Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania (projekt grupowy)

Sprawozdanie z projektu i ćwiczenia laboratoryjnego nr 1, zadanie nr 15

Michał Pióro, Radosław Ślepowroński, Jan Szymczak

Warszawa, 2024

Spis treści

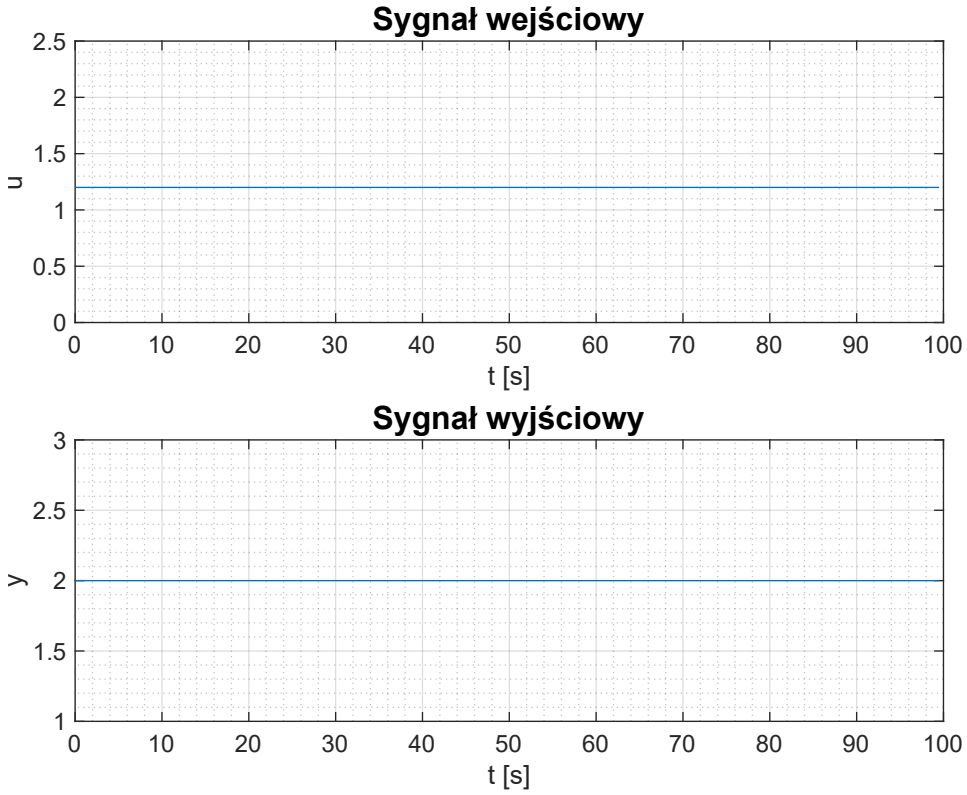
1. Projekt

1.1. Poprawność wartości U_{pp} i Y_{pp}

Wartości punktu pracy naszego obiektu wynoszą odpowiednio: $U_{pp} = 1.2$ oraz $Y_{pp} = 2$. Aby sprawdzić ich poprawność, przeprowadzono prostą symulację, w której od chwili początkowej k=1 obiekt znajdował się w podanym punkcie pracy. Następnie obliczono wyjście obiektu w następnych chwilach, do k = 200, zgodnie z kodem:

```
kk=200; % koniec symulacji
u(1:kk)=1.2; y(1:11)=2;

% główna pętla symulacyjna
for k=12:kk
    % symulacja obiektu
    y(k)=symulacja_obiektu15y_p1(u(k-10),u(k-11),y(k-1),y(k-2));
end
```

Tak prezentują się wyniki symulacji:



Rys. 1.1. Sprawdzenie poprawności punktu pracy

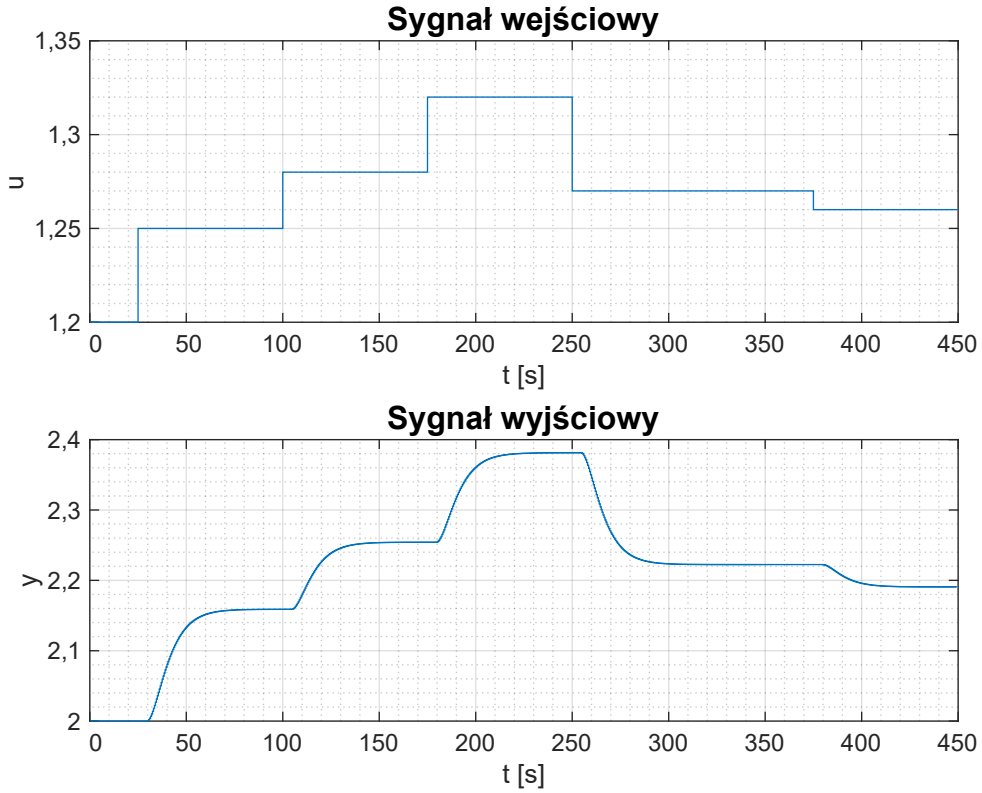
Jak widać, obiekt jest stabilny i podane wartości sygnałów faktycznie stanowią punkt pracy tego obiektu, ponieważ nie nastąpiły żadne zmiany obu sygnałów w prezentowanej symulacji.

1.2. Odpowiedzi skokowe, sprawdzenie liniowości obiektu

W trakcie symulacji przeprowadzono cztery zmiany sygnału sterującego, kolejno: 1,2→1,3; 1,3→1,4; 1,4→1,5 oraz ostatecznie: 1,5→0,9. Kod programu odpowiadający za symulację:

```
kk=750; % koniec symulacji

% skoki sygnału sterującego
u(1:50)=1.2; u(51:200)=1.25; u(201:350)=1.28; u(351:500)=1.32;
u(501:750)=1.27; u(751:900)=1.26; u(501:750)=0.9; u(751:900)=1.26;

y(1:11)=2; y(12:kk)=0;
% główna pętla symulacyjna
for k=12:kk
    % symulacja obiektu
    y(k)=symulacja_obiektu15y_p1(u(k-10),u(k-11),y(k-1),y(k-2));
end
```

Wyniki symulacji:



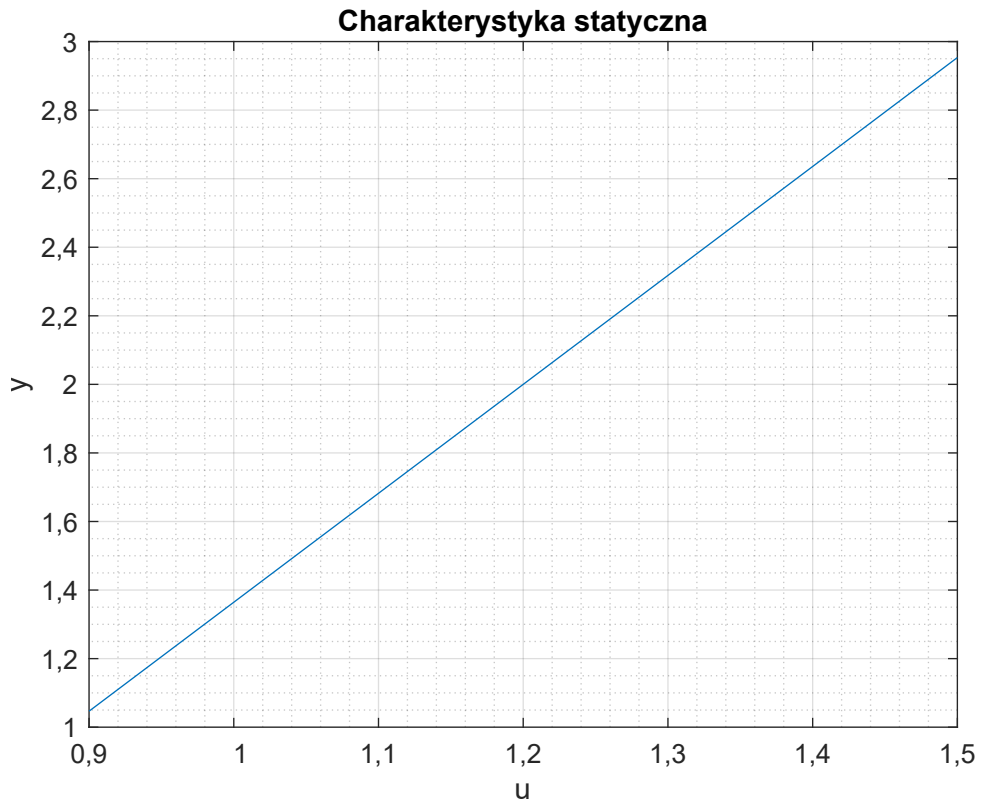
Rys. 1.2. Odpowiedzi skokowe

Z uzyskanych danych można odczytać, że skok wartości sygnału wyjściowego po ustabilizowaniu się jest w przybliżeniu proporcjonalny do skoku sygnału wejściowego. To świadczy o liniowej charakterystyce dynamicznej obiektu.

W celu wyznaczenia charakterystyki statycznej obliczono wyjścia obiektu dla wartości sygnału sterującego od 0,9 do 1,5 i krokiem 0,05. Kod programu:

```
kk=500; % koniec symulacji

us=0.9:0.05:1.5;
ys=zeros(length(us),1);

for i=1:length(us)
    u(1:kk)=us(i); y(1:11)=2; y(12:kk)=0;
    % główna pętla symulacyjna
    for k=12:kk
        % symulacja obiektu
        y(k) = symulacja_obiektu15y_p1(u(k-10), u(k-11), ...
        y(k-1), y(k-2));
    end
    ys(i) = y(kk); % zapisanie wartości y po ustabilizowaniu
end
```

Oraz wyniki symulacji:



Rys. 1.3. Charakterystyka statyczna

Właściwości statyczne obiektu także są w przybliżeniu liniowe, ponieważ otrzymany wykres jest (w przybliżeniu) liniowy. Z otrzymanych danych obliczono wzmocnienie statyczne, które wynosi około 3,1778.

1.3. Odpowiedź skokowa do zastosowania w algorytmie DMC

Odpowiedź skokowa pozyskiwana jest przez zmianę sygnału sterującego w ustabilizowanym obiekcie z 0 na 1 i rejestrowanie sygnału wyjściowego do ponownego ustabilizowania obiektu. Jednak z powodu właściwości obadanego obiektu - punktu pracy $U_{pp} = 1{,}2$ i $Y_{pp} = 2$ oraz nałożonych ograniczeń na sygnał sterujący:

$$0{,}9 \leqslant u \leqslant 1{,}5 \quad \text{oraz} \quad -5 \cdot 10^{-2} \leqslant \Delta u \leqslant 5 \cdot 10^{-2} \tag{1.1}$$

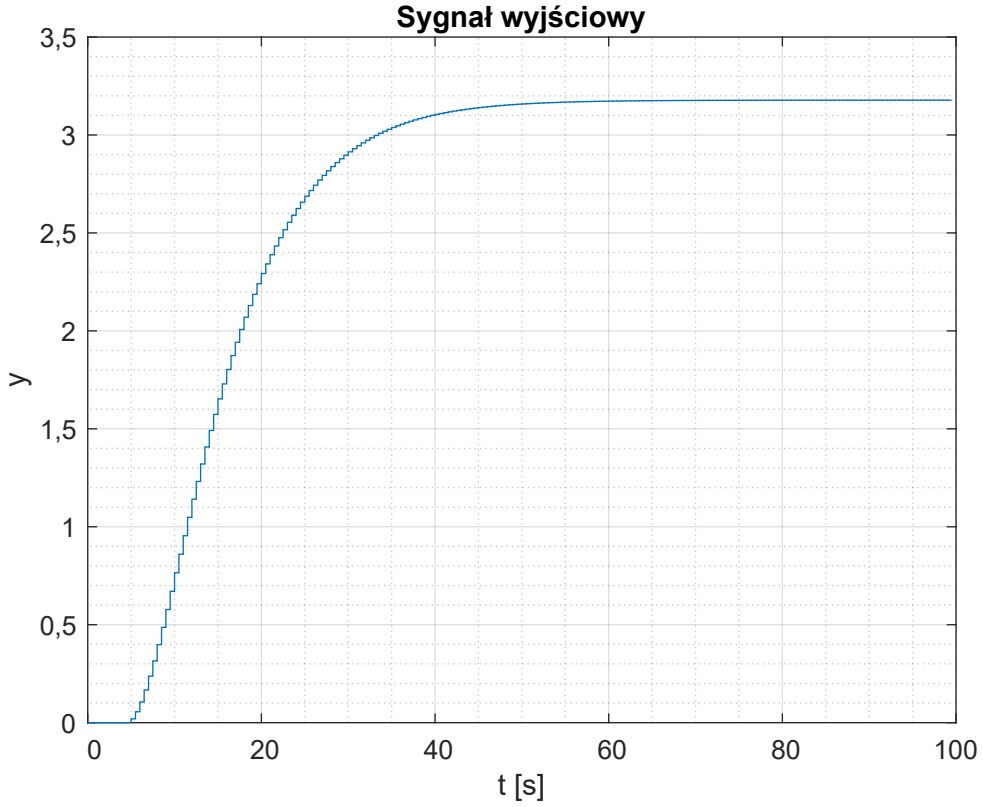
nie ma możliwości symulacji obiektu dla skoku jednostkowego. Dlatego, aby pozyskać odpowiedź skokową dla algorytmu DMC przeprowadzono skok od punktu pracy $U_{pp} = 1{,}2$ do $U_{pp} = 1{,}25$ i dokonano normalizacji odpowiedzi skokowej. Dodatkowo, aby sygnał u miał wartość 1 od chwili dyskretnej $k = 0$ (lub chwili ciągłej $t = 0$) włącznie, porzucono pierwsze elementy wektorów.

Tak przedstawia się kod funkcji do wyznaczenia odpowiedzi skokowej:

```
function [y,u] = p3_odpowiedz_skokowa(kk)
kk = kk+1;
Ypp=2; Upp=1.2; dU=0.05;

u(1:kk)=Upp+dU; y(1:11)=2; y(12:kk)=0;

% główna pętla symulacyjna
for k=12:kk
    % symulacja obiektu
    y(k) = symulacja_obiektu15y_p1(u(k-10), ...
    u(k-11), y(k-1), y(k-2));
end

% normalizacja odpowiedzi skokowej
u=(u-Upp)/dU;
y=(y-Ypp)/dU;
u(1)=[];y(1)=[];
end
```

Tak pozyskana odpowiedź skokowa jest gotowa do użycia w algorytmie DMC.

Wykres odpowiedzi skokowej:



Rys. 1.4. Odpowiedź skokowa dla algorytmu DMC

1.4. Programy do symulacji cyfrowych algorytmów sterowania

Przygotowane programy zostały zrealizowane w formie funkcji, aby móc wygodnie je wykorzystać do przyszłych zadań m.in. optymalizacji, gdzie podawane będą różne nastawy regulatorów. Ogólna idea parametrów wejściowych funkcji prezentuje się następująco:

Parametr	**Opis**
`kk`	ilość kroków symulacji
`yzad`	wektor trajektorii zmian sygnału zadanego
`...`	pozostałe parametry będące skalarami, różne dla wybranego algorytmu

Tab. 1.1. Opis parametrów wejściowych

W każdym przypadku funkcja zwraca dwa wektory, $\boldsymbol{y}$ oraz $\boldsymbol{u}$, będące odpowiednio trajektorią sygnału wyjściowego oraz wejściowego obiektu po przeprowadzonej symulacji.

