Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania (projekt grupowy)

Sprawozdanie z projektu i ćwiczenia laboratoryjnego nr 3, zadanie nr 4

Piotr Chachuła, Cezary Dudkiewicz, Piotr Roszkowski

Warszawa, 2019

Spis treści

Część I

Projekt

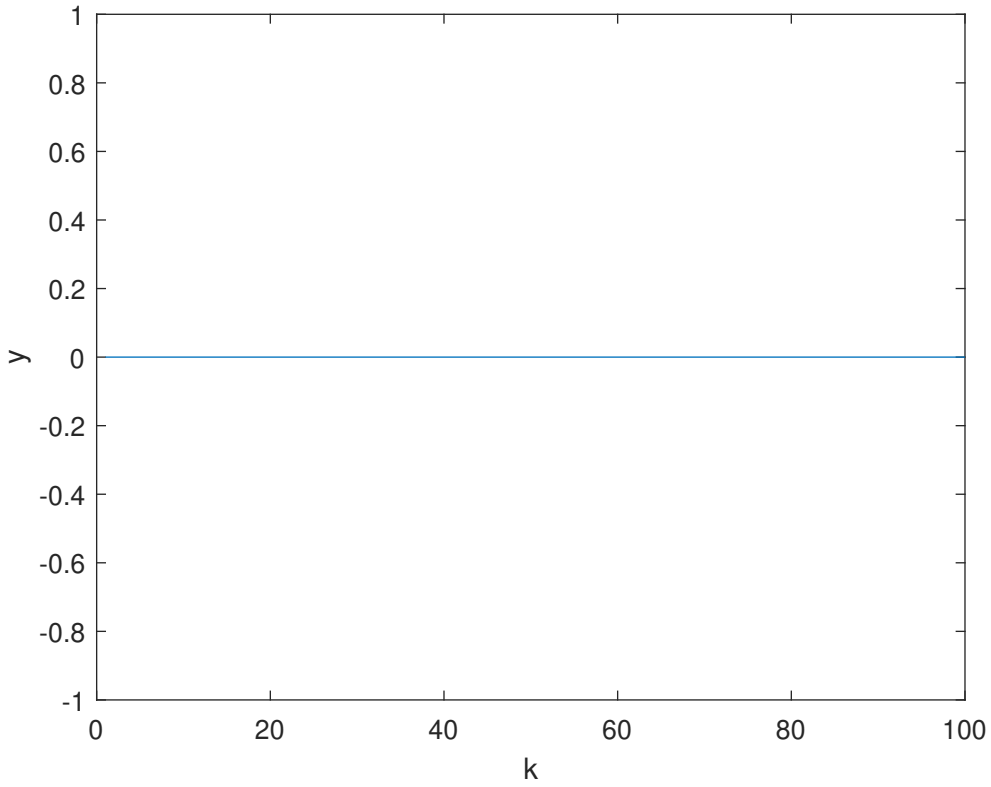
1. Weryfikacja punktu pracy

1.1. Opis postępowania

W celu sprawdzenia poprawności punktu pracy pobudzono obiekt sterowaniem o wartości $u = 0{,}0$ i sprawdzeniu czy stabilizuje się on w punkcjie pracy $y = 0{,}0$. Do symulacji wyjscia obiektu użyto udostępnionej funkcji `symulacja_obiektu4y`. Do testów napisano skrypt `Zad1.m`. Wyniki przedstawiono poniżej.

1.2. Wyniki

Zgodnie z przewidywaniami wyjscie obiektu ustaliło się na wartości $y = 0{,}0$. Punkt pracy ustalony jest więc poprawnie.



Rys. 1.1. Odpowiedź obiektu na sterowania $u = 0{,}0$

Część II

Laboratoria

2. Pomiar w punkcie pracy

2.1. Komunikacja z obiektem

Komunikacja z obiektem odobywa się za pomocą funkcji napisanych w środowisku MatLab. Najprostszy program użyty w tym projekcie, który posłużył także do późniejszego pobierania odpowiedzi skokowych znajduje się poniżej (jest to fragment skryptu `L3_1.m`).

```
addpath('F:\SerialCommunication'); % add a path to the functions
initSerialControl COM3 % initialise com port

N = 420;
step_response = zeros(N+1,1);
i = 0;

while(i<=N)
    %% obtaining measurements
    measurements = readMeasurements(1:7); % read measurements from 1 to 1

    %% processing of the measurements and new control values calculation
    disp([measurements(1),i]);
    step_response(i+1)=measurements(1);

    %% sending new values of control signals
    sendNonlinearControls(29)  % new corresponding control valuesdisp(measurements); % process measurements

    i=i+1;
    step_response(i)=measurements(1);

    waitForNewIteration(); % wait for new batch of measurements to be ready

end
```

W tym zadaniu używamy funkcji `sendNonlinearControls`, który wysyła sterowanie w sposób symulujący nieliniowość obiektu. Napisany skrypt działał poprawnie, pozwala na sterowanie sygnałami G1, W1 oraz pomiar T1.

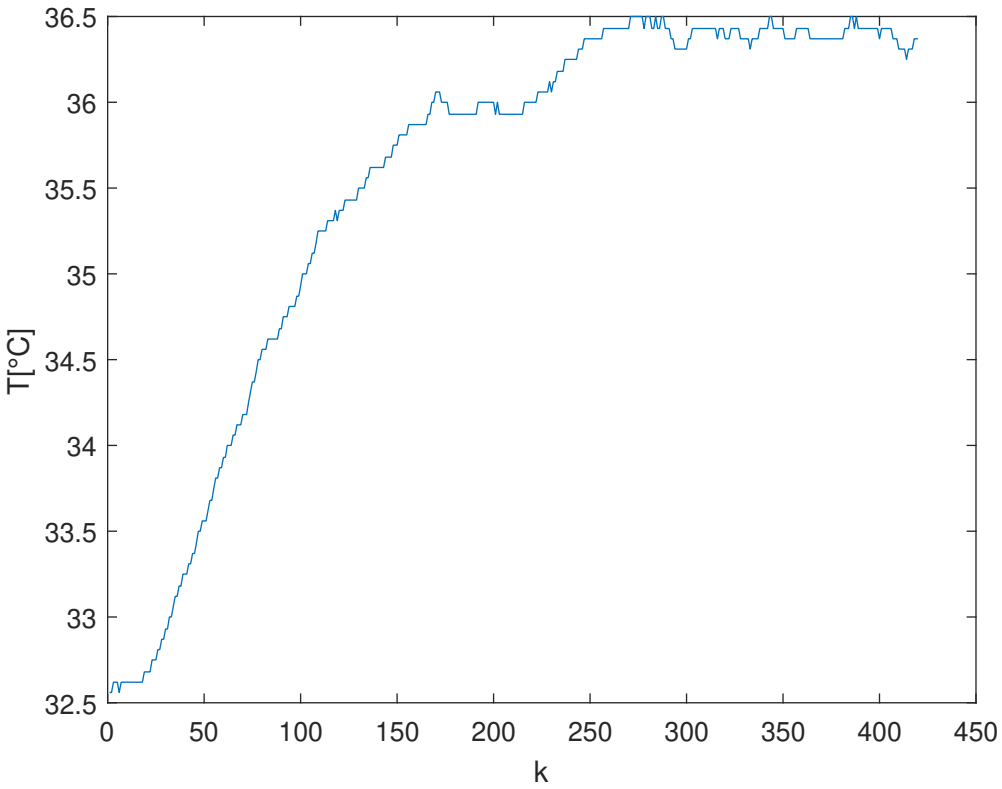
2.2. Punkt pracy

Doprowadzono obiekt do punktu pracy, tj. ustawiono wartości sygnałów W1 na 50, G1 na 29 i poczekano na ustabilizowanie obiektu (ponieważ obiekt jest rzeczywisty wahania temperatury są nieuniknione, zwłaszcza biorąc pod uwagę fakt lokalizacji stanowiska nr 4 w miejscu obok którego przechodzi dużo osób - wszelkie pomiary teraźniejsze oraz późniejsze mogą być zaburzone właśnie przez to). Wartość temperatury w punkcie pracy wynosi T1=35,43°C.

