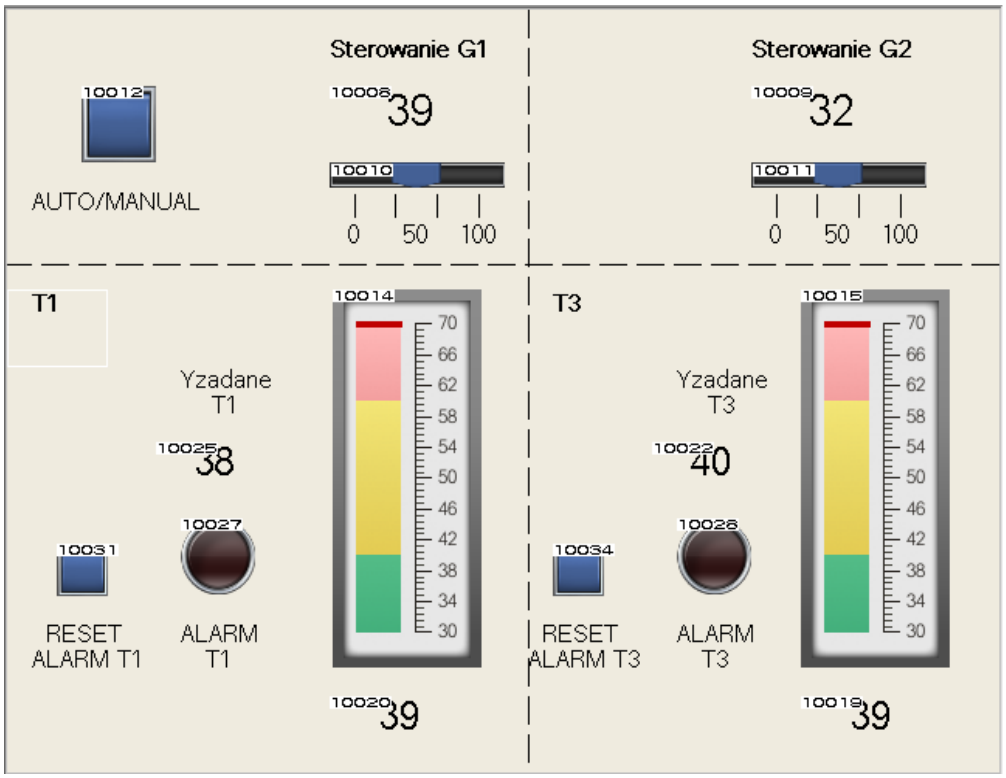
```
IF DMC_G2.Control_ON THEN
        //Ustawienie wartosci PV
        DMC_G2.PV := INT_TO_REAL(T_2)/100.0;

        // iloczyn wektorow wspolczynnikow Ku
        // i przeszlych zmian sterowania delta_u_past
        // u(k|k) = u(k-1) + Ke*e(k) - Ku*deltaUp(k)
        DMC_G2.tmp := 0.0;
        FOR n_ := 0 TO DMC_G2.D_-1 BY 1 DO
                DMC_G2.tmp := DMC_G2.tmp
                        + DMC_G2.Ku[n_]*DMC_G2.delta_u_past[n_];
        END_FOR;

        // wyznaczenie nowej zmiany sterowania
        DMC_G2.delta_u := DMC_G2.Ke*(DMC_G2.SV - DMC_G2.PV)
                - DMC_G2.tmp;
        // wyznaczenie nowej wartosci sterowania
        DMC_G2.tmp := DMC_G2.MV + DMC_G2.delta_u;
        // nalozenie ograniczen na sterowanie
        IF(DMC_G2.tmp >  100.0) THEN
                DMC_G2.tmp :=  100.0;
        END_IF;
        IF(DMC_G2.tmp < 0.0) THEN
                DMC_G2.tmp := 0.0;
        END_IF;

        // przekazanie do regulatora osiagnietej zmiany sterowania
        DMC_G2.delta_u := DMC_G2.tmp - DMC_G2.MV;
        DMC_G2.MV := DMC_G2.tmp;
        // przesuniecie wektora przeszlych zmian sterowania
        // o jeden krok w tyl i wstawienie biezacej
        // zmiany sterowania na poczatek
        FOR n_ := DMC_G2.D_-2 TO 0 BY -1 DO
                DMC_G2.delta_u_past[n_+1] := DMC_G2.delta_u_past[n_];
        END_FOR;
        DMC_G2.delta_u_past[0] := DMC_G2.delta_u;

        G_2 := REAL_TO_INT(DMC_G2.MV*10.0);
END_IF;
```