1.4.1. Algorytm PID

Cyfrowa realizacja algorytmu PID wymaga pewnego przekształcenia względem wersji w czasie ciągłym. Należy przede wszystkim otrzymać równanie różnicowe, stanowiące prawo regulacji regulatora, które prezentuje się następująco:

$$u(k) = r_2 e(k-2) + r_1 e(k-1) + r_0 e(k) + u(k-1) \tag{1.2}$$

Takie równanie różnicowe posiada parametry r_0, r_1, r_2, które są powiązane ze znanymi z ciągłego regulatora PID parametrami K, T_i, T_d poprzez zależności podane wzorami:

$$r_2 = \frac{KT_d}{T_p}, \quad r_1 = K\left(\frac{T_p}{2T_i} - \frac{2T_d}{T_p} - 1\right), \quad r_0 = K\left(1 + \frac{T_p}{2T_i} + \frac{T_d}{T_p}\right) \tag{1.3}$$

W porównaniu od ich odpowiedników z wersji dyskretnej, parametry z wersji ciągłej regulatora są bardziej intuicyjne dla ludzi podczas ich analizy i modyfikacji, dlatego przygotowano dodatkową funkcję, przeliczającą te parametry podczas strojenia ręcznego, a w przypadku optymalizacji parametry r_2, r_1, r_0 będą podawane do funkcji bezpośrednio, bez potrzeby ich dodatkowego przeliczania. Ułatwi to pracę algorytmu optymalizacji i zapobiegnie tworzeniu dodatkowych minimów lokalnych.

```
function [r2, r1, r0] = p4_pid_dyskretyzacja_param(K, Ti, Td, Tp)

r2 = K*Td/Tp;
r1 = K*(Tp/(2*Ti) - (2*Td)/Tp - 1);
r0 = K*(1 + Tp/(2*Ti) + Td/Tp);

end
```

Cały program do symulacji cyfrowego algorytmu PID można podzielić na dwie części. Pierwszą część składa się z określenia wartości ograniczeń, inicjalizacji wektorów oraz przypisania wartości początkowych. Druga będąca pętlą symulacyjną, składa się z odczytania wyjścia z symulacji nieznanego obiektu, obliczenia uchybu regulacji, wyznaczenia sygnału sterującego oraz uwzględnienia ograniczeń na sygnale sterującym. Pętla ta będzie trwać aż do osiągnięcia zadanego kroku kk.

```
function [y, u] = p4_funkcja_pid(kk, yzad, r2, r1, r0)

% wartości ograniczeń
umin = 0.9; umax=1.5;
dumax = 0.05;

% inicjalizacja
u(1:kk)=0; y(1:kk)=0; e(1:kk)=0;

% warunki początkowe
u(1:11)=1.2; y(1:11)=2;

% główna pętla symulacyjna
for k=12:kk
    % symulacja obiektu
    y(k) = symulacja_obiektu15y_p1(u(k-10),u(k-11),y(k-1),y(k-2));

    % uchyb regulacji
    e(k)=yzad(k)-y(k);

    % sygnał sterujący regulatora PID
    u(k)=r2*e(k-2)+r1*e(k-1)+r0*e(k)+u(k-1);

    % ograniczenia przyrostu sygnału sterującego
    delta_u = u(k)-u(k-1);
    if  delta_u < -dumax
        u(k) = u(k-1) - dumax;
```

```
    elseif delta_u > dumax
        u(k) = u(k-1) + dumax;
    end

    % ograniczenia wartości sygnału sterującego
    if u(k) < umin
        u(k) = umin;
    elseif u(k) > umax
        u(k) = umax;
    end
end

end
```

Po wywołaniu powyższej funkcji z odpowiednimi parametrami otrzymuje się wektory obliczonych sygnałów wyjściowych i wejściowych obiektu, co będzie wykorzystywane do prezentacji działania algorytmu w dalszych częściach sprawozdania.

1.4.2. Algorytm DMC

Algorytm DMC wymaga dodatkowego podania następujących argumentów:

Parametr	Opis
`N`	horyzont predykcji
`Nu`	horyzont sterowania
`D`	horyzont dynamiki
`lambda`	współczynnik kary za zbyt duże przyrosty sterowania

Tab. 1.2. Parametry wejściowe algorytmu DMC

Cały program do symulacji algorytmu DMC w wersji analitycznej można podzielić na dwie części, obliczenia offline oraz online.

Tak jak w przypadku algorytmu PID, przed główną pętlą symulacyjną należy podać wartości ograniczeń, zainicjować wektory sygnałów oraz podać warunki początkowe. Dodatkowo w części offline należy załadować odpowiedź skokową modelu obiektu i na jej podstawie skonstruować macierz dynamiczną $\boldsymbol{M}$ oraz macierz $\boldsymbol{M_p}$, a następnie wyznaczyć macierz sterującą $\boldsymbol{K}$. Stosując wersję oszczędną należy także wyznaczyć k_e będący sumą elementów pierwszego wiersza macierzy $\boldsymbol{K}$ oraz wyznaczyć $\boldsymbol{k_u}$ poprzez wymnożenie pierwszego wiersza $\boldsymbol{K}$ z macierzą $\boldsymbol{M_p}$.

W części online znajdującej się w głównej pętli symulacyjnej najpierw należy odczytać wyjście z symulacji nieznanego obiektu, obliczyć uchyb oraz przyrost sygnału sterującego. Następnie należy uwzględnić wszelkie ograniczenia sygnału sterującego, zaktualizować sygnał sterujący oraz wektor przeszłych przyrostów sterowania.

```
function [y, u] = p4_funkcja_dmc(kk, yzad, N, Nu, D, lambda)

% Wartości ograniczeń
umin = 0.9; umax=1.5;
dumax = 0.05;

% Odpowiedź skokowa zdyskretyzowanego systemu
ys = p3_odpowiedz_skokowa(D);

% Konstrukcja macierzy M
M = zeros(N,Nu);
```

```
for column=1:Nu
    M(column:N,column) = ys(1:N-column+1);
end

% Konstrukcja macierzy Mp
Mp = zeros(N,D-1);
for j = 1:D-1
    for i = 1:N
        c = min([i+j,D]);
        Mp(i,j) = ys(c) - ys(j);
    end
end

% Obliczenie macierzy sterującej K
K = (M.'*M + lambda*eye(Nu,Nu))\M.';
K1 = K(1,:);
ke = sum(K1);
ku = K1*Mp;

% Inicjalizacja
u(1:kk)=0; y(1:kk)=0; e(1:kk)=0;
delta_u_p(1:D-1)=0; % Przeszłe przyrosty u

% Warunki początkowe
u(1:11)=1.2; y(1:11)=2;

% Główna pętla symulacyjna
for k=12:kk
    % Symulacja obiektu
    y(k)=symulacja_obiektu15y_p1(u(k-10),u(k-11),y(k-1),y(k-2));

    % Uchyb regulacji
    e(k)=yzad(k)-y(k);

    % Obliczenie przyrostu sygnału sterującego DMC
    delta_u = ke * e(k) - ku * delta_u_p';

    % Ograniczenia przyrostu sygnału sterującego
    if  delta_u < -dumax
        delta_u = -dumax;
    elseif delta_u > dumax
        delta_u = dumax;
    end

    % Ograniczenia wartości sygnału sterującego
    if u(k-1)+delta_u < umin
        delta_u = umin-u(k-1);
    elseif u(k-1)+delta_u > umax
        delta_u = umax-u(k-1);
    end
```

```
    % Aktualizacja sygnału sterującego
    u(k)=u(k-1)+delta_u;

    % Aktualizacja przeszłych przyrostów sterowania
    for n=D-1:-1:2
        delta_u_p(n) = delta_u_p(n-1);
    end
    delta_u_p(1) = delta_u;
end

end
```

Istotne jest tu zwrócenie uwagi na kolejność operacji z sygnałem sterującym, gdyż wszelkie ograniczenia należy sprawdzać i korygować na obliczonym w danej iteracji przyroście, aby do wektora przeszłych przyrostów zapisać prawdziwy przyrost sygnału sterującego, a dopiero później obliczyć sygnał sterujący,

1.5. Dobór nastaw regulatorów metodą eksperymentalną

1.5.1. Strojenie regulatora PID

Do ręcznego strojenia regulatora PID wykorzystano funkcję do przeliczania parametrów K, T_i, T_d na r_0, r_1, r_2 opisaną w podrozdziale 1.4.1. Proces strojenia rozpoczęto od zadania przypadkowych, ale rozsądnych początkowych wartości, bazując na wcześniejszym doświadczeniu w strojeniu regulatorów PID. Początkowo wybrane parametry regulatora przedstawiono w tabeli 2.7.

Tab. 1.3. Początkowe parametry regulatora PID

Parametr	Wartość
K	0,5
T_i	10,0
T_D	1,0

Analizę jakościową działania regulatora PID oparto na 2 skokach wartości zadanej jednej do 2,5 w celu sprawdzenia działania regulatora podczas zwiększenia wartości zadanej o standardową wartość, oraz do 1,2 w celu przetestowania pracy PID-a przy skoku w dół oraz w granicach końca zakresu regulacji wynikającego z ograniczeń wartości maksymalnej oraz minimalnej sygnału sterującego u. Po wielu symulacjach udało się wyznaczyć zadowalające przebiegi sygnału wejściowego oraz wyjściowego, które zostały przedstawione w tabeli 1.4.

Tab. 1.4. Parametry PID dobrane metodą eksperymentalną

Parametr	Wartość
K	0,09
T_i	7,5
T_D	1,8

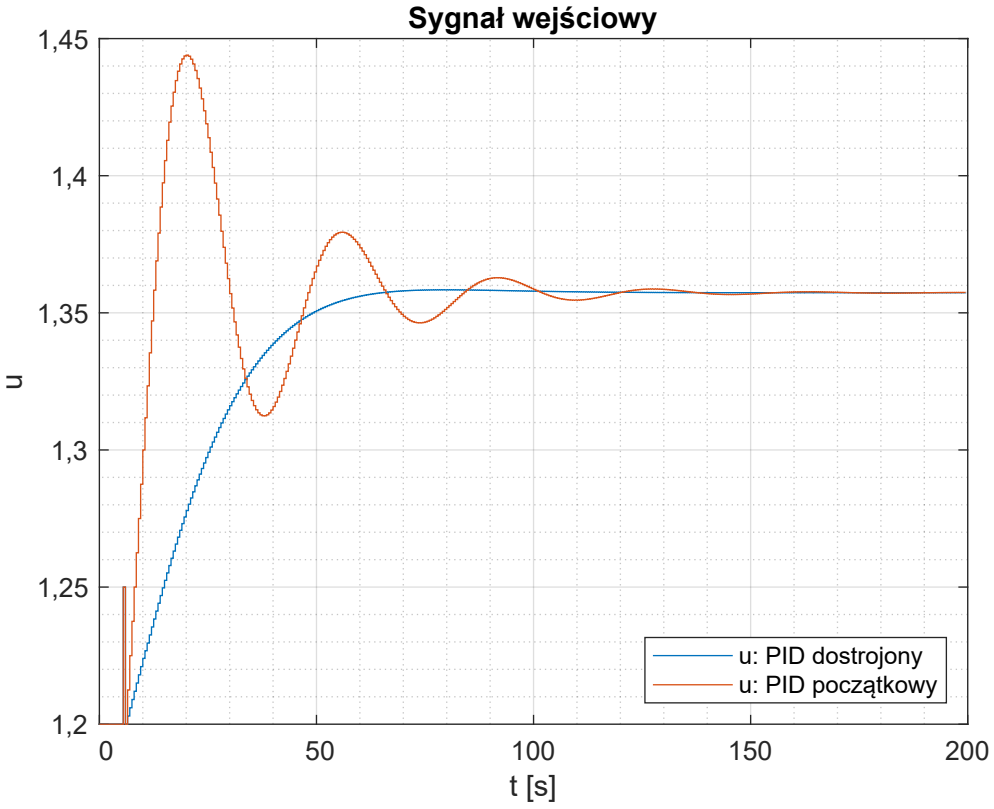
Porównanie przebiegów sygnałów sterujących oraz wyjściowych dla regulatora początkowego oraz ostatecznie dostrojonego metodą eksperymentalną przedstawiono na wykresach: 1.5, 1.6, 1.7, 1.8. Na podstawie tych wykresów można uznać, że PID początkowy znacznie szybciej zbliża się do wartości zadanej niż ostatecznie dobrany, ale za to posiada przeregulowanie oraz gasnące oscylacje. Ostatecznie dobrany regulator nie jest najszybszym regulatorem, ale za to posiada 2

bardzo ważne zalety. W przypadku wszystkich skoków wartości zadanej występujące przeregulowanie jest znikome, wręcz pomijalne i do tego nie występują oscylacje wokół wartości zadanej. Jego sygnał posiada pojedynczy nagły skok sygnału sterującego, jednak nie powinien sprawiać on problemów gdyż nie przekracza maksymalnej dopuszczalnej wartości dla symulowanego obiektu oraz nie wywołuje dziwnych zachować wyjścia obiektu. Dla obu tych regulatorów i dla każdego ze skoków obliczono wskaźnik jakości ze wzoru 1.4 w celu porównania ich ilościowo. Wyniki umieszczono w tabeli 1.5. Ze wskaźnika jakości wynika, że początkowe regulatory były lepsze, jednak ponownie przyglądając się wykresom sygnałów wejściowych i wyjściowych można stwierdzić, że dla zdecydowanej większości przypadków zastosowanie regulatora początkowe byłoby nieakceptowalne, z powodu przedstawionych powyżej jego wad.

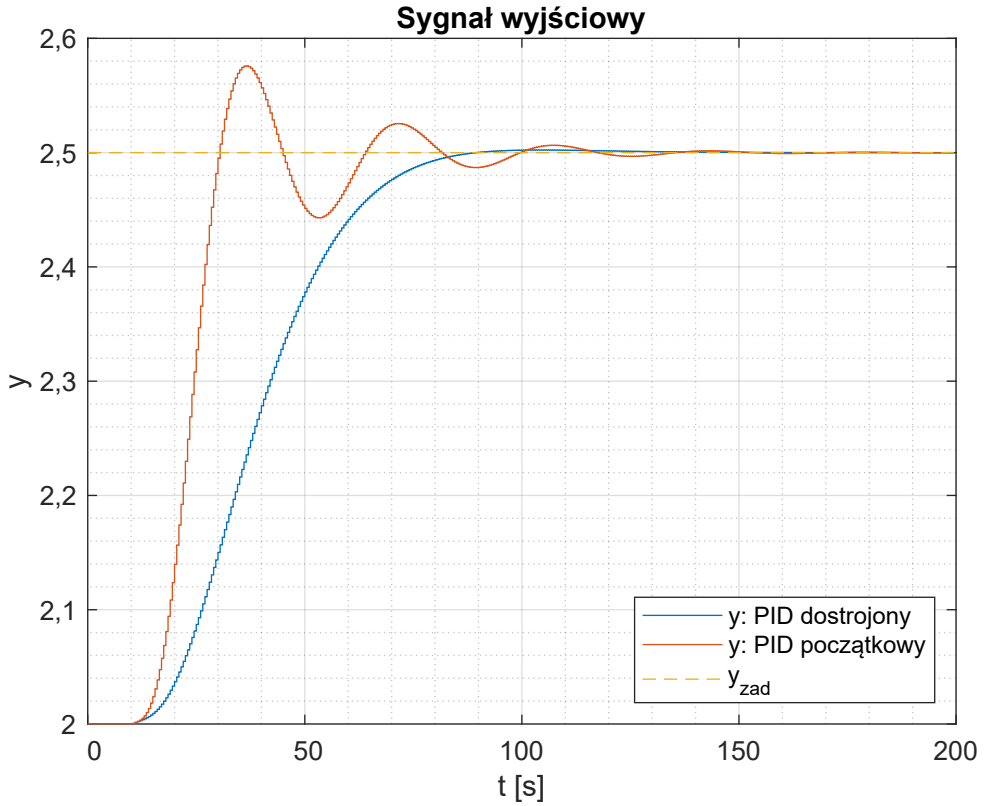
$$E = \sum_{k=1}^{k_{konc}} (y_{zad}(k) - y(k))^2 \tag{1.4}$$