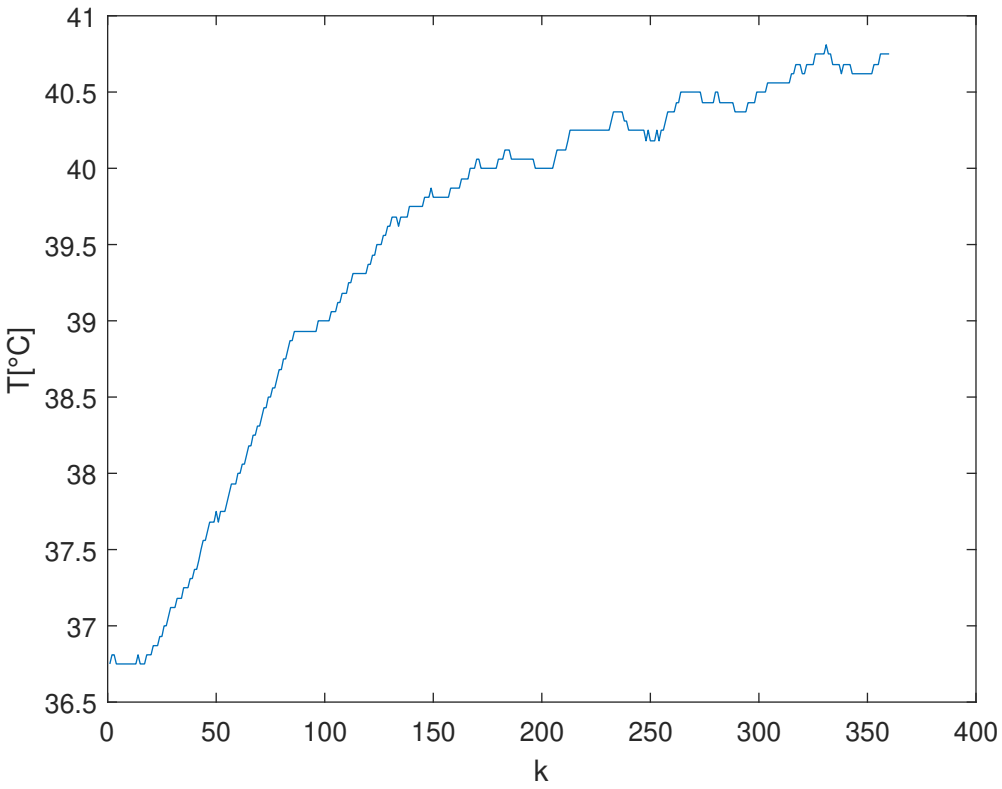
3. Charakterystyka obiektu

3.1. Inne punkty pracy

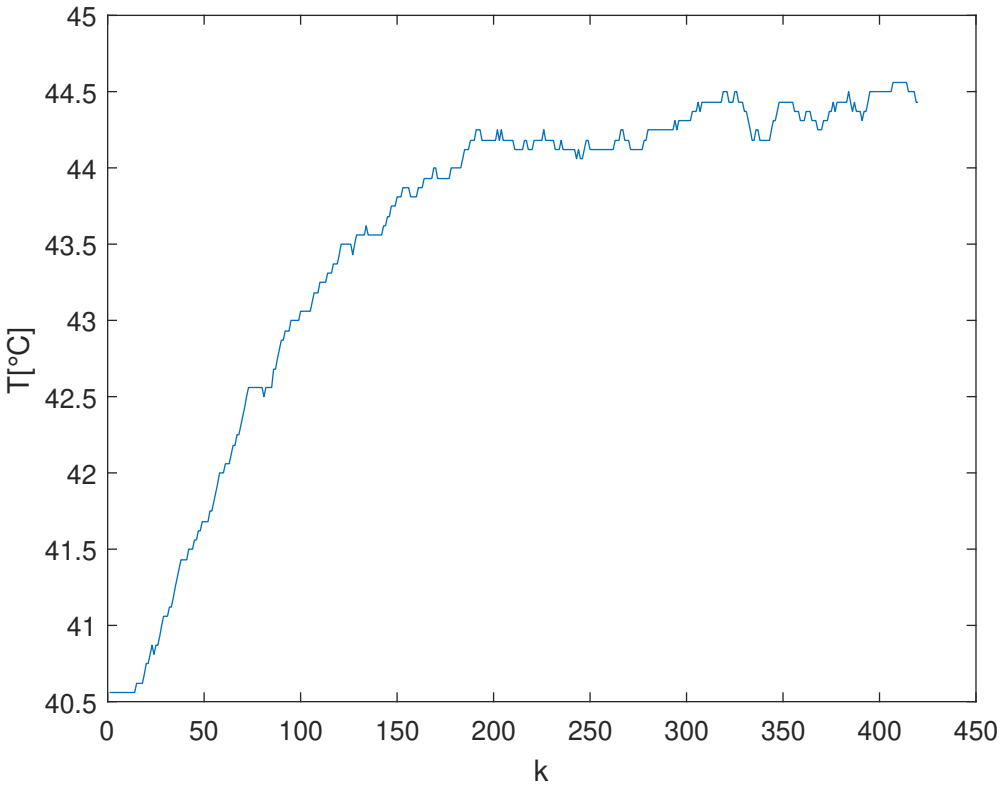
W celu pobrania wartości wyjścia dla innych punktów sterowania postępowano następująco: najpierw pobudzono układ sterowaniem równym G1=20 i poczekano na jego stabilizację. Następnie dokonywano skoków tej wartości sterowania o 10, aż do wartości G1=80. Przebieg eksperymentu ilustrują poniższe wykresy (skoki sterowania następowały w chwilii t=0:



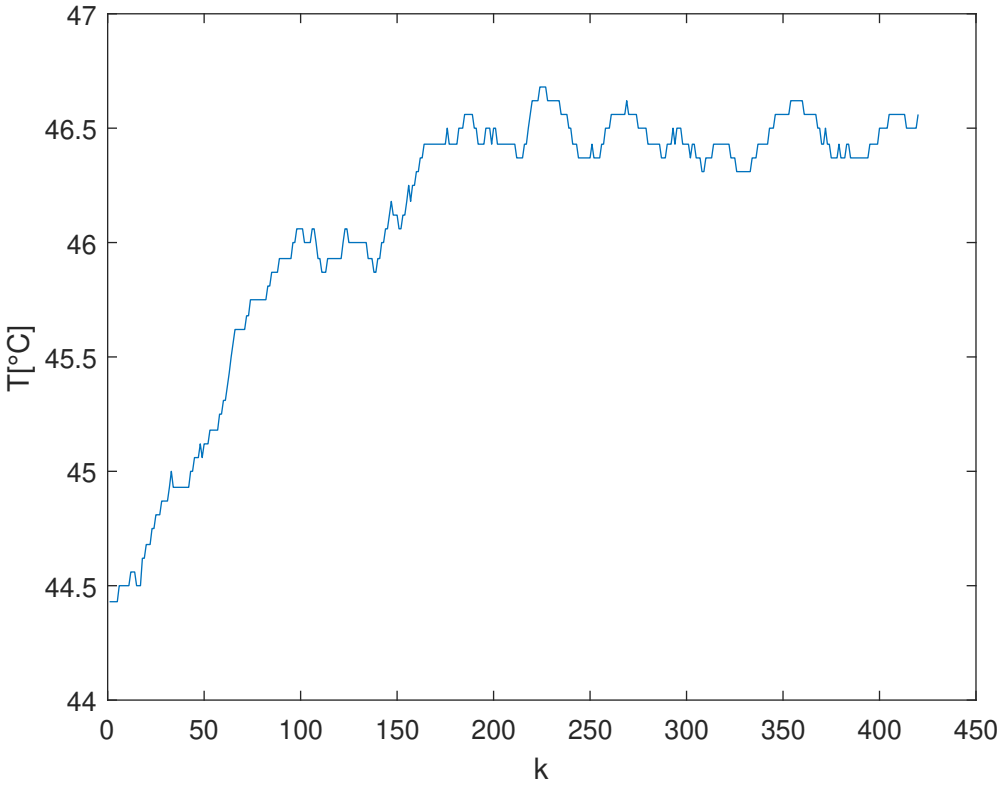
Rys. 3.1. Skok wartości sterowania z 20 do 30