Odpowiedzi skokowe

Dobór parametrów regulatora

2.1.5. Panel operatora

Panel operatora Wartości mierzone,zadane oraz sterowanie



Rys. 2.1. Graficzny interfejs operatora stanowiska grzejąco-chłodzącego

2.1.6. Automat stanów

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

Listing 2.12. "Automat stanów modyfikujący zadane wartości temperatur T1 i T3"

```
TIM_MAIN(PT:= T#200s);

CASE  Stan_MAIN  OF
        0:
                IF NOT TIM_MAIN.Q THEN
                        TIM_MAIN.IN := 1;
                        Stan_MAIN := 1;
                END_IF;
        1:
                IF TIM_MAIN.Q THEN
                        T_1_zad := 4000;
                        T_3_zad := 4200;
                        TIM_MAIN.IN := 0;
                        Stan_MAIN := 2;
                END_IF;
        2:
                IF NOT TIM_MAIN.Q THEN
                        TIM_MAIN.IN := 1;
                        Stan_MAIN := 3;
                END_IF;
        3:
                IF TIM_MAIN.Q THEN
                        T_1_zad := 5500;
                        T_3_zad := 4200;
                        TIM_MAIN.IN := 0;
                        Stan_MAIN := 4;
                END_IF;
        4:
                IF NOT TIM_MAIN.Q THEN
                        TIM_MAIN.IN := 1;
                        Stan_MAIN := 5;
                END_IF;
        5:
                IF TIM_MAIN.Q THEN
                        T_1_zad := 4000;
                        T_3_zad := 4200;
                        TIM_MAIN.IN := 0;
                        Stan_MAIN := 6;
                END_IF;
        6:
                IF NOT TIM_MAIN.Q THEN
                        TIM_MAIN.IN := 1;
                        Stan_MAIN := 7;
                END_IF;
        7:
                IF TIM_MAIN.Q THEN
                        T_1_zad := 4000;
                        T_3_zad := 5700;
                        TIM_MAIN.IN := 0;
                        Stan_MAIN := 0;
                END_IF;
END_CASE;
```

Listing 2.13. "Przypisanie zadanych wartości temperatur do regulatorów"

```
// przypisanie wartosci zadanych
DMC_G1.SV := INT_TO_REAL(T_1_zad)/100.0;
DMC_G2.SV := INT_TO_REAL(T_3_zad)/100.0;
PID_G1.SV := INT_TO_REAL(T_1_zad)/100.0;
PID_G2.SV := INT_TO_REAL(T_3_zad)/100.0;

//G_1 := T_1_zad;
//G_2 := T_3_zad;
```

2.2. Stanowisko INTECO - zbiorniki wodne

Stanowisko INTECO nie zostało opracowane przez zespół w trakcie laboratorium. Poniższe opisy i programy są propozycją realizacji poleceń opracowaną na podstawie wiedzy uzyskanej w trakcie opracowywania poprzednich laboratoriów.

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

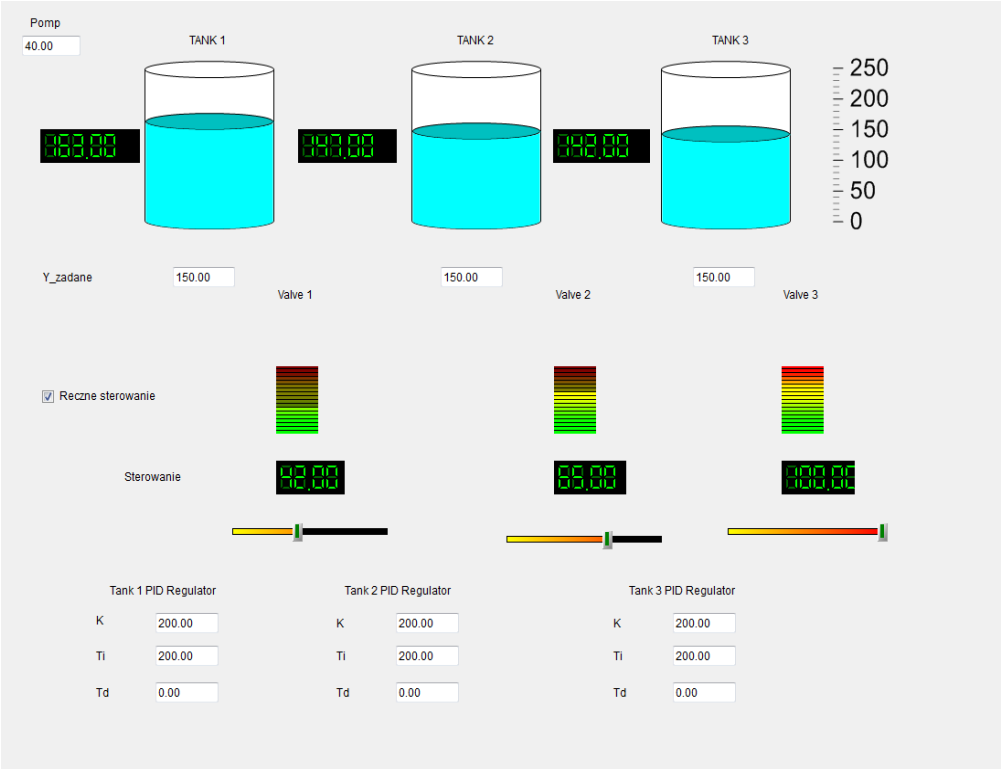
Dostosowac implementacje regulatora PID (wielopetlowego) do współpracy ze stanowiskiem Inteco. Regulator(y) dostroic. Omówic proces dobierania nastaw regulatorów. Uwzglednic ograniczenia jesli istnieja. Zamiescic wykresy w sprawozdaniu.

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.



Rys. 2.2. Graficzny interfejs operatora stanowiska Inteco

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.