Tab. 1.5. Wskaźnik jakości regulatorów PID dla skoków y_{zad}

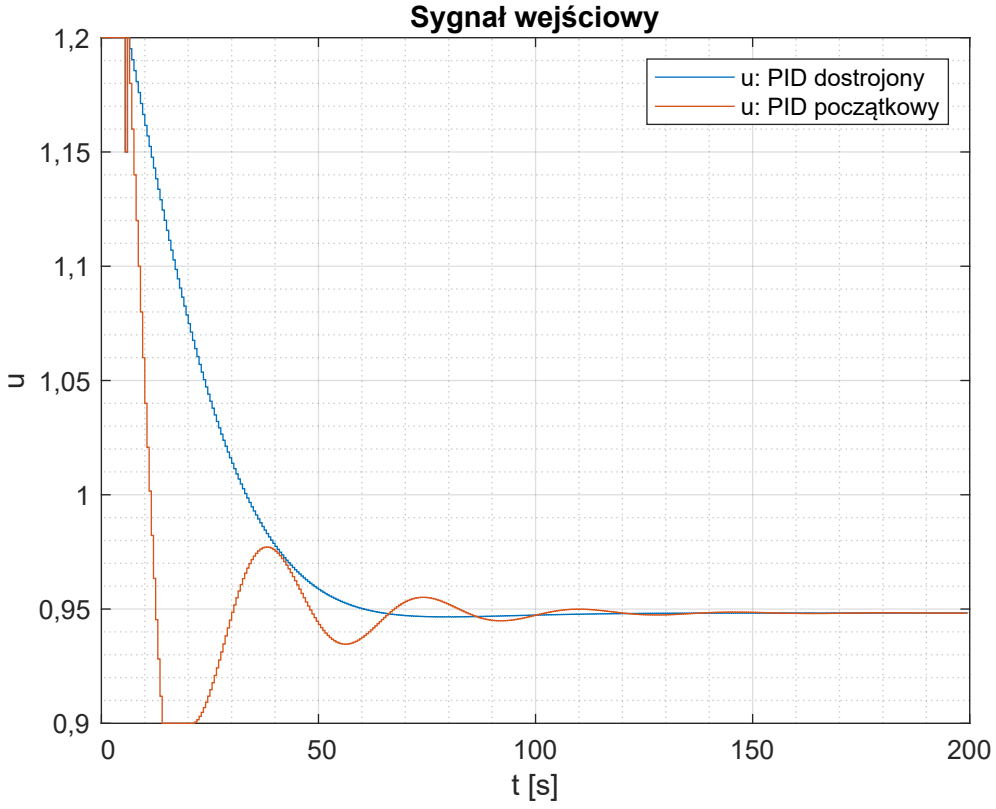
Regulator	$y_{zad} = 2{,}5$	$y_{zad} = 1{,}2$
Początkowy	10,3930	26,7901
Dostrojony	15,7049	40,2810



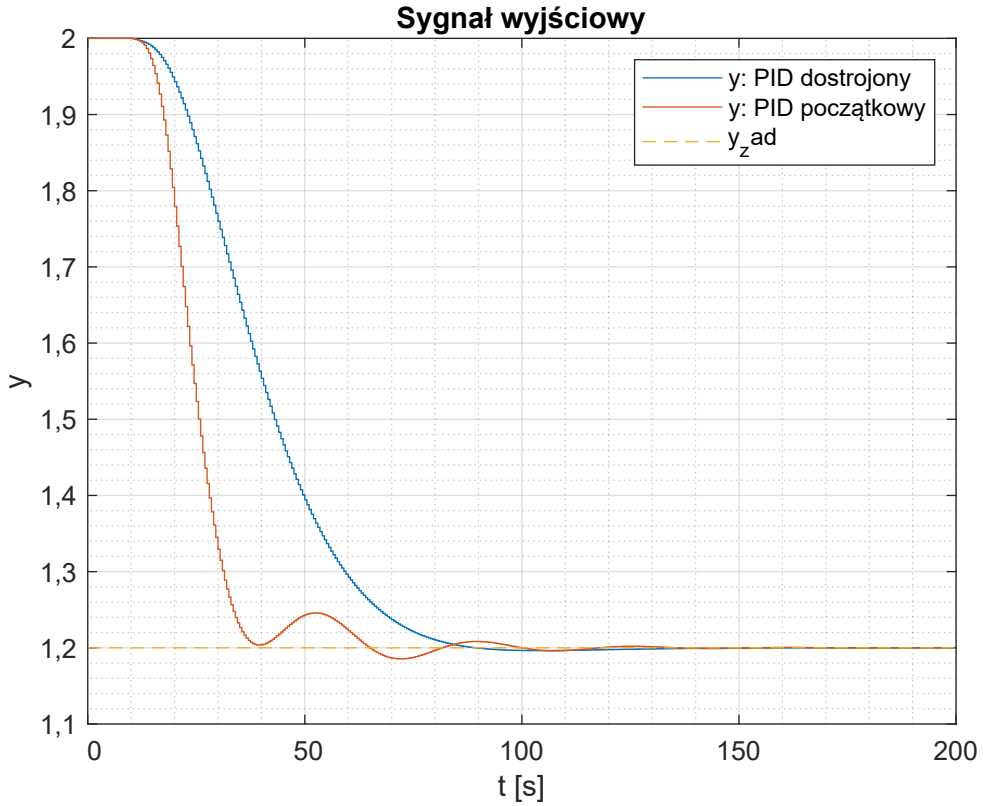
Rys. 1.5. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.6. Porównanie wyjścia wyjścia w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.7. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 1{,}2$



Rys. 1.8. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 1{,}2$

1.5.2. Strojenie regulatora DMC

W celu dostrojenia regulatora DMC jako pierwszy wyznaczono horyzont dynamiki obiektu D, który określono na podstawie wykresu odpowiedzi skokowej 1.4. Zauważono, że po $k = 130$, można uznać, że wyjście obiektu się ustabilizowało, więc przyjęto horyzont dynamiki równy 130, a pozostałe parametry bazując na wcześnie zdobytej wiedzy na temat regulatora DMC tak aby stanowiły odpowiedni początek do dalszego strojenia regulatora. Te parametry przedstawiono w tabeli 1.6.

Tab. 1.6. Parametry początkowe DMC

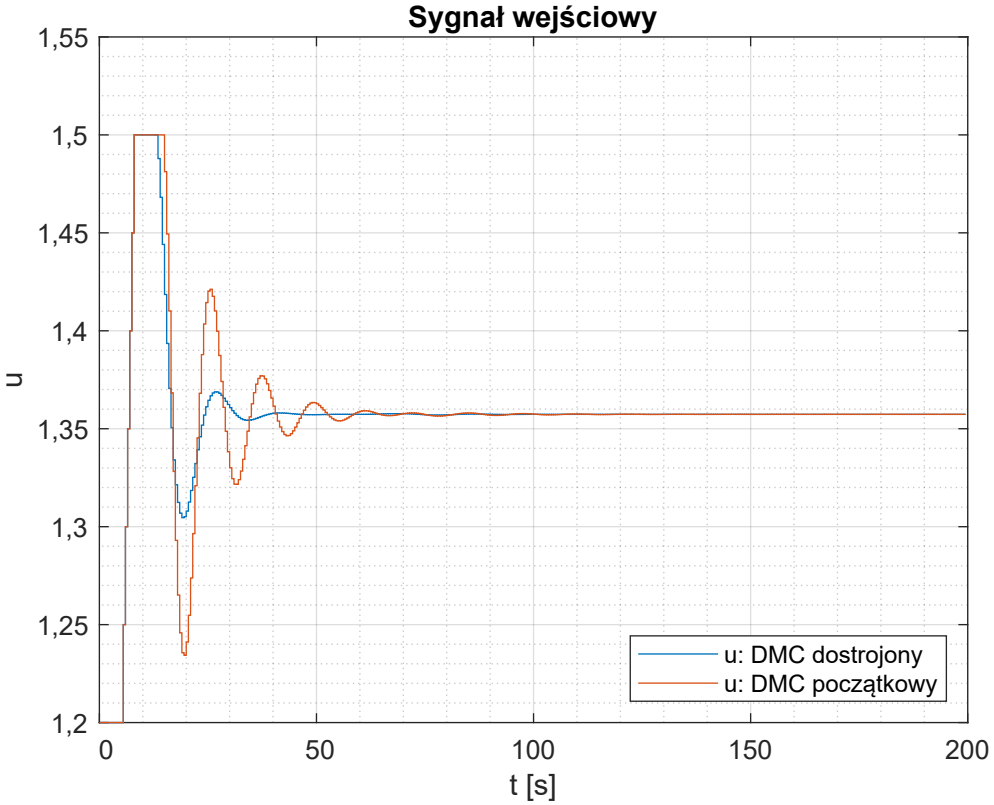
Parametr	Wartość
N	17
N_u	10
D	130
λ	1

Następnie przeprowadzono wiele symulacji dostrajając regultor DMC tak, aby otrzymać jak najmniejsze przeregulowanie, zniwelować gasnące oscylacje do minimum oraz aby sygnał sterujący maksymalnie wykorzystywał możliwości zmian swojej wartości skracając czas regulacji. Dodatkowo wzięto pod uwagę nakład obliczeń wymagany przy zwiększających się wymiarach macierzy, poprzez nie zwiększanie odpowiadających za to parametrów gdy nie przynosiło to zauważalnej poprawy regulacji. Przedstawione cele udało się osiągnąć dla regulatora DMC z parametrami przedstawionymi w tabeli 1.7.

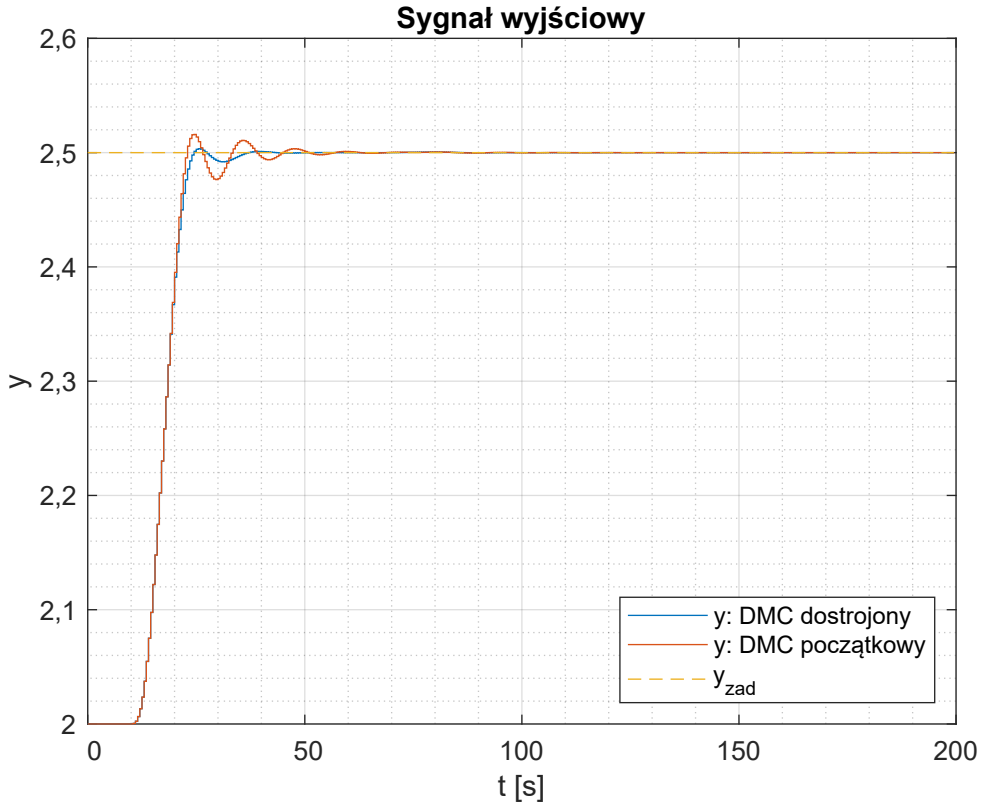
Tab. 1.7. Parametry DMC dobrane metodą eksperymentalną

Parametr	Wartość
N	35
N_u	30
D	130
λ	3

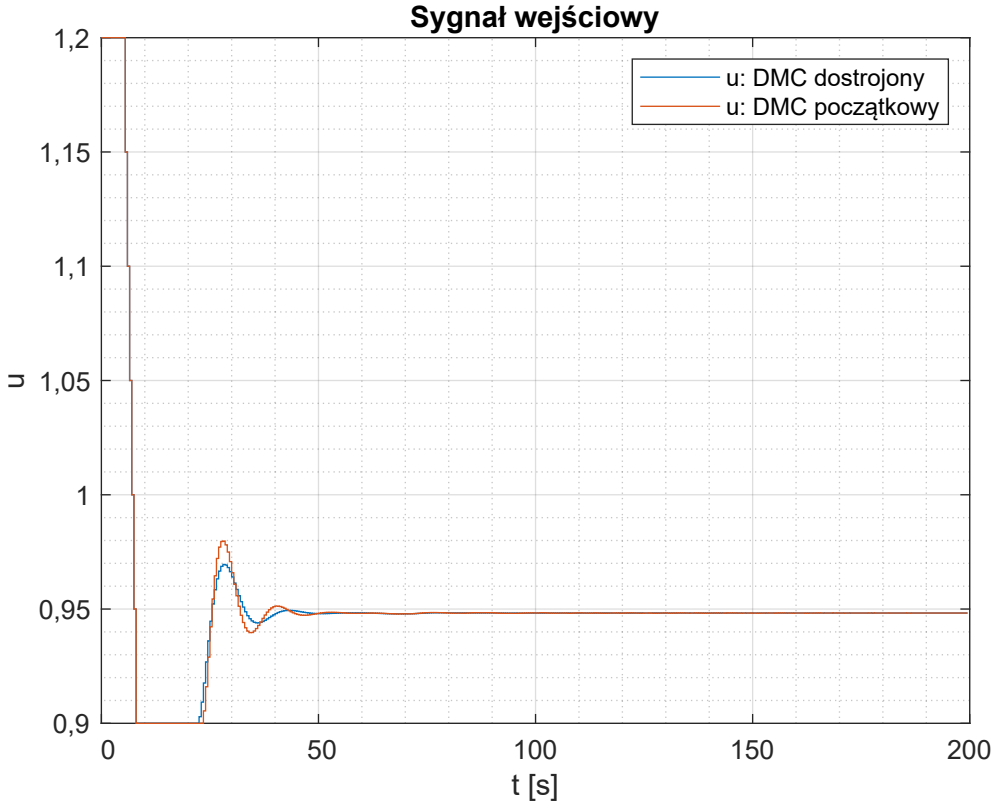
W celu porównania regulatorów wykonano skoki wartości zadanej takie jak w przypadku regulatora PID, czyli do $y_{zad} = 2{,}5$ oraz $y_{zad} = 1{,}2$. Porównanie jakościowe tych dwóch regulatorów można zaobserwować na wykresach: 1.9, 1.10, 1.11, 1.12. Jak można zauważyć początkowy regulator ma problem z przeregulowaniem oraz znacznymi oscylacjami wokół wartości zadanej. Ostatecznie dobrany regulator jest znacznie szybszy od regulatora PID dobranego eksperymentalnie, ponieważ znacznie lepiej wykorzystuje zakres możliwości zmian sygnału zadanego. Niestety w tym regulatorze pojawia się problem z odbiciem od wartości zadanej co jest głównym problemem tak dostrojonego regulatora, jednakże nie jest to przeregulowanie, dzięki czemu można uniknąć przekroczenia maksymalnej wartości wyjścia, gdyby taka istniała w obiekcie, jednocześnie mając bardzo szybki regulator. Da się również zaobserwować minimalne oscylacje wokół wartości zadanej. jednak one nie są znaczące.



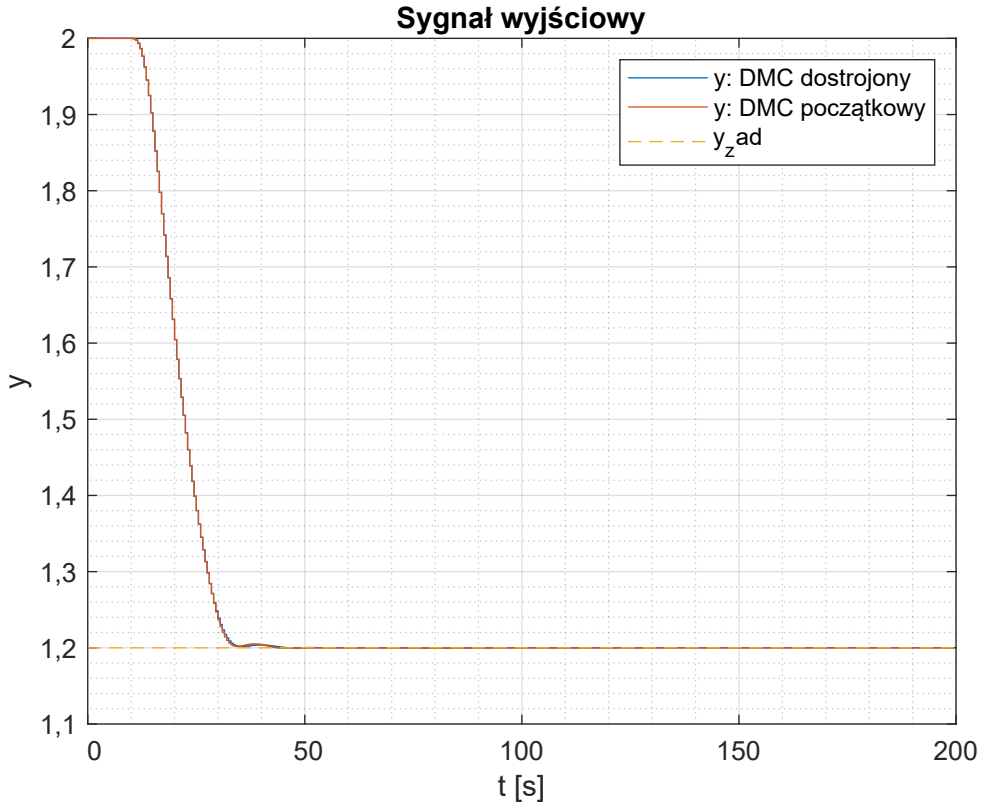
Rys. 1.9. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.10. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.11. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 1{,}2$



Rys. 1.12. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 1{,}2$

Dokonano również porównania ilościowego powyżej opisanych regulatorów DMC przy pomocy wskaźnika jakości o postaci 1.4, a ich wyniki przedstawiono w tabeli 1.8. Z porównania ilościowego tych wskaźników wynika, że oba regulatory DMC działają bardzo podobnie, więc wybór ostatecznego należy podjąć bazując na porównaniu jakościowym.