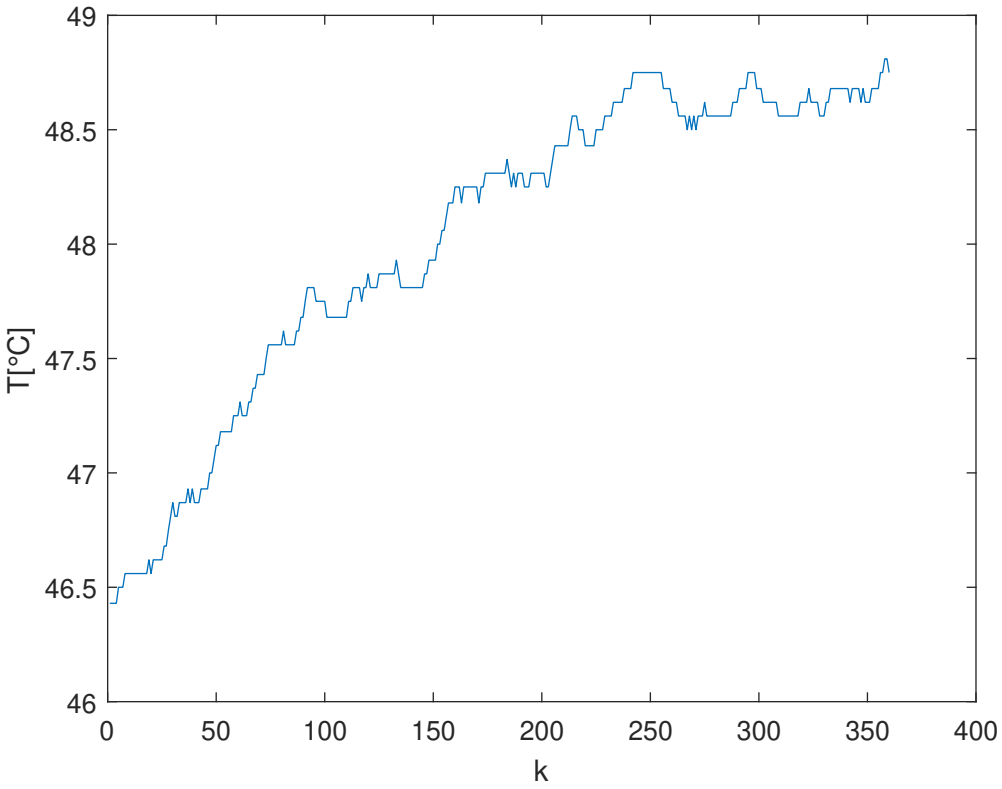
Rys. 3.2. Skok wartości sterowania z 30 do 40



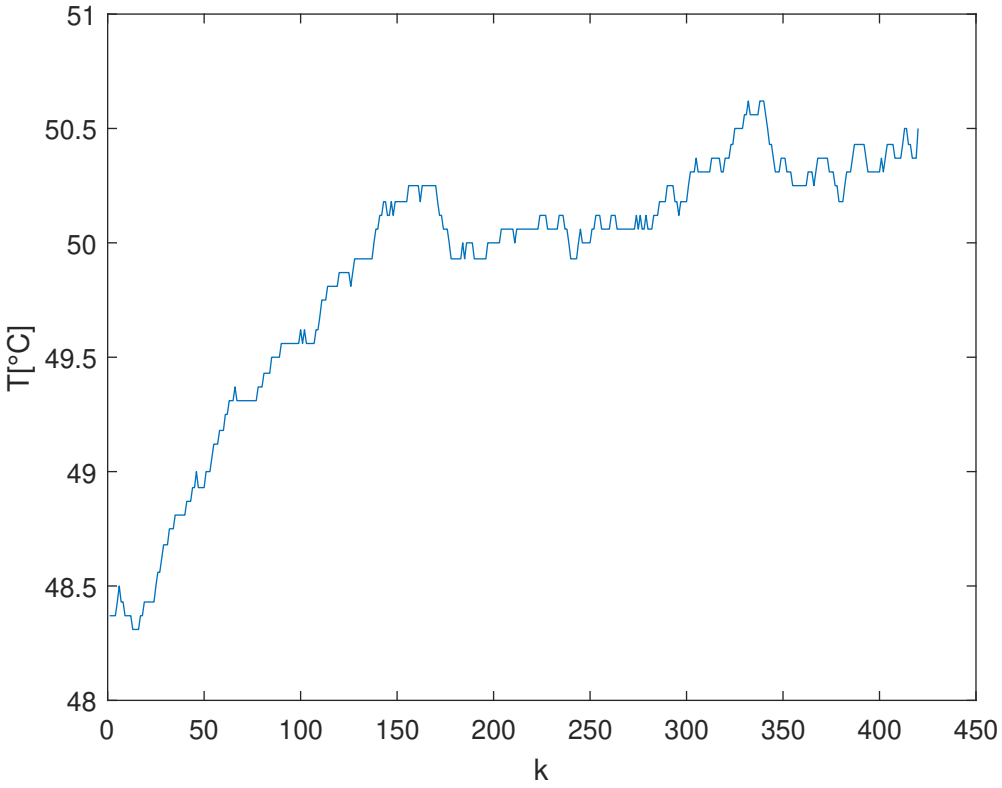
Rys. 3.3. Skok wartości sterowania z 40 do 50



Rys. 3.4. Skok wartości sterowania z 50 do 60



Rys. 3.5. Skok wartości sterowania z 60 do 70



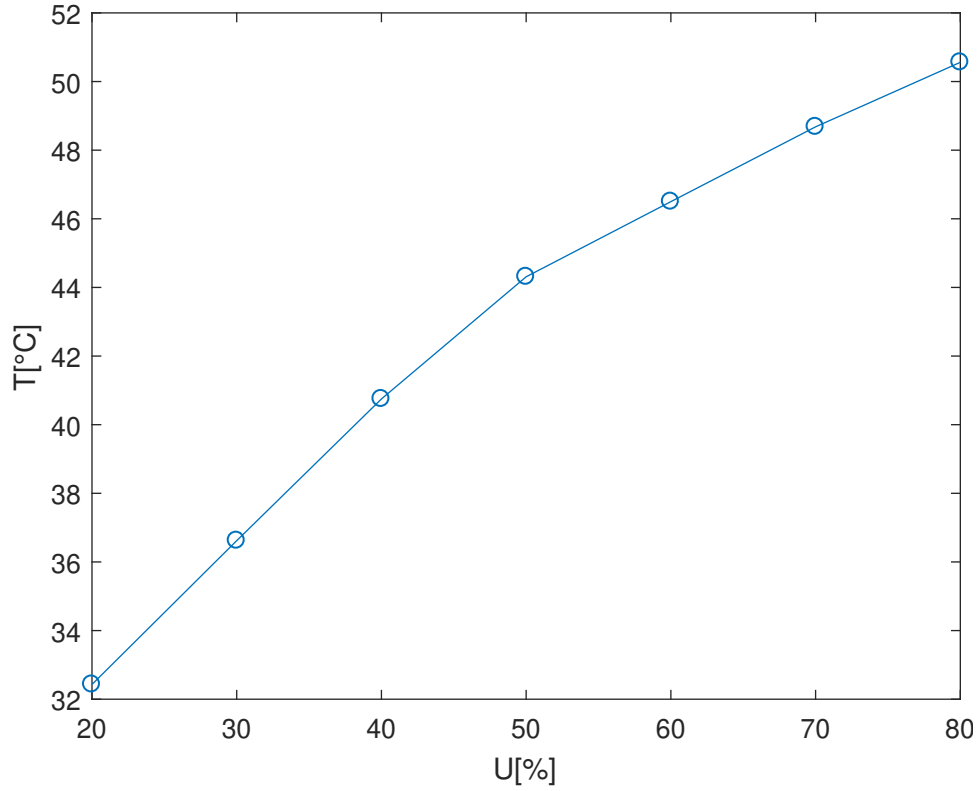
Rys. 3.6. Skok wartości sterowania z 70 do 80

Wyniki przedstawiono w tabeli:

G1[%]	T[°C]
20	32,43
30	36,62
40	40,75
50	44,31
60	46,5
70	48,68
80	50,56

3.2. Charakterystyka statyczna obiektu

Charakterystykę statyczną obiektu w przedziale sterowań G1 od 20 do 80% przedstawiono na wykresie 3.7:



Rys. 3.7. Charakterystyka statyczna obiektu

3.3. Wzmocnienie statyczne

Z wykresu charakterystyki liniowej można stwierdzić, że obiekt nie jest całkowicie liniowy, występuje załamanie charakterystyki w punkcie U=50%. Jednak obiekt jest kawałkami liniowy, przejawia właściwości liniowe w przedziale 20-50% oraz inne właściwości liniowe w przedziale 50-80% - na tych odcinkach obiekt zachowuje się praktycznie w sposób liniowy. Dlatego też możemy policzyć wzmocnienie statyczne obiektu w tych przedziałach sterowania:

$$K_{20-50\%} = \frac{Y(50) - Y(20)}{50 - 20} = \frac{44,31 - 32,43}{50 - 20} = 0,396 \tag{3.1}$$

$$K_{50-80\%} = \frac{Y(80) - Y(50)}{80 - 50} = \frac{50,56 - 44,31}{80 - 50} \approx 0,208 \tag{3.2}$$