Tab. 1.8. Wskaźnik jakości regulatorów DMC dla skoków y_{zad}

Regulator	$y_{zad} = 2{,}5$	$y_{zad} = 1{,}2$
Początkowy	7,9521	22,9344
Dostrojony	7,9543	22,9356

1.6. Dobór nastaw regulatorów metodą optymalizacji współczynnika jakości regulacji

W celu optymalizacji wykorzystano kryterium przedstawione we wzorze 1.5, które minimalizowano w celu znalezienia najlepszego regulatora.

$$E = \sum_{k=1}^{k_{konc}} (y_{zad}(k) - y(k))^2 \tag{1.5}$$

1.6.1. Strojenie regulatora PID

Do optymalizacji użyto funkcji `fmincon` z ograniczeniami pozwalającymi na znalezienie regulatora z najmniejszym błędem 1.5. Otrzymane parametry dyskretnego regulatora PID przedstawiono w tabeli 1.9. Poniżej przedstawiono kod służący do optymalizacji funkcji błędu.

```
function E = pid_funkcja_kosztu(parametry, kk, yzad)
    r0 = parametry(1);
    r1 = parametry(2);
    r2 = parametry(3);

    [y, ~] = p4_funkcja_pid(kk, yzad, r2, r1, r0);

    E = (yzad-y')'*(yzad-y');
end

clear;
parametry_pocz = [0.0, 0.0, 0.0];
kk = 400;
yzad = 2.5;
yzad = yzad*ones(kk,1);

ogr_dol = [-1, -1, -1];
ogr_gor = [1, 1, 1];

parametry_optymalne = fmincon(@(parametry) ...
    pid_funkcja_kosztu(parametry, kk, yzad), ...
    parametry_pocz, [], [], [], [], ogr_dol, ogr_gor);

r2=parametry_optymalne(3);
r1=parametry_optymalne(2);
r0=parametry_optymalne(1);
```

Tab. 1.9. Parametry regulatora PID dobrane metodą optymalizacji

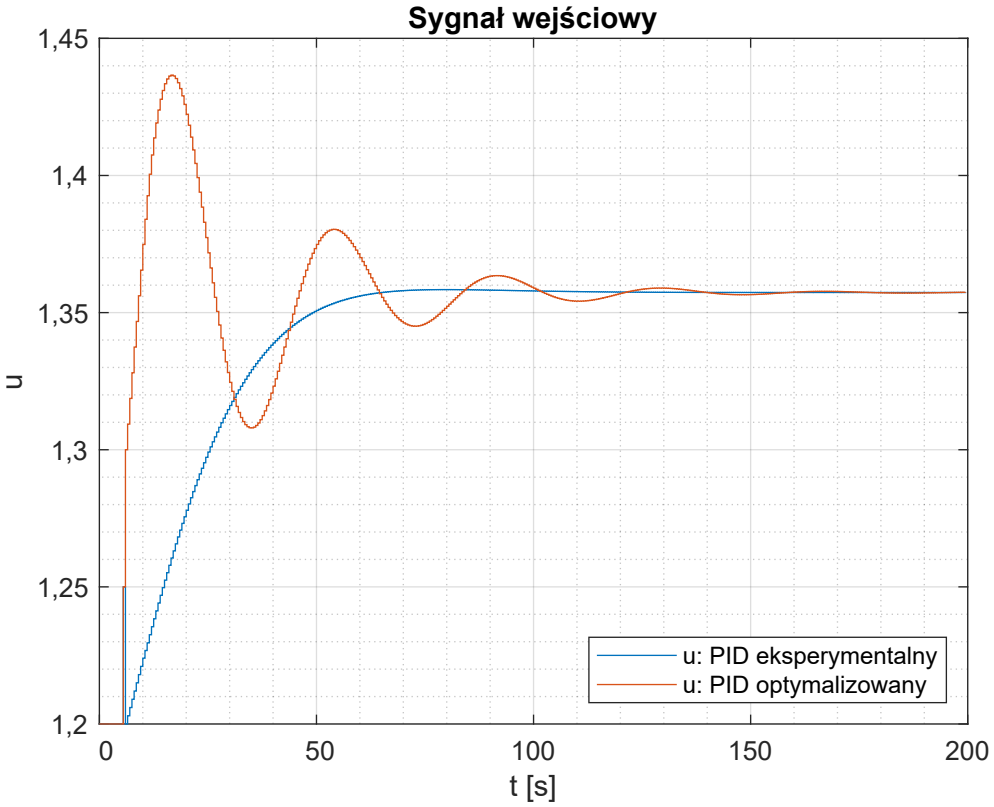
Parametr	Wartość
r_0	0,362 430
r_1	−0,262 430
r_2	−0,081 289

Otrzymany drogą optymalizacji regulator PID, porównano z wcześniej dobranym regulatorem PID metodą eksperymentalną, którego parametry można znaleźć w tabeli 1.4. Porównując te regulatory jakościowo można zauważyć że regulator otrzymany poprzez optymalizację jest szybszy przy zbliżaniu się do wartości zadanej, ale wolniej osiąga wartość zadaną względem

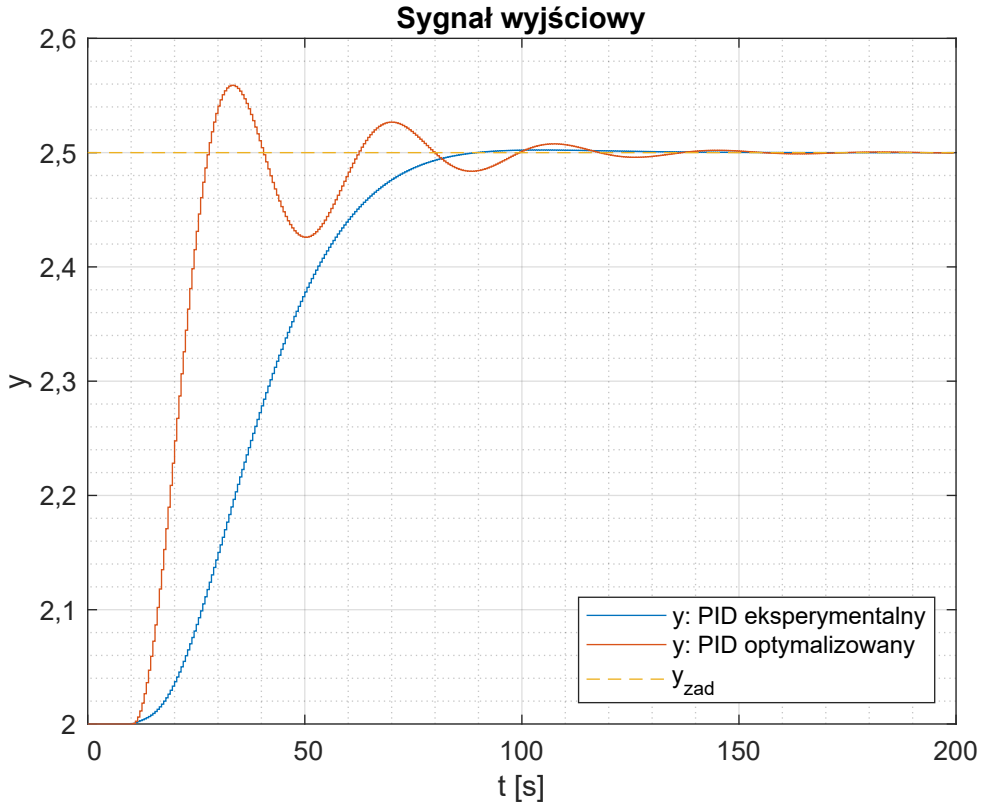
regulatora eksperymentalnego. Dodatkowo posiada oscylacje gasnące dość długo oraz znaczne przeregulowanie, które w wielu przypadkach okazałoby się nieakceptowalne. Za to porównujący błędy przedstawione w tabeli 1.10 błąd tego regulatora jest znacznie mniejszy od eksperymentalnego, gdyż jest szybszy, pomimo swoich oscylacji. Pokazuje to, że sugerowanie się jedynie wyliczanym błędem niekoniecznie jest najlepszym rozwiązaniem w przypadku strojenia regulatora PID. Jednakże można je z powodzeniem wykorzystać zamiast metody Zieglera-Nicolsa w celu znalezienia wstępnych parametrów do dalszego samodzielnego dostrojenia regulatora na fizycznym obiekcie, co zostało wykonane podczas zadań laboratoryjnych.

Tab. 1.10. Wskaźnik jakości regulatorów PID dla skoków y_{zad}

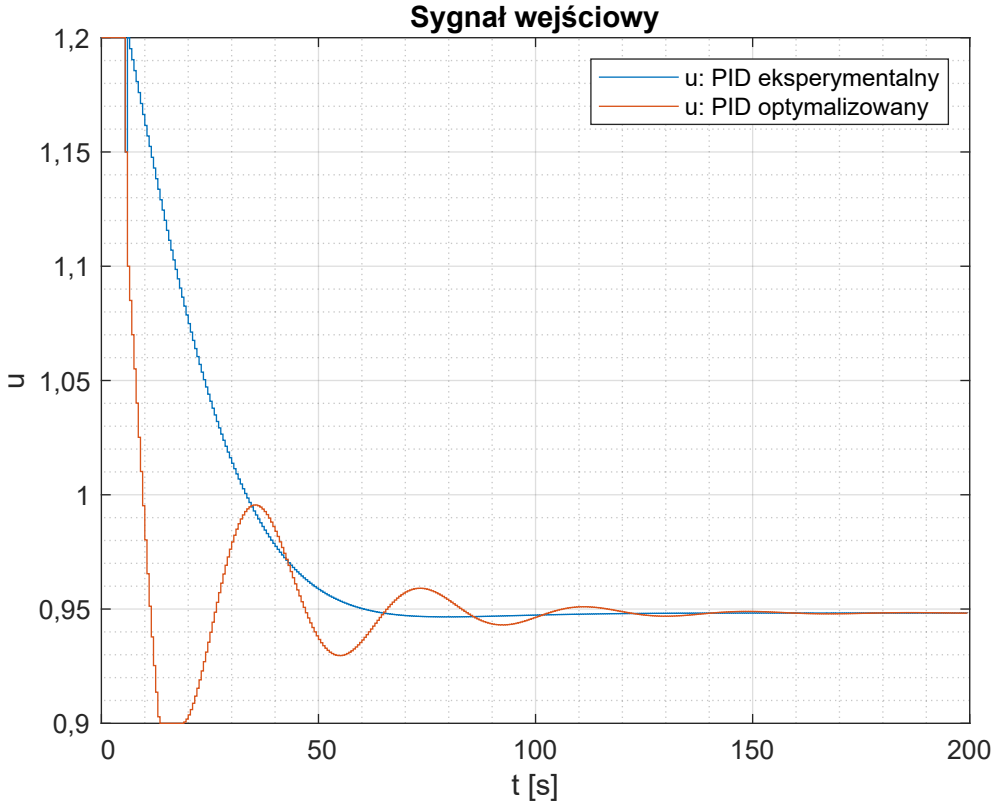
Regulator	$y_{zad} = 2{,}5$	$y_{zad} = 1{,}2$
Eksperymentalny	15,7049	40,2810
Optymalizowany	9,0710	24,8841



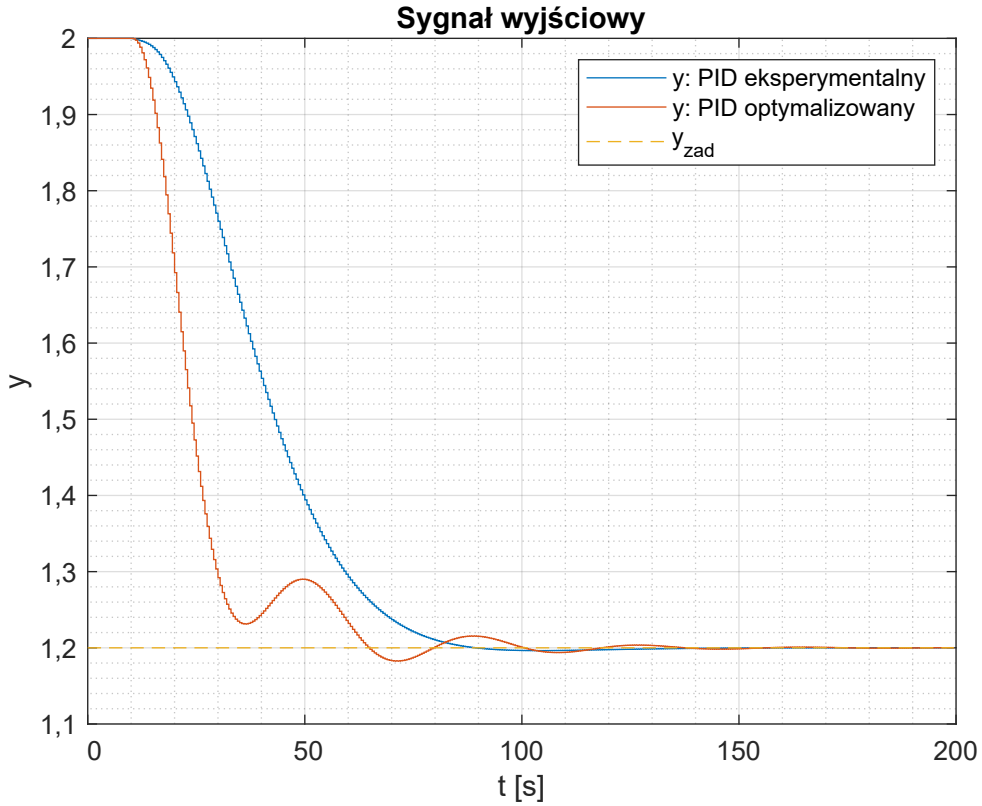
Rys. 1.13. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.14. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.15. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 1{,}2$



Rys. 1.16. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 1{,}2$

1.6.2. Strojenie regulatora DMC

Do optymalizacji użyto funkcji `ga`, zamiast `fmincon`, ponieważ parametry N oraz N_u muszą być liczbami całkowitymi co umożliwia funkcja `ga`. Minimalizowano funkcję błędu przedstawioną we wzorze 1.5. Jedyne ustawienia jakich wymagała funkcja to ograniczenia wartości parametrów, których poszukiwano. Od dołu ograniczono je kierując się tym jakie mogą być minimalne wymiary macierzy oraz tym, że funkcja kosztu został stworzona w celu karania za zbyt dużą zmienność sygnału sterującego, więc musiała być nieujemna, aby nie nagradzać za za duża zmienność sterowania. Od góry ograniczano parametry tak aby znalezione minimum funkcji kosztu nie korzystało z wartości maksymalnej parametrów oraz aby jednocześnie miało jak maksymalnie zawężony zakres poszukiwania. Ostatecznie znalezione parametry poprzez minimalizację funkcji błędu przedstawiono w tabeli 1.11, a kod służący do optymalizacji w poniższym listingu.

```
function E = dmc_funkcja_kosztu(parametry, kk, yzad, D)
    N = parametry(1);
    Nu = parametry(2);
    lambda = parametry(3);

    [y, ~] = p4_funkcja_dmc(kk, yzad, N, Nu, D, lambda);

    E = (yzad - y')' * (yzad - y');
end

parametry_pocz = [30, 30, 1];
kk = 400;
yzad = 2.5;
yzad = yzad * ones(kk, 1);
```

```
D = 200;

ogr_dol = [1, 1, 0.0];
ogr_gor = [60, 100, 50];

IntCon = [1, 2];
options = optimoptions('ga', 'Display', 'iter');

parametry_optymalne = ga(@(parametry) ...
    dmc_funkcja_kosztu(parametry, kk, yzad, D), ...
    3, [], [], [], [], ogr_dol, ogr_gor, [], IntCon, options);

N=parametry_optymalne(1);
Nu=parametry_optymalne(2);
lambda=parametry_optymalne(3);
```

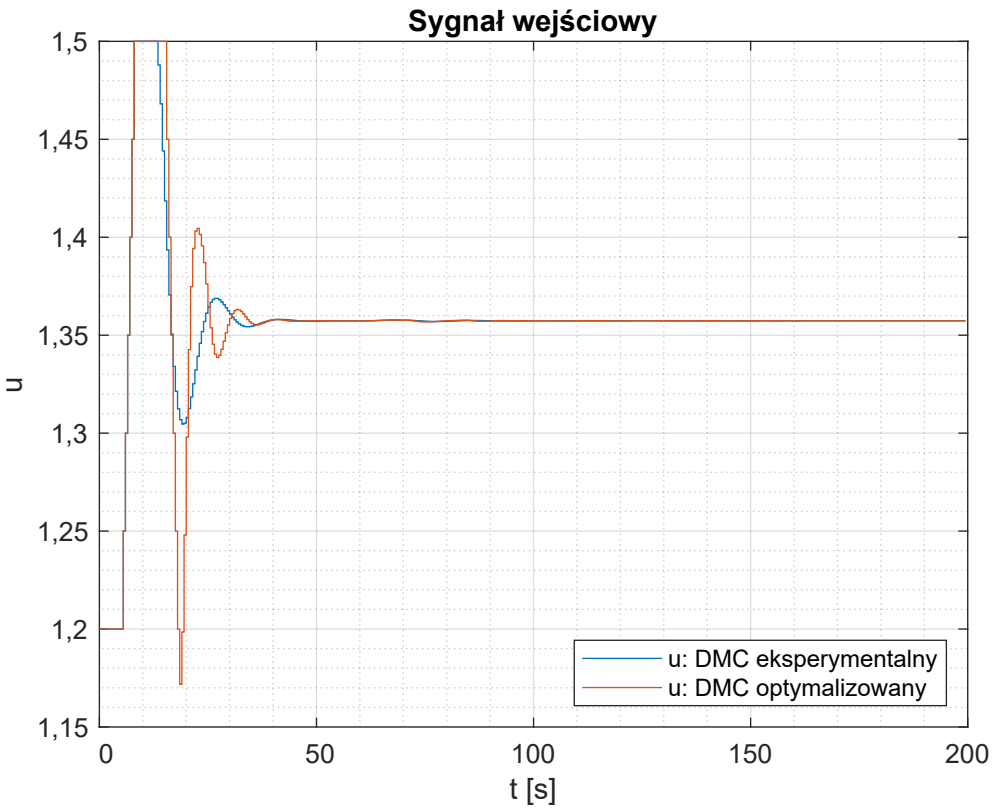
Tab. 1.11. Parametry DMC dobrane metodą optymalizacji

Parametr	Wartość
N	35
N_u	30
D	130
λ	3

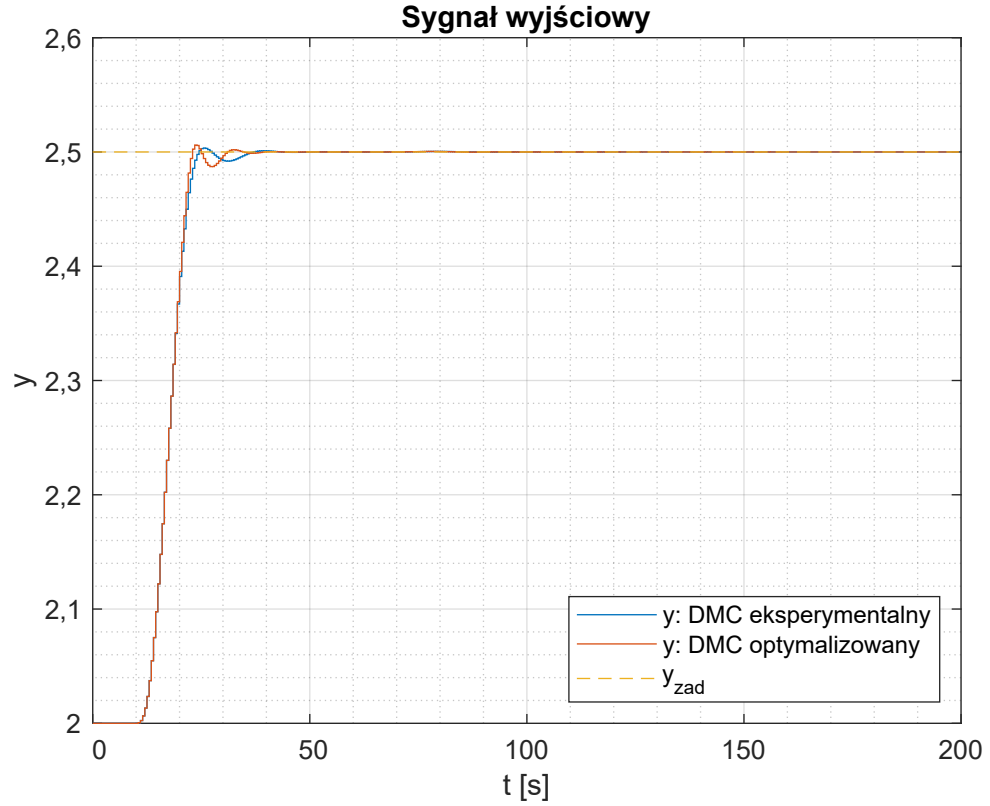
Dobrany metodą optymalizacji regulator DMC porównano z regulatorem DMC dobranym metodą eksperymentalną z parametrami z tabeli 1.7. Porównanie ilościowe wskazuje, że dobrany metodą optymalizacji regulator jest nieznacznie lepszy. Porównując wykresy sygnału wejściowego obiektu oraz sygnału wyjściowego dla tych dwóch regulatorów można zauważyć, że eksperymentalny regulator DMC ma mniejsze przeregulowanie oraz mniejsze oscylacje, za to regulator optymalizowany szybciej osiąga zadaną wartość wyjścia obiektu. Dodatkowo ten regulator mniej płynnie zmienia wartość wejścia obiektu, co nie zawsze może być pożądane, zwłaszcza przy tak małych różnicach w jakości regulacji.

Tab. 1.12. Wskaźnik jakości regulatorów DMC dla skoków y_{zad}

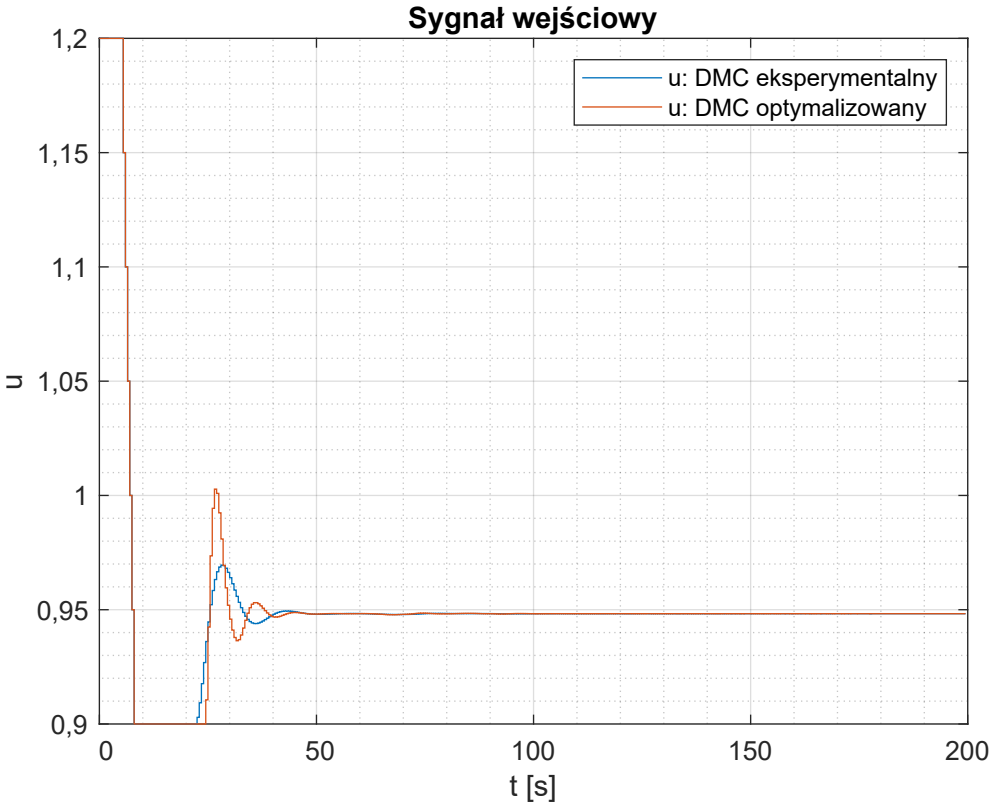
Regulator	$y_{zad} = 2{,}5$	$y_{zad} = 1{,}2$
Eksperymentalny	7,9543	22,9356
Optymalizowany	7,9473	22,9343



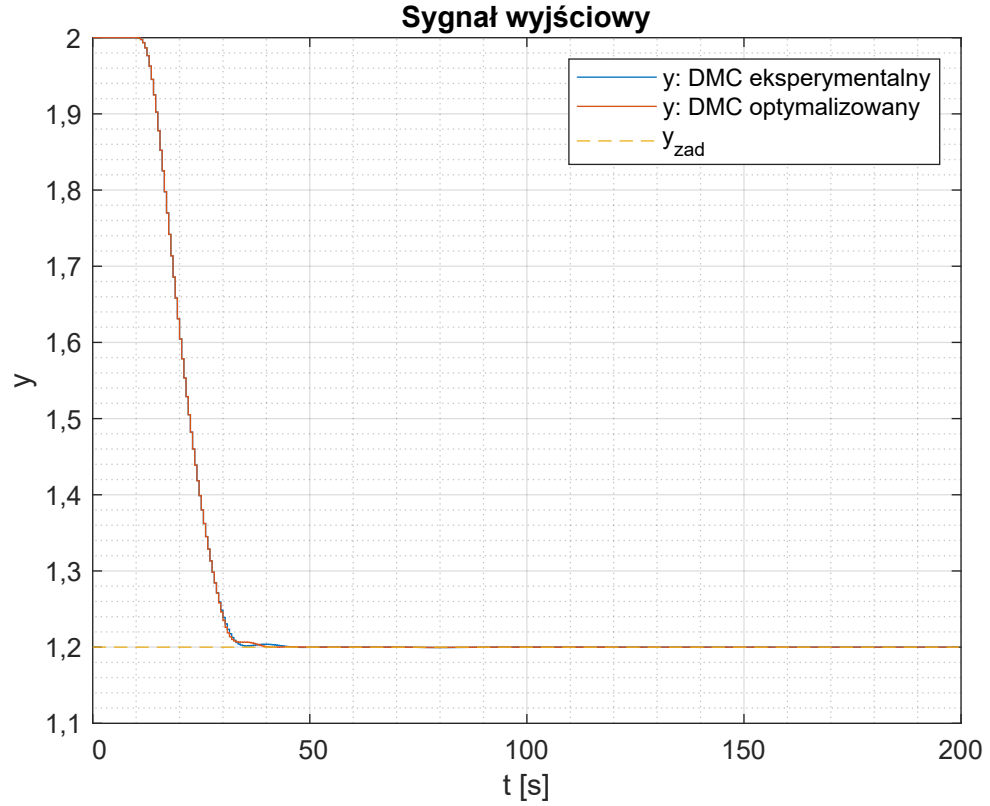
Rys. 1.17. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.18. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 2{,}5$



Rys. 1.19. Porównanie wartości sterowania w czasie dla skoku do $y_{zad} = 1{,}2$



Rys. 1.20. Porównanie wartości wyjścia w czasie dla skoku do $y_{zad} = 1{,}2$

1.7. Wnioski

Część projektowa w tym sprawozdaniu skupiała się na: zbadaniu obiektu pod kątem jego punktu pracy, liniowości statyki i dynamiki; wyznaczeniu odpowiedzi skokowej z uwzględnieniem punktu pracy; implementacji algorytmów regulowania i doborze ich nastaw. Wszystkie te punkty zostały stosownie opisane w odpowiednich rozdziałach. Postanowiono jeszcze porównać regulator PID i DMC patrząc na wszystkie wykresy zarówno z części eksperymentalnej jak i z optymalizacji.

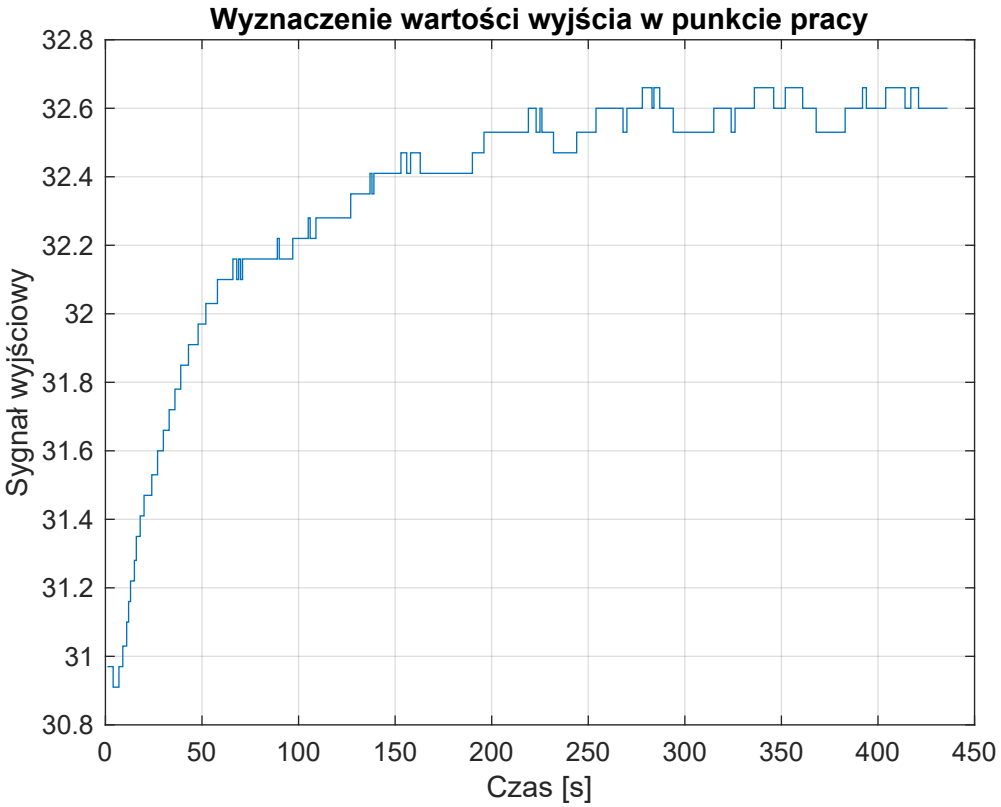
Z tego podsumowania wynika, że w zdecydowanej większości aspektów regulator DMC przeważa nad regulatorem PID. Przede wszystkim DMC jest znacznie szybszy niż PID, ponieważ wykorzystuje lepiej zakres zmian sygnału sterującego. Dodatkowo DMC jest stabilniejszy i łatwiej jest się pozbyć w nim oscylacji oraz przeregulowania. Również w przypadku DMC kryterium ilościowe lepiej odwzorowuje porównanie jakościowo, dzięki czemu do strojenia z powodzeniem można wykorzystać minimalizację funkcji błędu. Za to zaletą PID jest brak potrzeby stworzenia modelu odpowiedzi skokowej obiektu regulowanego oraz mniejszy nakład obliczeń w każdej z iteracji. Podsumowując, mając możliwość otrzymania odpowiedzi skokowej obiektu zdecydowanie lepszym rozwiązaniem będzie skorzystanie z regulatora DMC niż PID.

2. Laboratoria

2.1. Sterowanie i komunikacja ze stanowiskiem grzewczo-chłodzącym przy użyciu programu MATLAB oraz określenie punktu pracy

Dla zespołu o numerze 15 moc grzałki w punkcie pracy ma wartość 25.

W celu określenia pomiaru temperatury w punkcie pracy ustawiono wartości sygnałów G_1 na 25 oraz W_1 na 50 i odczekano do ustabilizowania się pomiaru.