4. Zastosowanie tradycyjnych regulatorów PID i DMC

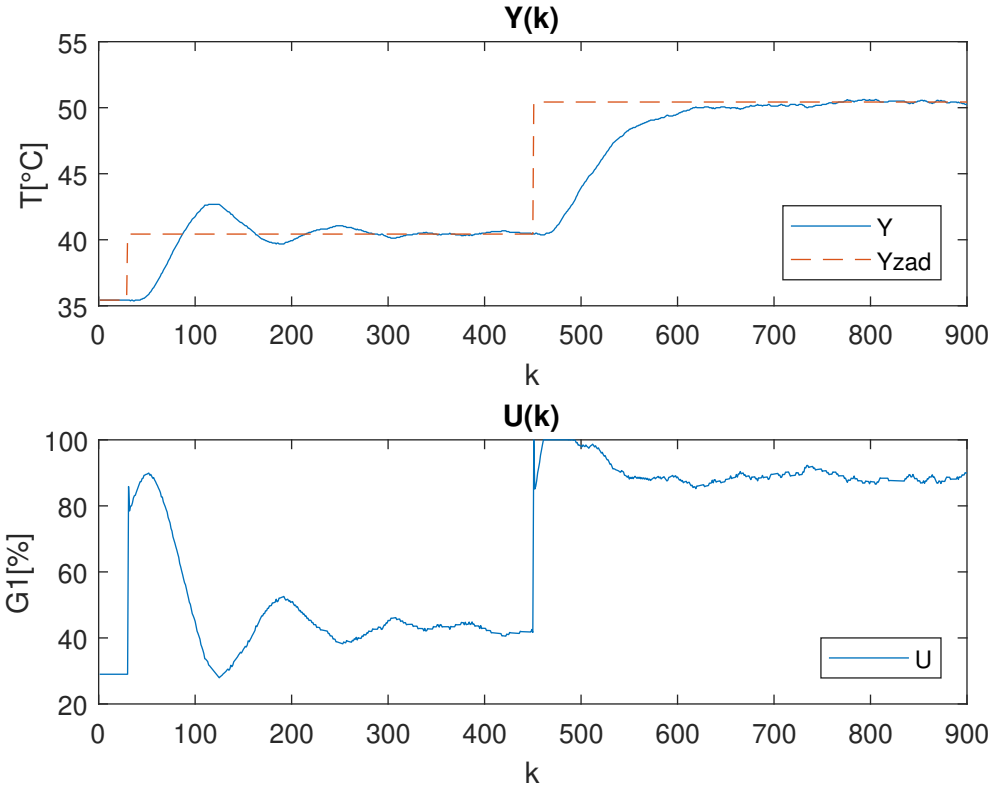
4.1. Regulator PID

4.1.1. Postępowanie

Zarówno w przypadku regulatora PID oraz DMC zostaną użyte skrypty (`doFuzzyPID.m` `doFuzzyDMC.m`) z odpowiednio ustawionym parametrem odpowiadającym za typ regulatora. Jedyna zmiana względem projektu nr 1 nastąpi w wysyłaniu sterowania do obiektu, gdzie funkcję `sendControls` zastępujemy funkcją `sendNonlinearControls`, która ma na celu symulację braku liniowości obiektu na całym obszarze wartości sterowań.

4.1.2. Wyniki symulacji

Wyniki symulacji przestawiono na wykresie 4.1:

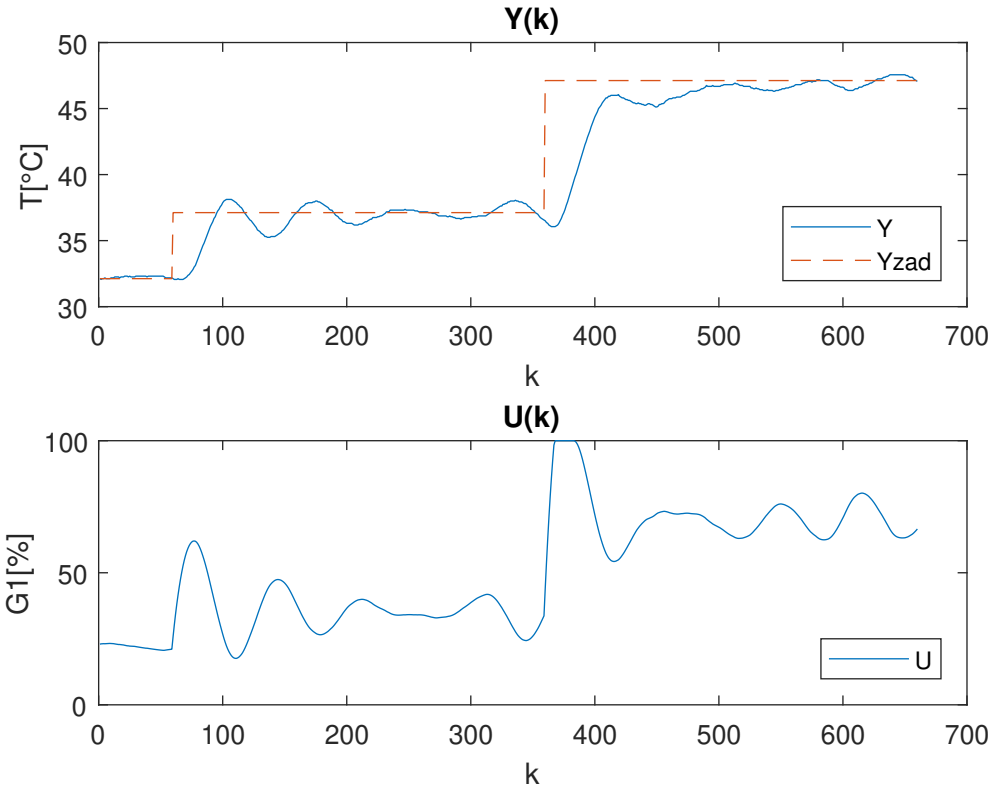


Rys. 4.1. Symulacja dla pojedynczego regulatora PID o parametrach $K = 9,65, T_i = 60, T_d = 0,17$

Błąd (suma kwadratów odchyłek) wyniósł E= 6077.

4.2. Regulator DMC

Wyniki symulacji przedstawiono na wykresie 4.2:



Rys. 4.2. Symulacja dla pojedynczego regulatora DMC o parametrach $D = 360, N = 120, N_u = 20, \lambda = 1$

Błąd (suma kwadratów odchyłek) wyniósł E= 3795 (wyniki można ze sobą porównywać mimo krótszego czasu trawnia symulacji, liczba skoków pozostała ta sama - tam są generowane głównie uchyby).

4.3. Wnioski

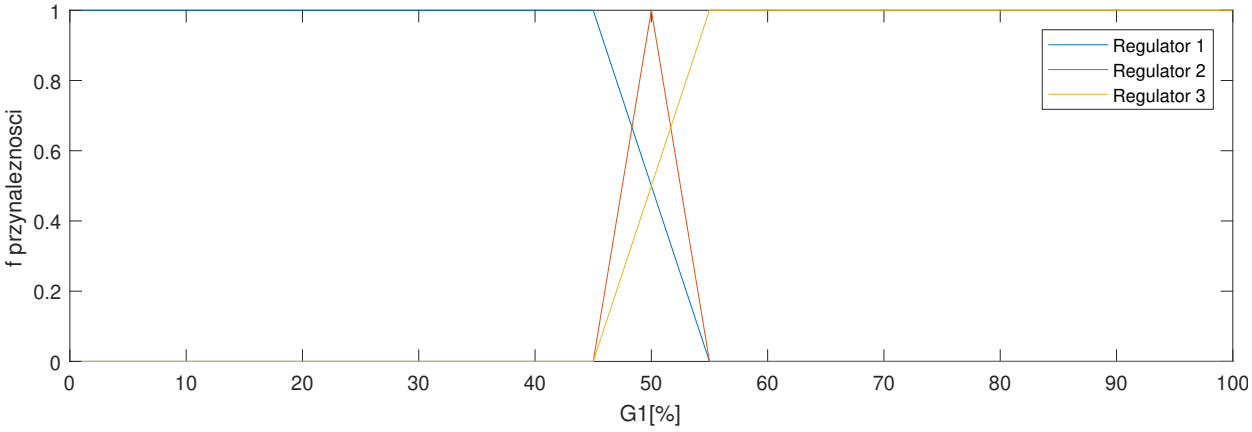
Pomimo że regulatory działają, temperatura zadana jest mniej więcej osiągana, jednak ich regulacja jest dosyć wolna, a wyjscie oscyluje. W celu poprawy regulacji dokonano rozmycia tych regulatorów.

5. Rozmywanie regulatorów

5.1. Funkcje aktywacji

Dla obu regulatorów dobrano identyczne funkcje aktywacjii dla trzech regulatorów lokalnych. Rozmywane one są względem wartości sterowana w poprzednim momencie. Zdecydowano się na to z dwóch powodów: po pierwsze układ może zmieniać swoje punkty pracy w zależności od warunków atmosferycznych otoczenia, rozmywanie po wyjściu naraża nas na wpływ takich odchyleń i spadek jakości regulacji; po drugie nieliniowość na obiekcie jest wprowadzana sterowaniem, tzn. obiekt (według tego co ustalono na projektach 1 oraz 2) jest w miarę liniowy, a jego właściwości nie mogły się zmienić, nieliniowość narzucana jest przez funkcję `sendNonlinearControls`, która jakoby przerabia sterowanie tak, aby układ zachowywał się jak nieliniowy.