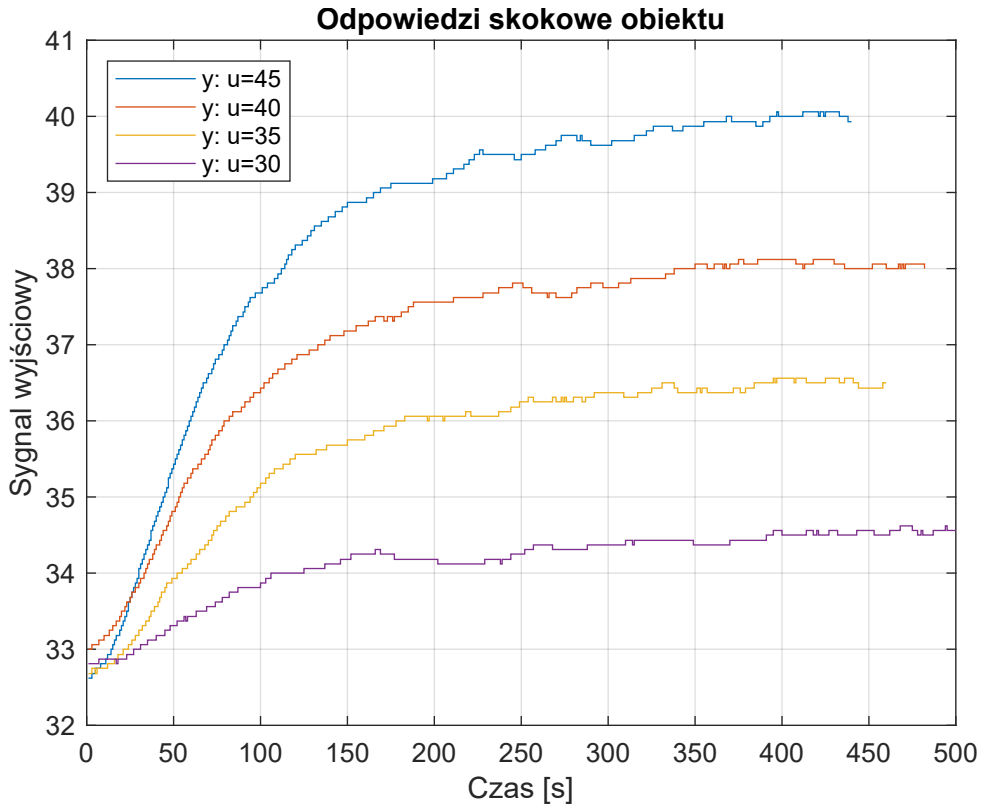


Rys. 2.1. Wykres pomiaru temperatury w punkcie pracy $G_1 = 25$

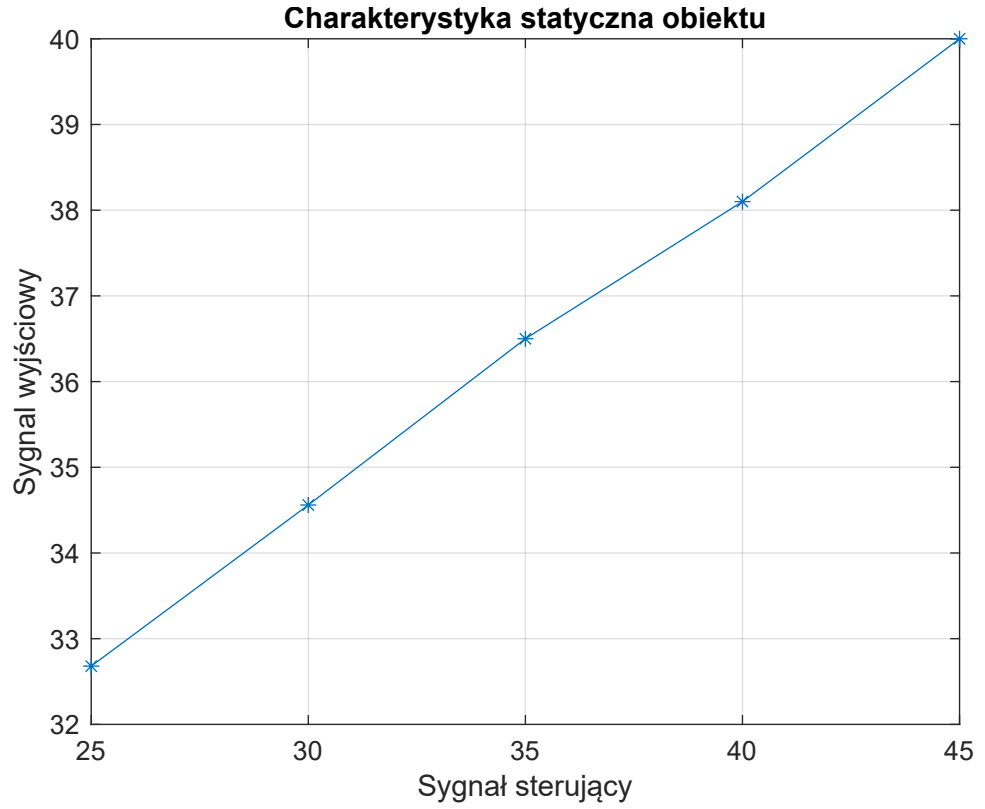
Temperatura po pięciu minutach ustabilizowała się na wartości około 32,6 °C. W kolejnych częściach laboratorium właśnie ta wartość będzie traktowana jako warunek początkowy.

2.2. Odpowiedzi skokowe procesu

Odpowiedzi skokowe wyznaczono rozpoczynając z punktu pracy, gdzie przy $G_1 = 25$ temperatura była w okolicach 32,6. Wyznaczono 4 odpowiedzi skokowe dla różnych wartości G_1 aby mieć szerszy pogląd na statykę obiektu przy skokach sygnału sterującego do wartości 30, 35, 40 oraz 45. Otrzymane odpowiedzi pokazano na rys. 2.4.



Rys. 2.2. Przebiegi odpowiedzi skokowych dla różnych wartości sygnałów sterujących



Rys. 2.3. Charakterystyka statyczna procesu

Tab. 2.1. Wzmocnienie statyczne dla poszczególnych zmian sygnału sterującego

G_1	K
30	0,341
35	0,380
40	0,335
45	0,365

Na rys. 2.3 naniesione zostały temperatury uzyskane temperatury po ich stabilizacji dla różnych wartości G_1. Dla zakresu G_1 od 25 do 45 obiekt wydaje się posiadać charakterystykę liniową. Przy takiej ilości skoków i takim czasie oczekiwania do ustabilizowania się temperatury, rzeczywiste wartości wzmocnienia statycznego mogą być inne. Widać również, że wyliczone wzmocnienia w tab. 2.1 oscylują wokół wartości 0,355, bez wyraźnej tendencji w jakimkolwiek kierunku, co zdaje się potwierdzać przypuszczenia o liniowości statyki obiektu.

2.3. Aproksymacja odpowiedzi skokowej do zastosowania w algorytmie DMC

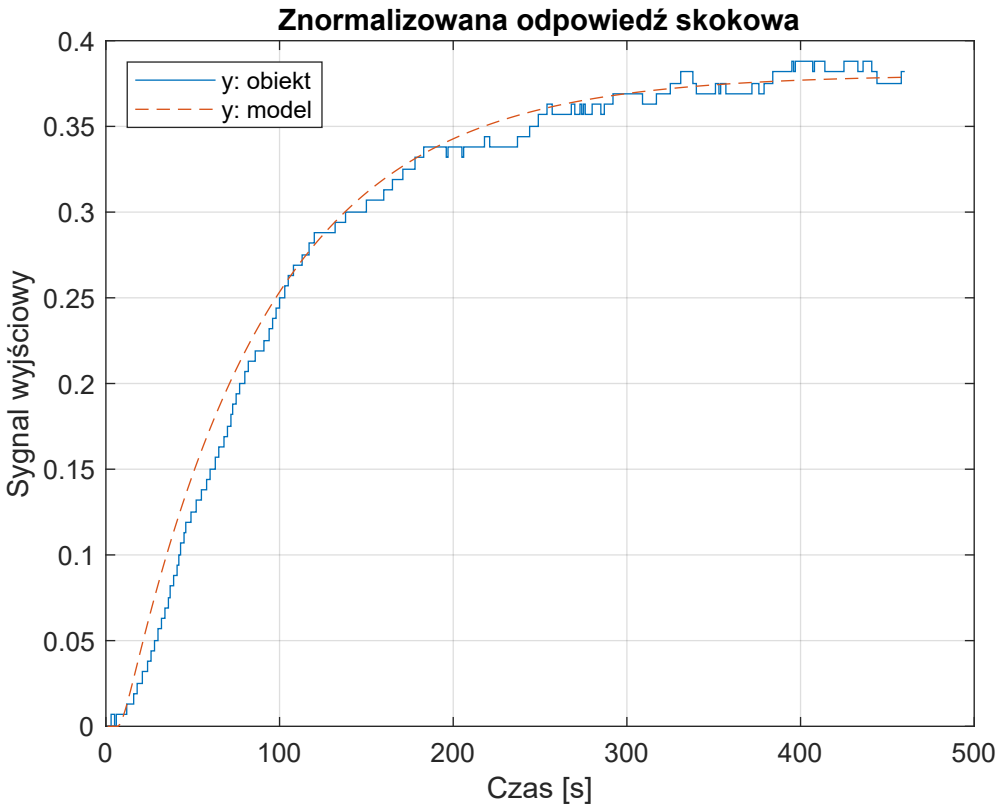
Do wykorzystania w algorytmie DMC odpowiedzi skokowej do $G_1 = 35$ pozyskanej w rozdziale 2.2 należy ją znormalizować oraz pozbyć się z niej zakłóceń i wpływu niepewności pomiaru na wyliczane w algorytmie macierze (temperatura odczytywana jest z rozdzielczością 0,06). W tym celu dokonano aproksymacji odpowiedzi skokowej używając do tego członu dwu inercyjnego z opóźnieniem oraz optymalizacji z zastosowaniem algorytmu genetycznego, ponieważ parametr T_d musi być liczbą całkowitą.

```
kk = 460;
u(1:kk)=1;
ymierz = odczyt_danych_z_txt("pomiary/skok_3_35.txt");
ymierz = (ymierz-32.68)/10; % skok z 25 do 35

parametry_pocz = [1, 1, 1, 1];
ogr_dol = [1, 1, 0, 0];
ogr_gor = [120, 120, 50, 12];

IntCon = 4;  % Td musi być liczbą całkowitą
options = optimoptions('ga', 'Display', 'iter');

parametry_optymalne = ga(@(parametry) ...
    p3_funkcja_kosztu(parametry, kk, u, ymierz), 4, ...
    [], [], [], [], ogr_dol, ogr_gor, [], IntCon, options);

T1 = parametry_optymalne(1);
T2 = parametry_optymalne(2);
K = parametry_optymalne(3);
Td = parametry_optymalne(4);
```

Funkcja użyta w powyższym kodzie do testowania modelu i wyliczania jego błędu względem pomiarów:

```
function E = p3_funkcja_kosztu(parametry,kk,u,ymierz)
T1 = parametry(1);
T2 = parametry(2);
K = parametry(3);
Td = parametry(4);
```

```
% przekształcenie parametrów transmitancji
%    na parametry równania różnicowego
alfa1 = exp(-1/T1);
alfa2 = exp(-1/T2);
a1 = - alfa1 - alfa2;
a2 = alfa1*alfa2;
b1 = (K/(T1-T2))*(T1*(1-alfa1)-T2*(1-alfa2));
b2 = (K/(T1-T2))*(alfa1*T2*(1-alfa2)-alfa2*T1*(1-alfa1));

% symulacja modelu otrzymanego w danej iteracji
y = zeros(kk+1,1);
for k=Td+3:kk+1
    y(k) = b1*u(k-Td-1) + b2*u(k-Td-2) - a1*y(k-1) - a2*y(k-2);
end
y(1)=[];

% obliczenie kosztu - błędu między pomiarami a modelem
E = (ymierz - y)' * (ymierz - y);
end
```



Rys. 2.4. Przebiegi znormalizowanej odczytanej i aproksymowanej odpowiedzi skokowej dla u=35

Tab. 2.2. Parametry aproksymacji odpowiedzi skokowej

Parametr	Wartość
T_1	82,0474
T_2	3,3333
K	0,3802
T_d	6

Parametr K jest niemal identyczny z tym w tabeli 2.1, ponieważ wyjście obu odpowiedzi - znormalizowanej oraz aproksymowanej - stabilizuje się na podobnej wartości, jednak pozostałe parametry członu dwu inercyjnego z opóźnieniem są ciężkie do uzasadnienia.

Dodatkową zaletą używania odpowiedzi skokowej określonej transmitancją jest łatwa adaptacja, kiedy potrzebna jest zmiana okresu próbkowania, np. podczas migracji projektu na inne stanowisko. W powyższym kodzie ta możliwość nie została uwzględniona, ponieważ czas próbkowania stanowiska grzewczo-chłodzącego wynosi 1s co doprowadziło do uproszczenia wzorów. Dla odpowiedzi skokowej uzyskanej pomiarami taka migracja byłaby utrudniona.

2.4. Programy do regulacji procesu stanowiska algorytmami cyfrowymi

2.4.1. Algorytm PID

Algorytm PID działający na rzeczywistym obiekcie, z którym pracowano na laboratorium w zasadzie nie różni się w koncepcji niczym od regulatora projektowego. Jedyną różnicą jest naturalnie fakt, że sygnał wyjściowy y w każdej iteracji odczytywano z obiektu, a następnie wyliczony sygnał sterujący u przekazywano na obiekt, przy czym należy pamiętać ze operowano mocą grzałki, a moc wentylatora była stała i wynosi 50%. Warto też zaznaczyć, że w trakcie działania programu tworzone były wykresy w czasie rzeczywistym obrazujące przebiegi sygnału sterującego oraz wyjściowego na tle wartości zadanej.

```
% konfiguracja wykresu
figure('Position', [100, 100, 800, 600]);

% Pierwszy subplot dla wykresu 'u'
subplot(2, 1, 1);
h1 = plot(nan, nan, 'DisplayName', 'u');
xlabel('Czas (s)');
ylabel('u');
title('u');
legend('show');
% Drugi subplot dla wykresów 'y' i 'y_zad'
subplot(2, 1, 2);
h2 = plot(nan, nan, 'DisplayName', 'y');
hold on;
h3 = plot(nan, nan, 'DisplayName', 'y_zad');
xlabel('Czas (s)');
ylabel('y');
title('y i y_zad');
legend('show');
```

Kod regulatora PID zawiera na początku inicjalizacje nastaw i parametrów symulacji (takich samych jak przy każdej poprzedniej symulacji) oraz podaniu warunków początkowych - jedyne różne od zera to u oraz y, a ich wartość jest zgodna z wyznaczonym wcześniej punktem pracy. Następnie odczytano wartość wyjścia, obliczono uchyb oraz sterowanie, które dalej sprawdzono pod

względem spełniania ograniczeń: $u_{min} = 0 \leqslant u(k) \leqslant u_{max} = 100$. Na końcu wysłano wyliczone sterowanie do obiektu i zaktualizowano wykresy ukazujące przebiegi w czasie rzeczywistym.

```
kk = 3000;
yzad(1:450) = 35; yzad(451:900) = 40; yzad(901:kk)=33;
umin = 0; umax=100;

K = 15.3505;
Ti = 45;
Td = 1.3602;
[r2, r1, r0] = p4_pid_strojenie_2(K,Ti,Td,1);

% warunki początkowe
u(1:2)=25; y(1:2)=32.68; e(1:kk)=0;
u(3:kk)=0; y(3:kk)=0;


% główna pętla symulacyjna
for k=3:kk
    % symulacja obiektu
    y(k) = readMeasurements(1);

    % uchyb regulacji
    e(k)=yzad(k)-y(k);

    % sygnał sterujący regulatora PID
    u(k)=r2*e(k-2)+r1*e(k-1)+r0*e(k)+u(k-1);

    if u(k) < umin
        u(k) = umin;
    elseif u(k) > umax
        u(k) = umax;
    end

    % kolejno W1 i G1
    sendControls ([ 1 , 5 ] , [ 50 , int64(u(k)) ]);

    % aktualizacja wykresu
    set(h1, 'YData', u(1:k), 'XData', 1:k);
    set(h2, 'YData', y(1:k), 'XData', 1:k);
    set(h3, 'YData', yzad(1:k), 'XData', 1:k);
    drawnow;

    waitForNewIteration () ; % wait for new iteration
end
```

2.4.2. Algorytm DMC

Podobnie jak opisano w rozdziale o algorytmie PID - algorytm DMC w wersji "na laboratoria" w swoim działaniu nie różni się niczym od wcześniej opisanego regulatora obiektu projektowego. W trakcie regulacji również tworzono wykresy przebiegu sygnałów w czasie rzeczywistym. Kod programu ponownie rozpoczyna się inicjalizacją nastaw regulatora oraz tym razem pobraniem

wartości współczynników odpowiedzi skokowej wyznaczonych wcześniej. Następnie ma miejsce konstrukcja macierzy oraz definicja warunków początkowych, zależnych od punktu pracy. Ostatecznie następuje wejście w pętle główną programu, w której analogicznie: odczytywana jest wartość y, obliczany uchyb, aktualizacja sygnału sterującego z uwzględnieniem ograniczeń.

```
kk=3000; % koniec symulacji
umin = 0; umax=100;

yzad(1:450) = 35; yzad(451:900) = 40; yzad(901:kk)=33;
N = 200;
Nu = 5;
D = 500;
lambda = 0.2;

% Odpowiedź skokowa zdyskretyzowanego systemu
ys = odp_jedn(750);

% Konstrukcja macierzy M
M = zeros(N,Nu);
for column=1:Nu
    M(column:N,column) = ys(1:N-column+1);
end

% Konstrukcja macierzy Mp
Mp = zeros(N,D-1);
for j = 1:D-1
    for i = 1:N
        c = min([i+j,D]);
        Mp(i,j) = ys(c) - ys(j);
    end
end

% Obliczenie macierzy sterującej K
K = (M.'*M + lambda*eye(Nu,Nu))\M.';
K1 = K(1,:);
ke = sum(K1);
ku = K1*Mp;

% Warunki początkowe
u(1:kk)=0; y(1:kk)=0; e(1:kk)=0;
u(1)=25; y(1)=32.68;
delta_u_p(1:D-1)=0; % Przeszłe przyrosty u

% Główna pętla symulacyjna
for k=2:kk
    % Symulacja obiektu
    y(k) = readMeasurements (1);

    % Uchyb regulacji
    e(k)=yzad(k)-y(k);

    % Obliczenie przyrostu sygnału sterującego DMC
```

```
    delta_u = ke * e(k) - ku * delta_u_p';

    % Ograniczenia
    if u(k-1)+delta_u < umin
        delta_u = umin-u(k-1);
    elseif u(k-1)+delta_u > umax
        delta_u = umax-u(k-1);
    end

    % Aktualizacja sygnału sterującego
    u(k)=u(k-1)+delta_u;

    % Aktualizacja przeszłych przyrostów sterowania
    for n=D-1:-1:2
        delta_u_p(n) = delta_u_p(n-1);
    end
    delta_u_p(1) = delta_u;

    % kolejno W1 i G1
    sendControls ([ 1 , 5 ] , [ 50 , int64(u(k)) ]);

    % aktualizacja wykresu
    set(h1, 'YData', u(1:k), 'XData', 1:k);
    set(h2, 'YData', y(1:k), 'XData', 1:k);
    set(h3, 'YData', yzad(1:k), 'XData', 1:k);
    drawnow;

    waitForNewIteration () ; % wait for new iteration
end
```

2.5. Strojenie algorytmów regulacji

2.5.1. Symulacja stanowiska grzewczo-chłodzącego

Zaproponowana trajektoria posiada dwa skoki wartości zadanej: w chwili $t = 0$ do $y_{zad} = 35$ oraz w chwili $t = 450$ do $y_{zad} = 40$, ostatnia wartość jest utrzymywana do $t = 900$, co oznacza, że jedno badanie zajmuje równo 15 minut. W celu przyspieszenia badania regulatorów na rzeczywistym obiekcie zdecydowano się wykorzystać pozyskaną w rozdziale 2.3 odpowiedź skokową do symulacji działania obiektu, aby usprawnić poszukiwania odpowiednich nastaw regulatorów.

```
K = 0.38024;
Td = 6;
T1 = 82.0474;
T2 = 3.3333;
[a1, a2, b1, b2] = parametry_rownania_roznicowego(K,Td,T1,T2);

% równanie różnicowe do symulacji obiektu
y(k) = Ypp -a1*(y(k-1)-Ypp)-a2*(y(k-2)-Ypp) + ...
        b1*(u(k-7)-Upp) + b2*(u(k-8)-Upp);
```

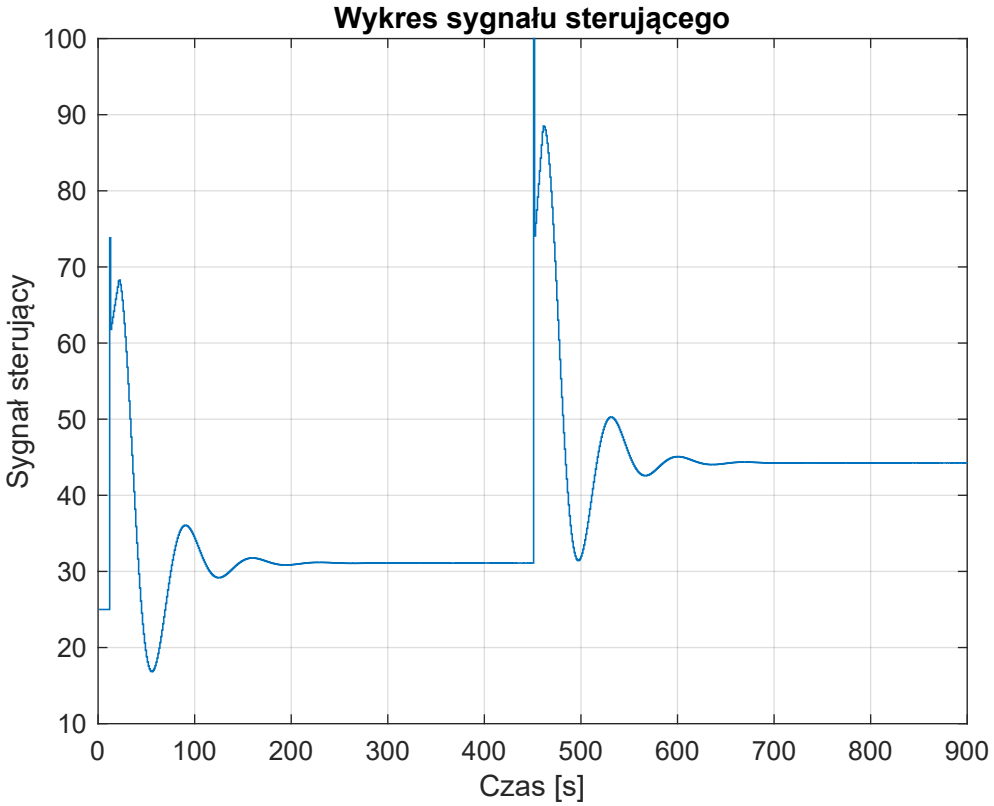
Tak przygotowane równanie różnicowe zostało wstawione do funkcji algorytmów, które szczegółowo zostały opisane w części projektowej. Skorzystano również z przygotowanych funkcji optymalizacji, aby w pełni wykorzystać możliwości przeprowadzenia symulacji.

2.5.2. Algorytm PID

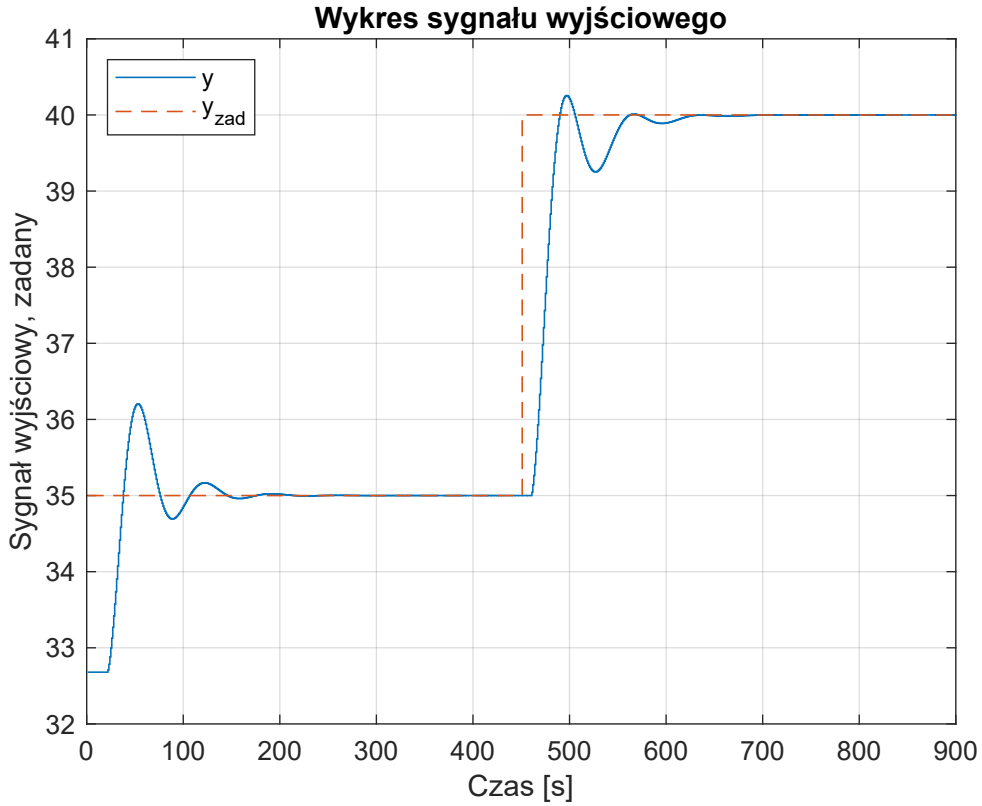
Optymalizacja regulatora PID z symulowanym obiektem wskazała na wartości parametrów podane w tab. 2.3, a wykresy działania przedstawione są na rys. 2.5 oraz rys. 2.6. Wykresy działania takiego regulatora wydają się być akceptowalne i zostaną przetestowane na rzeczywistym stanowisku.

Tab. 2.3. Parametry PID dobrane metodą optymalizacji

Parametr	Wartość
K	15,3505
T_i	45,0000
T_D	0,3602

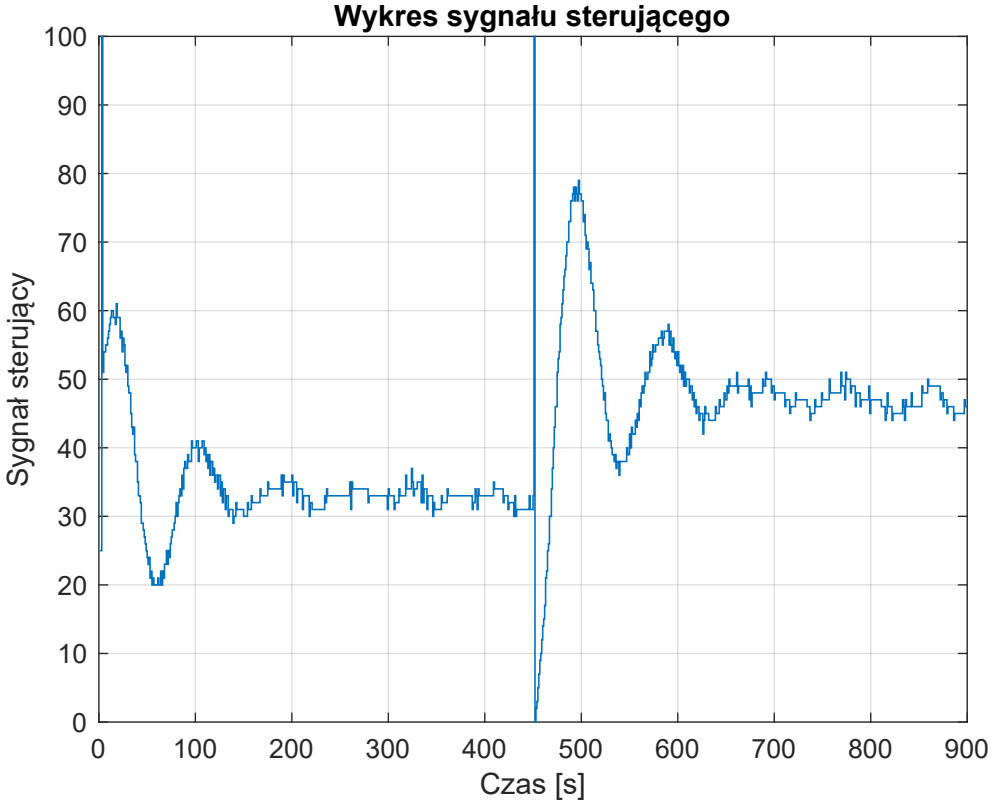


Rys. 2.5. Sygnał wejściowy symulowanego regulatora PID w odpowiedzi na dwie różne wartości zadane

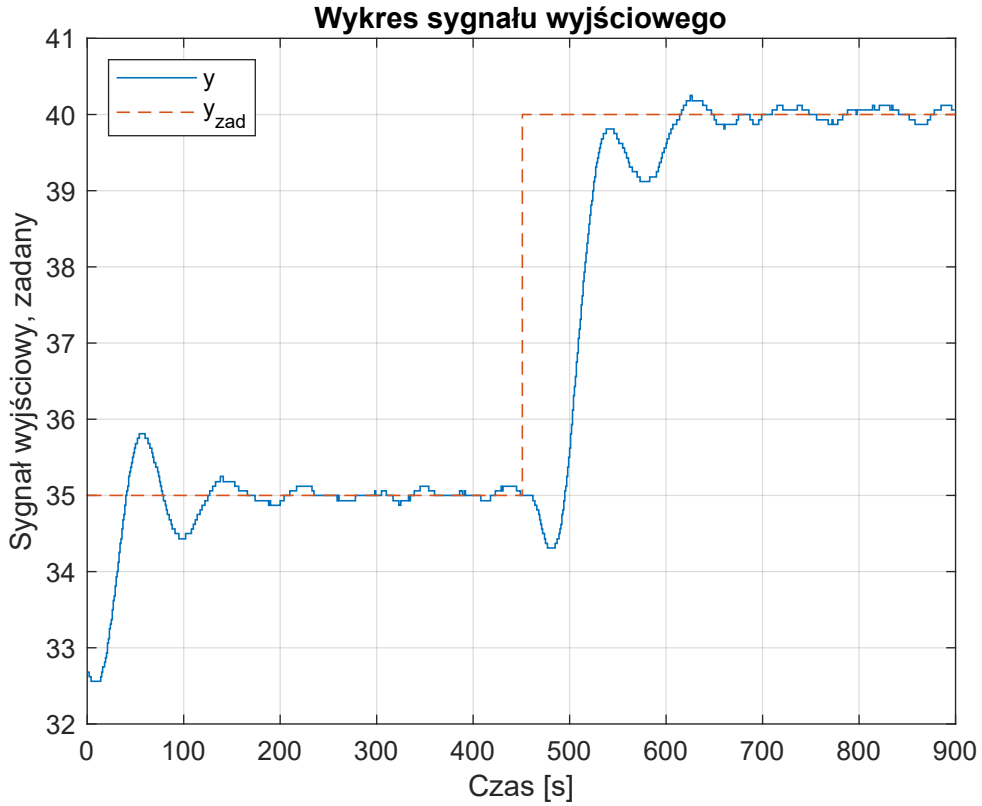


Rys. 2.6. Sygnał wyjściowy symulowanego regulatora PID w odpowiedzi na dwie różne wartości zadane

Oczywiście, spodziewano się, że wyliczone nastawy (tab. 2.3) nie będą idealne, rzeczywisty obiekt różni się od idealnego modelu, ponadto w modelu nie uwzględniono żadnych zakłóceń, jednak założono, że nastawy te będą co najmniej poprawne.