Pierwszy regulator lokalny będzie aktywny głównie w przedziale 0-50% (na tym przedziale obiekt zachowuje się jak liniowy), drugi w przedziale 45-55%, a trzeci głównie w przedziale 50-100%, patrz rysunek 5.1. Na tym etapie warto wspomnieć, iż regulator nr 2 umieszczany jest tylko w celu zaspokojenia potrzeb polecenia, sama charakterystyka wskazuje na użycie jedynie dwóch regulatorów lokalnych. Wszelkie próby dostajania regulatorów rozmytych mogą kończyć się fiaskiem z uwagi właśnie na ten sztucznie wytworzony regulator lokalny (zwłaszcza w przypadku regulatora DMC).



Rys. 5.1. Funkcje aktywacji regulatorów lokalnych

5.2. Rozmyty regulator PID

5.2.1. Algorytm

Algorytm działa bardzo podobnie do tego z projektu pierwszego, tylko że każde wyjście trzech regulatorów lokalnych jest ważone przez odpowiadającą mu wartość funkcji aktywacji. Poglądowy algorytm programu porównawczego obsługującego zarówno rozmytą jak i nierozmytą wersję algorytmu wraz z bardziej szczegółowym omówieniem w postaci komentarzy został umieszczony poniżej:

```
% Inicjalizacja polaczenia
addpath('F:\SerialCommunication'); % add a path to the functions
initSerialControl COM3 % initialise com port

% Typ regulatora
```

```
regulator=1;

% Inicjalizacja parametrow
K=[10 10 20];
Ti=[60 60 60];
Td=[0 0 0];
T=1;
% Inicjalizacja wskaznika jakosci
error=0;
% Inicjalizacja czasu trwania symluacji
sim_len=900;

% Parametry dyskretnego PIDa
for i=1:3
r0(i)=K(i)*(1+T/(2*Ti(i))+Td(i)/T);
r1(i)=K(i)*(T/(2*Ti(i))-2*Td(i)/T-1);
r2(i)=K(i)*Td(i)/T;
end

% Inicjalizacja
Y=zeros(sim_len,1);
U=zeros(sim_len,1);
e=zeros(sim_len,1);
y=zeros(sim_len,1);
u=zeros(sim_len,1);
Yzad=zeros(sim_len,1);
kk=linspace(1,sim_len,sim_len)';

% Zakladamy ze przed rozpoczeciem symulacji obiekt znajdowal sie w punktu pracy
Ypp=readMeasurements(1);
Upp=29;
Y(1:30)=Ypp;
U(1:30)=Upp;

% Inicjalizacja horyzontu wartosci zadanych
Yzad(1:sim_len/3-1)=Ypp+5;
Yzad(sim_len/3:2*sim_len/3-1)=Ypp+15;
Yzad(2*sim_len/3:sim_len)=Ypp;

% Ograniczenia U
Umin=0;
Umax=100;
umin=Umin-Upp;
umax=Umax-Upp;

% Glowna petla programu
for i=31:sim_len
% Odczyt wartosci temperatury
measurements = readMeasurements(1:7);
Y(k)=measurements(1)
% Rzutowanie wzgledem wartosci punktu pracy
y(k)=Y(k)-Ypp;
% Liczenie uchybu i uaktualnienie wskaznika jakosci
e(k)=Yzad(k)-Y(k);
error=error+e(k)^2;
% Wyliczenia wartosci funkcji aktywacji
w=f_przyn(U(k-1));

% Uzycie PIDa to wyliczenia strerowania
%Jesli regulator jest nierozmyty przyjmujemy pierwsze elementy macierzy jako parametry
if regulator==1
u_wyliczone=r2(1)*e(k-2)+r1(1)*e(k-1)+r(1)*e(k)+u(k-1);

else
    for q=1:3
        u_wyliczone=u_wyliczone+w(q)*(r2(q)*e(k-2)+r1(q)*e(k-1)+r0(q)*e(k)+u(k-1));
    end
end

% Rzutowanie ograniczen na wartosc sterowania
if u_wyliczone<umin
u_wyliczone=umin;
elseif u_wyliczone>umax
u_wyliczone=umax;
end
u(k)=u_wyliczone;
% Rzutowanie sterowania wzgledem punktu pracy
U(k)=u_wyliczone+Upp;
% Wyslanie sterowania do regulatora
sendNonlinearControls(U(k))
end
```

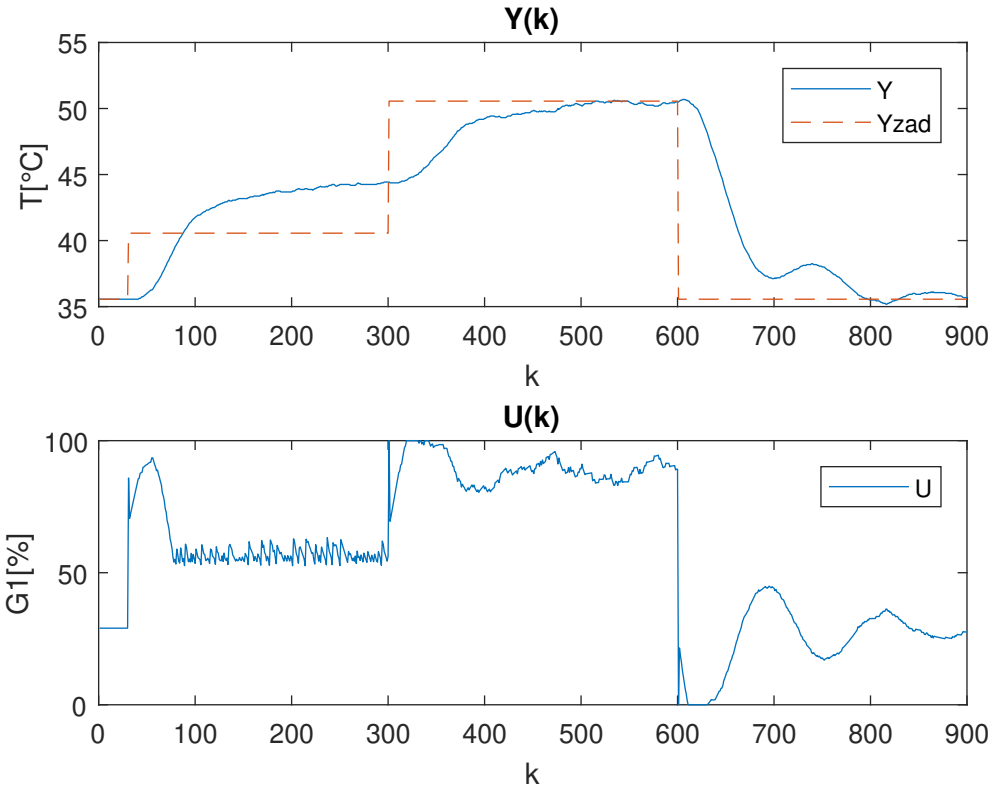
Listing 5.1. Implementacja regulatora PID

5.2.2. Przykładowe PIDy

W celu weryfikacji poprawności napisanego algorytmu przesymulowano obiekt dla parametrów bardzo zbliżonych do tych znalezionych w projekcie nr 1. Stanowiły one także dobry punkt wyjścia do dalszego strojenia. Jako, że przebieg był praktycznie analogiczny do tego znalezionego w zadaniu 3, nie zamieszczano go ponownie.

W celu poprawienia jakości regulacji, w pierwszym kroku zdecydowano się na zwiększenia wzmocnień regulatorów lokalnych 2 oraz 3, tam gdzie były największe problemy z szybkością regulacji.

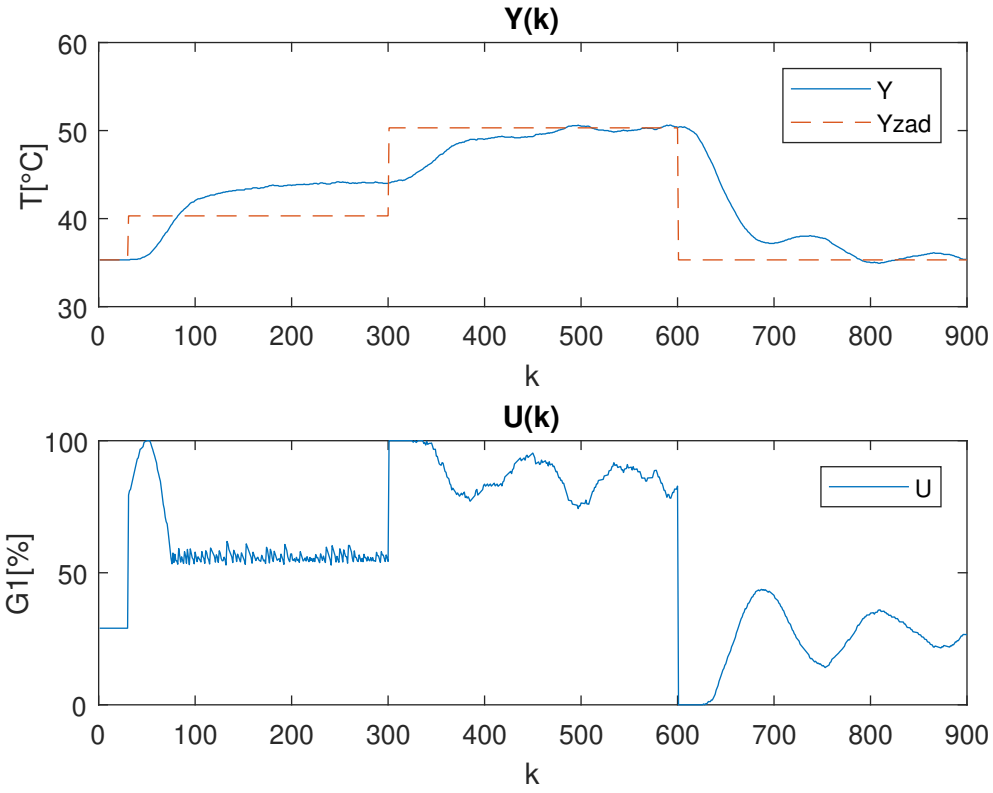
$K_1 = \mathbf{9{,}65}, T_{i1} = \mathbf{60}, T_{d1} = \mathbf{0{,}17}, K_2 = \mathbf{14}, T_{i2} = \mathbf{60}, T_{d2} = \mathbf{0{,}17}, K_3 = \mathbf{20}, T_{i3} = \mathbf{60}, T_{d3} = \mathbf{0{,}17}$



Rys. 5.2. Symulacja regulatora rozmytego nr 1

Widać, że regulator zachowuje się bardzo źle zwłaszcza w okolicach drugiego skoku wartości zadanej. Wynika to z faktu, iż punkt ten znajduje się blisko punktu przełączania regulatorów (sterowanie okołow wartości 50) i pozostaje w sferze aktywacji regulatora lokalnego nr 2. Wartość błędu całkowitego jest równa $E_c = 14\,397$, a dla ocinka porównywalnego z wcześniejszym (niekompletnym) eksperymentem dla regulatora nierozmytego E=4693,3. O ile sama wartość błędu zmalała, wynika to bardziej z faktu iż w przypadku skoku wartości zadanej o 15 stopni regulator już znajdował się bliżej tego punktu i naliczał mniejsze kary (które są kwadratem odchyłki). Regulacja nie poprawiła się. W następnym kroku zdecydowano się na zbliżenie parametrów regulatora lokalnego nr 2 do regulatora nr 1 oraz sprawdzenie jak regulator zachowuje się po całkowitym odłączeniu członu różniczkującego.

$K_1 = \mathbf{10}, T_{i1} = \mathbf{60}, T_{d1} = \mathbf{0}, K_2 = \mathbf{10}, T_{i2} = \mathbf{60}, T_{d2} = \mathbf{0}, K_3 = \mathbf{20}, T_{i3} = \mathbf{60}, T_{d3} = \mathbf{0}$



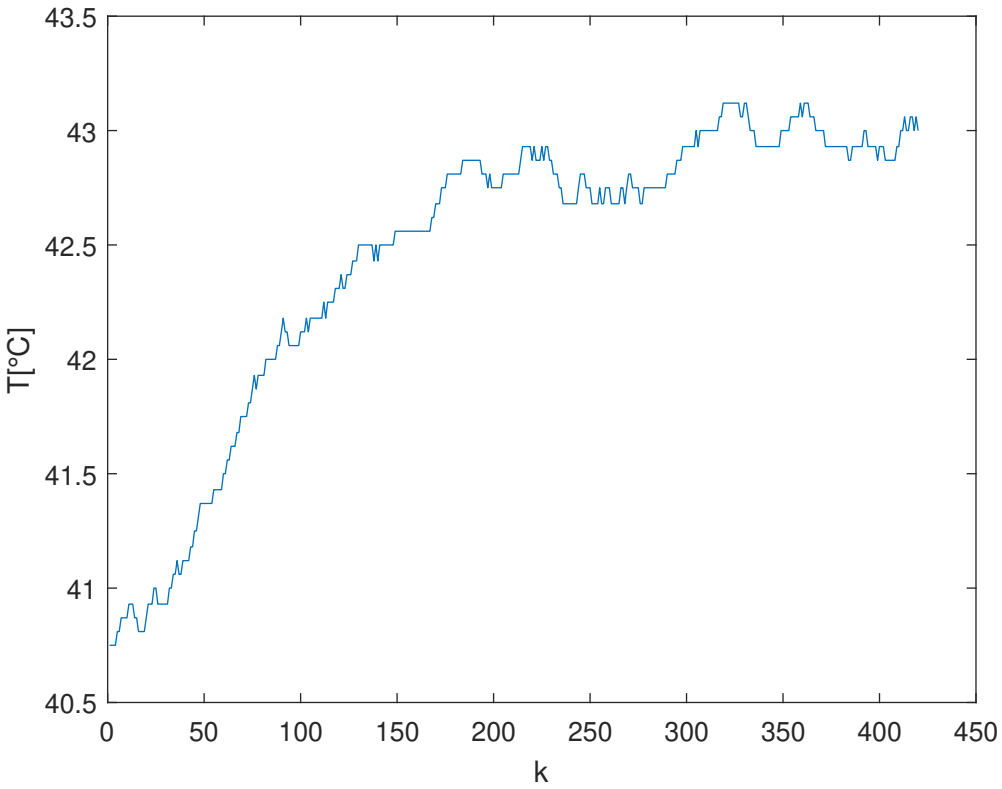
Rys. 5.3. Symulacja regulatora rozmytego nr 2

Praktycznie nie nastąpiła żadna poprawa w porównaniu z poprzednim regulatorem, zachowuje się on bardzo podobnie i ma podobne problemy. Wartość błędu całkowitego jest równa $E_c = 13\,866$, a dla ocinka porównywalnego z wcześniejszym (niekompletnym) eksperymentem dla regulatora nierozmytego E=4638,3, czyli są odrobinę mniejsze.

5.3. DMC

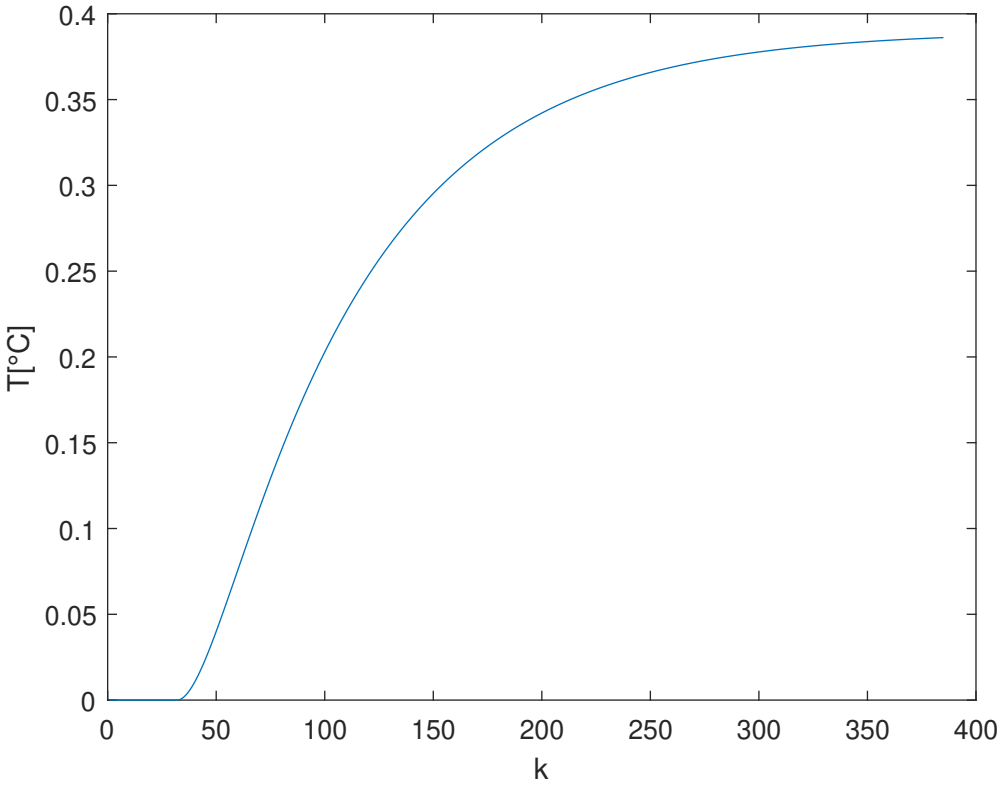
5.3.1. Odpowiedzi skokowe

Ponieważ w obszarze sterowań 20-50% układ zachowuje się w ten sam liniowy sposób, nie powinno być różnicy, z jakiego miejsca tego przedziału weźmiemy odpowiedz skokową, analogicznie dla przedziału sterowań 50-80%. Zdecydowano się na skoki 20->30% oraz 50->80%, które zdawały się mieć najlepsze własności. Do stworzenia zoptymalizowanych odpowiedzi skokowych można użyć przebiegów już wcześniej wyznaczonych. Dla regulatora lokalnego nr 2 wykonano skok sterowania 47,5->52,5%. Przedstawiono go na rysunku 5.4:

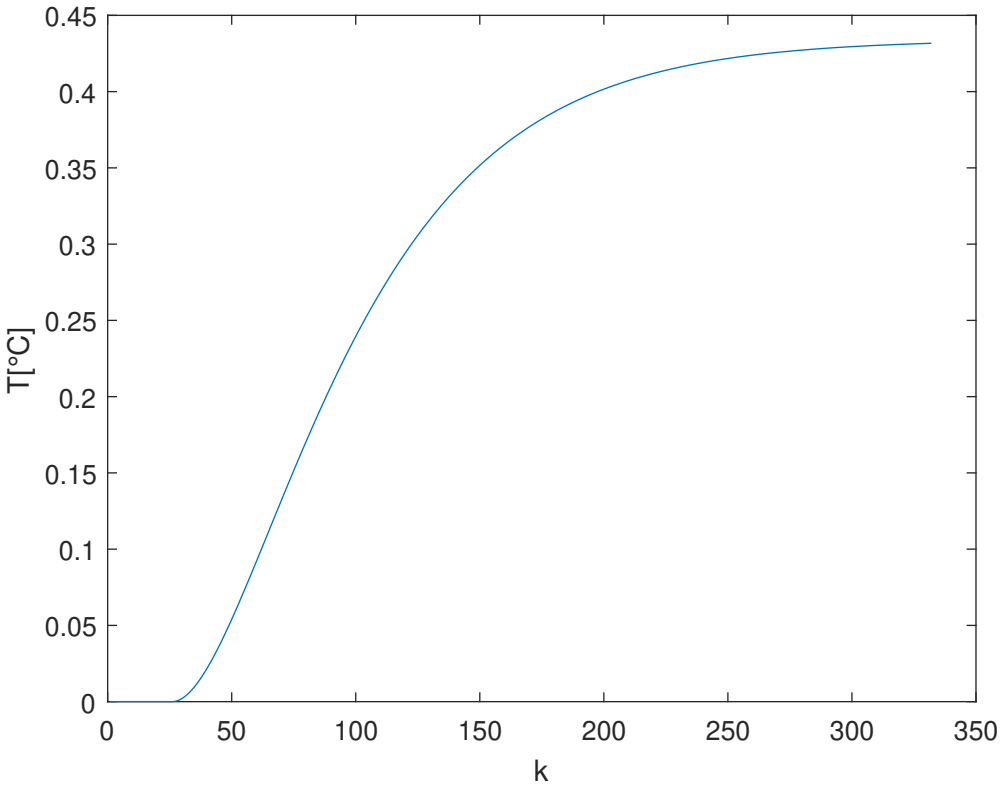


Rys. 5.4. Skok wartości sterowania z 47,5 do 52,5

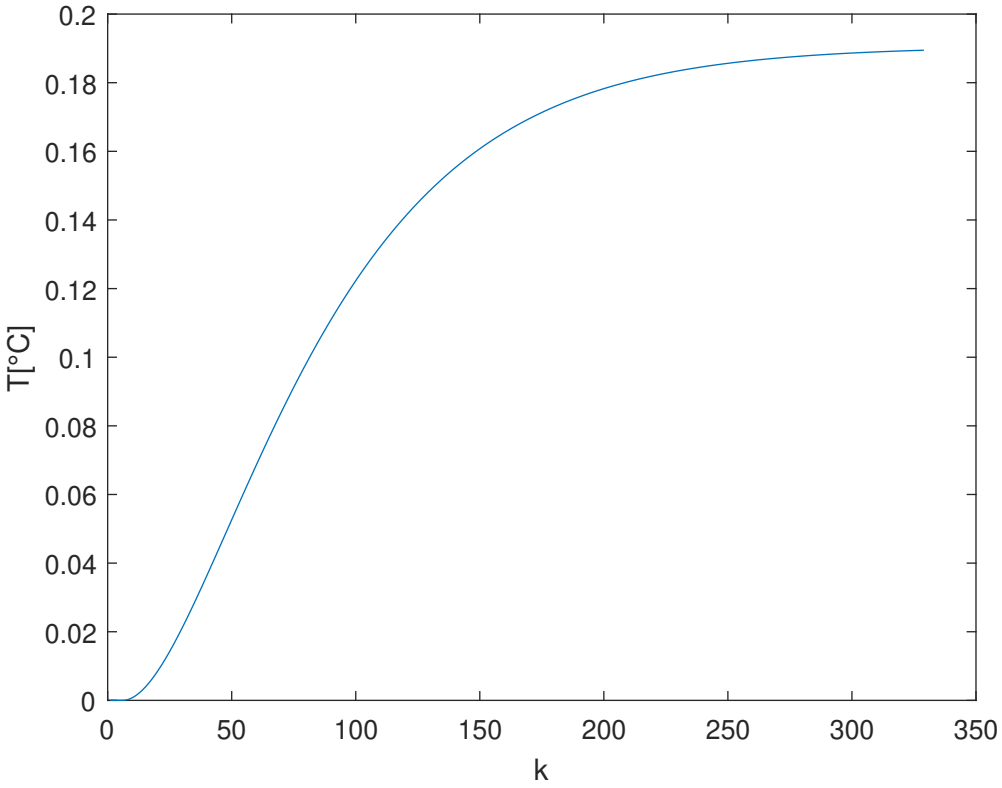
Następnie dokonano normalizacji wszystkich tych przebiegów w celu użycia ich w regulatorze DMC. W tym celu dokonano ich aproksymacji jako członów inercyjnych drugiego rzędu z opóźnieniem oraz dokonano ich przycięcia do wartości horyzontu dynamiki, który ustalono jako D=360. Przebiegi znormalizowanych odpowiedzi przedstawiono poniżej:



Rys. 5.5. Znormalizowany skok wartości sterowania z 20 do 30



Rys. 5.6. Znormalizowany skok wartości sterowania z 47,5 do 52,5



Rys. 5.7. Znormalizowany skok wartości sterowania z 70 do 80

5.3.2. Algorytm

```
% Zaladowanie odpowiedzi skokwej
load('step_response_70_80_approx');
load('step_response_20_30_approx');
load('step_response_central_approx');
% Inicjalizacja komunikacji z obiektem
addpath('F:\SerialCommunication');
initSerialControl COM3

% Ustalenie maksymalnych przedzialow sterowania
Umax=100;
Umin=0;

%Ustalenie iloscie regulatorow lokalnych
il = 3;

% Ustawienie parametrow regulatora
D = 320;
N=D;
Nu=D;
lambda = [100 200 100];

% Inicjalizacja macierzy przechowujacych zmienne
ku = zeros(il,D-1);
ke = zeros(1,il);
deltaup=zeros(1,D-1);
deltauk = zeros(il,1);
DELTAuk = zeros(il,1);
% Zaladowanie odpowiedzi skokwych
S(:,1)=step_response_20_30_approx(1:D);
S(:,2)=step_response_central_approx(1:D);
S(:,3)=step_response_70_80_approx(1:D);

% Stworzenie macierzy M, Mp oraz wyliczenie paratmetrow ku oraz ke dla
% kazdego regulatora lokalnego
for r = 1:il
    s = S(:,r);

    M=zeros(N,Nu);
    for i=1:N
       for j=1:Nu
          if (i>=j)
             M(i,j)=s(i-j+1);
          end
       end
    end

    MP=zeros(N,D-1);
    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

    I=eye(Nu);
    K=((M'*M+lambda(r)*I)^-1)*M';
    ku(r,:)=K(1,:)*MP;
    ke(r)=sum(K(1,:));
end


% Ustalenie punktu pracy
U0 = 29;
Y0 = readMeasurements(1);

% Ustalenie czasu symulacji
start = 10;
n=750;

% Inicjacja macierzy przechowujacej wyjscia, sterowanie oraz uchyby
U = U0*ones(1,n);
Y = Y0*ones(1,n);
e = zeros(1,n);

% Utworzenie horyzontu wartosci zadanej
Yz = Y;
Yz(1:10)=Y0;
Yz(11:n/3) = Y0+5;
Yz(n/3+1:2*n/3)=Y0+15;
```

```
Yz(2*n/3+1:end)=Y0;

% Petla glowna regulatora
for k = start:n
    % Odczyt wartosci wyjscia
    Y(k) = readMeasurements(1);
    % Policzenie wartosci uchybu
    e(k) = Yz(k) - Y(k);

    % Dla regulatora nierozmytego liczymy jedno sterowanie
    if regulator==1
        % Prawo regulacji
        DELTAuk(i)=ke(1)*e(k)-ku(1,:)*deltaup';
    else

    % Dla regulatora rozmytego liczymy sterowanie dla kazdego regulatora
    % lokalnego
    for i = 1:il
        deltauk(i) = ke(i)*e(k)-ku(i,:)*deltaup';
    end
    % Polczenie wartosci funkcji aktywacji dla kazdego regulatora lokalnego
    w = f_przyn(U(k-1));
    % Przemnozenie sterowan regulatorow lokalnych przez ich wartosci
    % funkcji aktywacji
    DELTAuk = w*deltauk/sum(w);
    end
    % Przesuniecie wektora dUp
    for i = D-1:-1:2
      deltaup(i) = deltaup(i-1);
    end
    % I wpisanie do niego wyliczonego elementu
    deltaup(1) = DELTAuk;
    % Wyliczenie obecnej wartosci sterowania
    U(k) = U(k-1)+deltaup(1);
    % Uwzglednienie ograniczen na wartosc sterowania
    U(k) = max(min(U(k),Umax),Umin);
    % Korekta zmiany wartosci sterowania
    deltaup(1)=U(k)-U(k-1);
    % Wyslanie sterowania do obiektu
    sendNonlinearControls(U(k))

    disp([Y(k),U(k)]);
    waitForNewIteration();
end
```

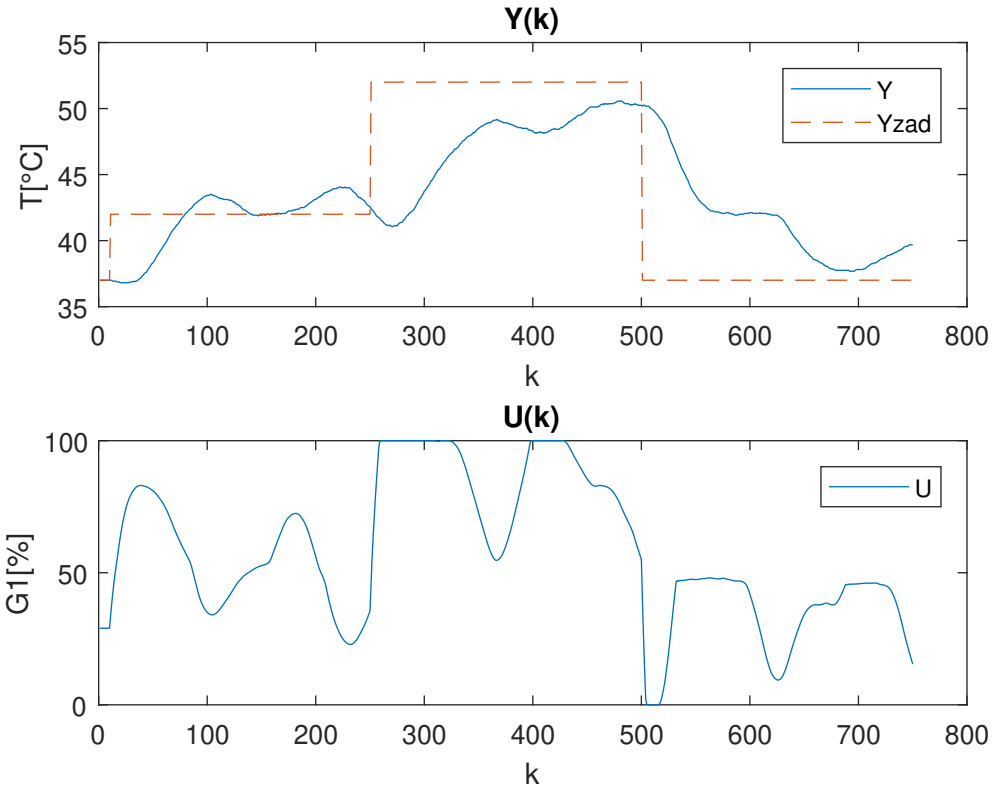
Listing 5.2. Implementacja regulatora DMC

5.3.3. Przykładowe DMC

Horyzonty zgodnie z poleceniem są równe, czyli $D = N = N_u = 360$, zmieniamy jedynie wielkości parametru λ.

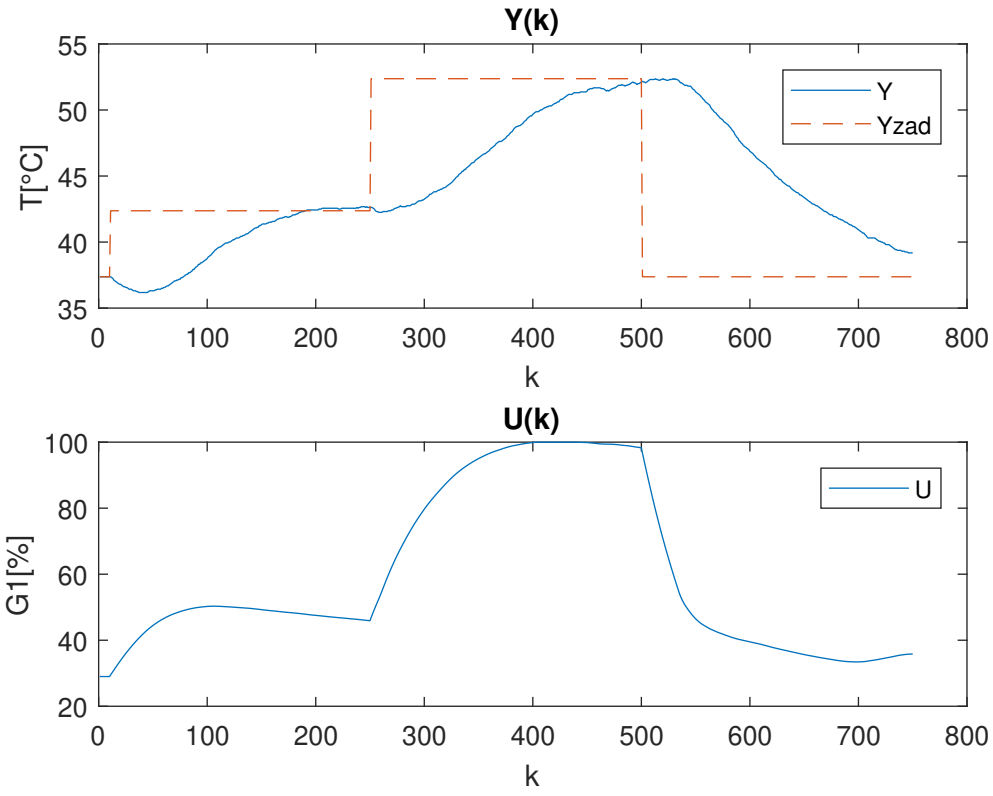
$\lambda_1 = 1, \lambda_2 = 1, \lambda_3 = 1$

Wartość błędu całkowitego jest równa $E_c = 17\,614$, a dla odcinka porównywalnego z wcześniejszym (niekompletnym) eksperymentem dla regulatora nierozmytego E=9240. Regulator działa źle. Sprawdzono jeszcze regulację dla zwiększonych parametrów λ. Czas laboratorium nie pozwolił na pozyskanie planowanej symulacji dla mniejszych wartości tego parametru.



Rys. 5.8. Rozmyty regulator nr 1

$\lambda_1 = 100, \lambda_2 = 200, \lambda_3 = 100$



Rys. 5.9. Rozmyty regulator nr 2

Wartość błędu całkowitego jest równa $E_c = 34\,133$, a dla odcinka porównywalnego z wcześniejszym (niekompletnym) eksperymentem dla regulatora nierozmytego E=11 814. Regulator działa jeszcze gorzej (jest wolniejszy).

5.4. Wnioski

Zadanie poprawy regulacji nie powiodło się - regulatry rozmyte, teoretycznie lepsze, nie przyniosły spodziewanego polepszenia jakości sterowania. Może to wynikać z:

— nieprawidłowego dobrania funkcji aktywacji poszczególnych regulatorów lokalnych, zwłaszcza drugiego, który wydaje się, że jest zbędny i tylko pogarsza regulację;
— w przypadku regulatora PID - niedostatkiem czasu - po przeprowadzoneniu większej ilości symulacji, możliwe, że regulator działałby dobrze;
— w przypadku regulatora DMC - przypadkowym użyciem nieaproksymowanych odpowiedzi skokowych (mimo posiadania ich zaproksymowanych wersji).