Rys. 2.7. Sygnał sterujący regulatora PID w odpowiedzi na dwie różne wartości zadane



Rys. 2.8. Sygnał wyjściowy regulatora PID w odpowiedzi na dwie różne wartości zadane

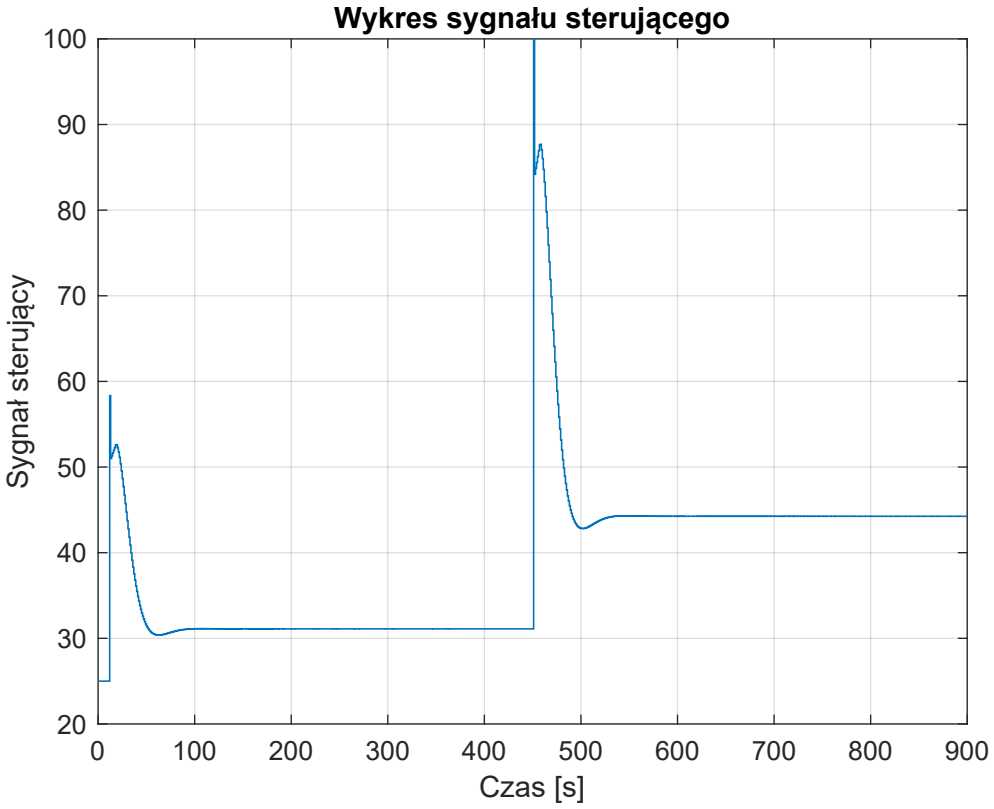
Widoczne są pewne oscylacje w trakcie pierwszych kilkudziesięciu sekund działania progra-

mu, jednak ustają one potem (widoczne skoki sygnału wyjściowego po stabilizacji wynoszą około +/- 0.2 względem wartości ustalonej i ich występowanie jest naturalne w przypadku tego obiektu, wynikają miedzy innymi z nieuwzględnionych zakłóceń oraz dokładności pomiaru). Niepożądanym jednak z pewnością jest nagły skok wartości sygnału sterującego widoczny po obu zmianach wartości zadanej. Wskazuje to na zbyt duży wpływ członu różniczkującego. Regulator PID działa dobrze z obiektami o małym bądź zerowym opóźnieniu, dlatego sygnał różniczkujący w tym przypadku ma negatywny wpływ. Dlatego też przy drugiej symulacji (rys. 2.9 i 2.10) przede wszystkim ten człon zmniejszono znacznie, jednak wpływ każdego członu zmniejszono delikatnie, w oparciu o przeprowadzone symulacje na modelu obiektu. Ostateczne nastawy wynosiły:

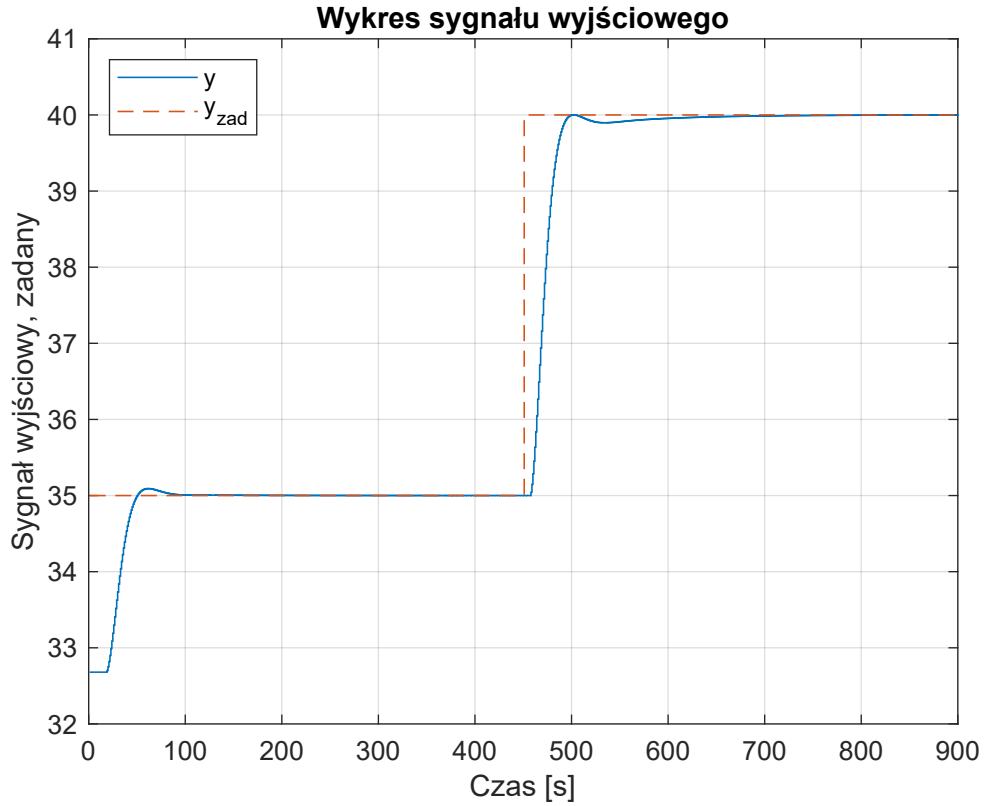
Tab. 2.4. Ostateczne parametry PID

Parametr	Wartość
K	11,0
T_i	80,0
T_D	0,3

Wyniki symulacji z nastawami z tab. 2.4:

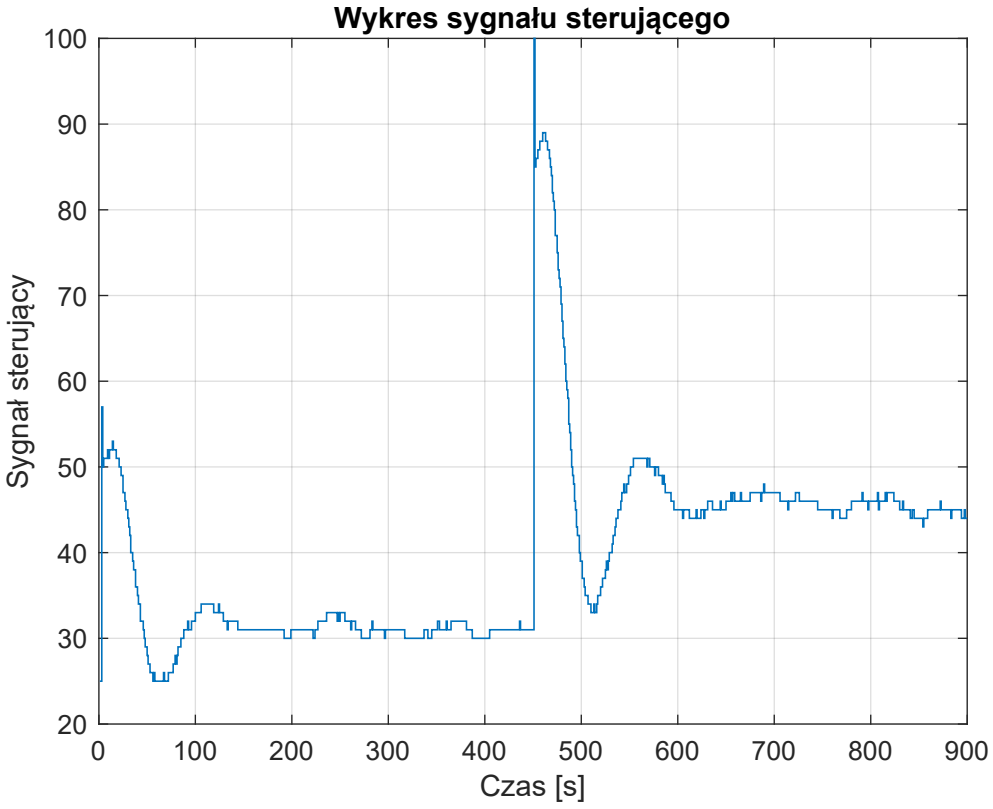


Rys. 2.9. Sygnał wejściowy symulowanego regulatora PID w odpowiedzi na dwie różne wartości zadane

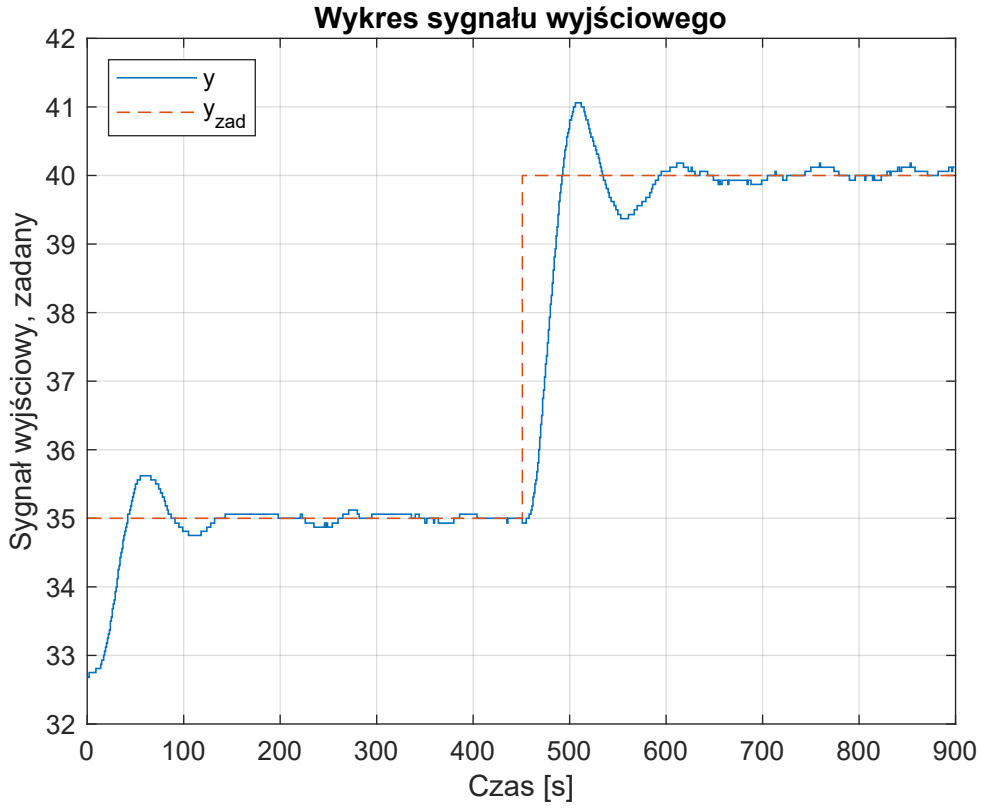


Rys. 2.10. Sygnał wyjściowy symulowanego regulatora PID w odpowiedzi na dwie różne wartości zadane

Tak prezentują się wyniki regulacji z dobranymi nastawami z tab. 2.4:



Rys. 2.11. Ostateczny sygnał sterujący regulatora PID w odpowiedzi na dwie różne wartości zadane



Rys. 2.12. Ostateczny sygnał wyjściowy regulatora PID w odpowiedzi na dwie różne wartości zadane

Dobrane w ten sposób nastawy pozwoliły uzyskać:

— Mniejsze przeregulowanie
— Mniejsze oscylacje i szybszą stabilizacje wokół wartości zadanej
— Brak nienaturalnego skoku sygnału sterującego z wartości maksymalnej do minimalnej. Co prawda nagły skok w górę przy zmianie wartości zadanej dalej występuje, ale jest on tylko chwilowy i sygnał nie odbija w przeciwnym kierunku do zmiany wartości zadanej.

Tab. 2.5. Wskaźnik jakości regulatorów PID

Regulator	Pierwszy	Poprawiony
Symulowany	585,01	556,09
Rzeczywisty	1782,76	715,88

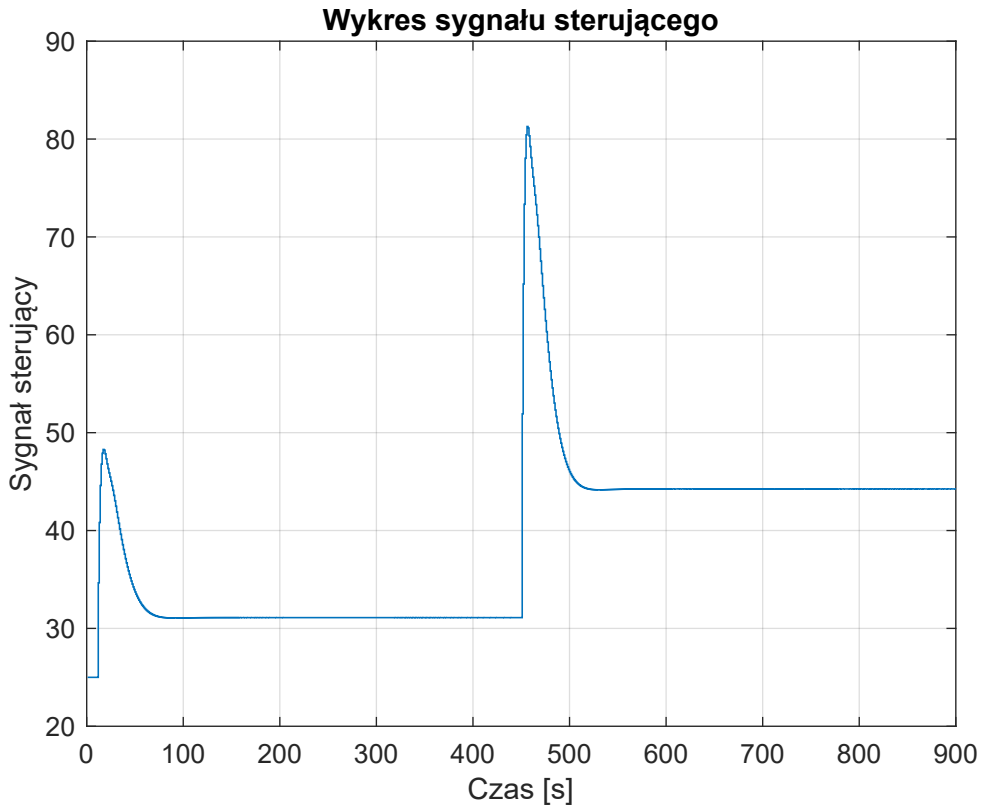
Patrząc sumę błędów kwadratowych w tab. 2.8, widać poprawę regulatorów z nowymi nastawami. Potwierdza to również, że symulacja daleka jest od rzeczywistego obiektu, ale jest pomocna w przyspieszeniu strojenia regulatorów.

2.5.3. Algorytm DMC

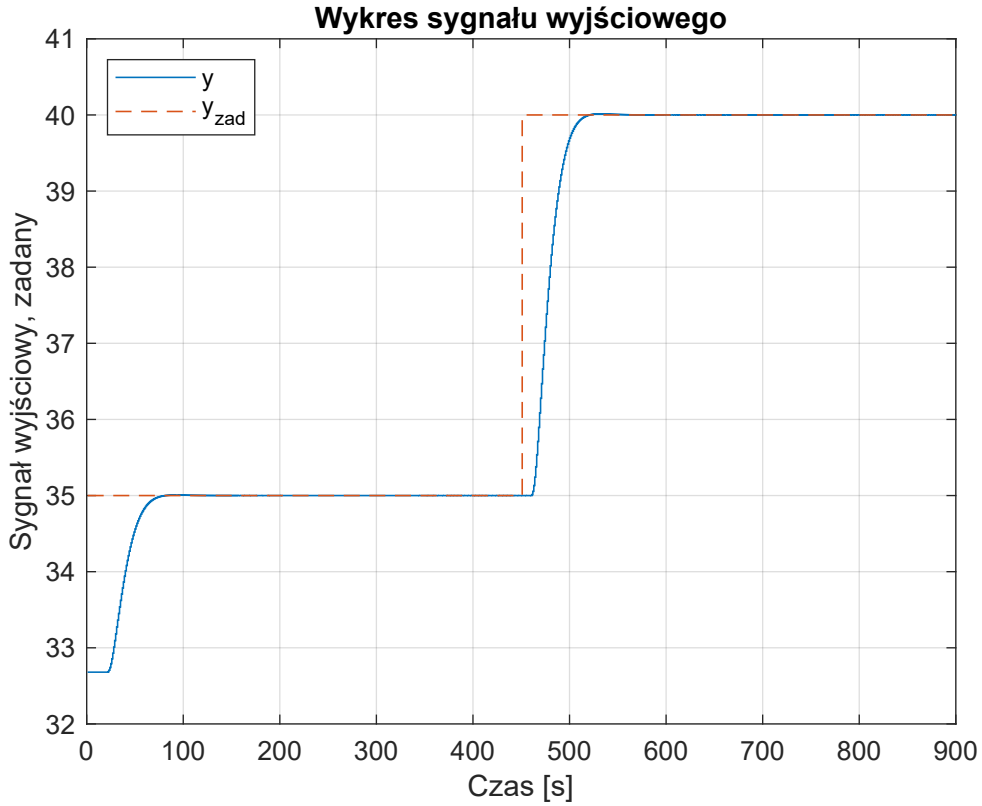
Analogicznie jak w przypadku pracy z regulatorem PID: początkowe nastawy zostały dobrane symulacyjnie z użyciem optymalizacji jako bazę do rozpoczęcia pracy z obiektem rzeczywistym. W tabeli 2.6 podano wyliczone nastawy oraz następnie przedstawiono przebiegi sygnałów symulowanych oraz rzeczywistych w odpowiedzi na dwa skoki wartości zadanej, kolejno do $y_{zad} = 35$ oraz $y_{zad} = 40$.

Tab. 2.6. Parametry DMC dobrane metodą optymalizacji

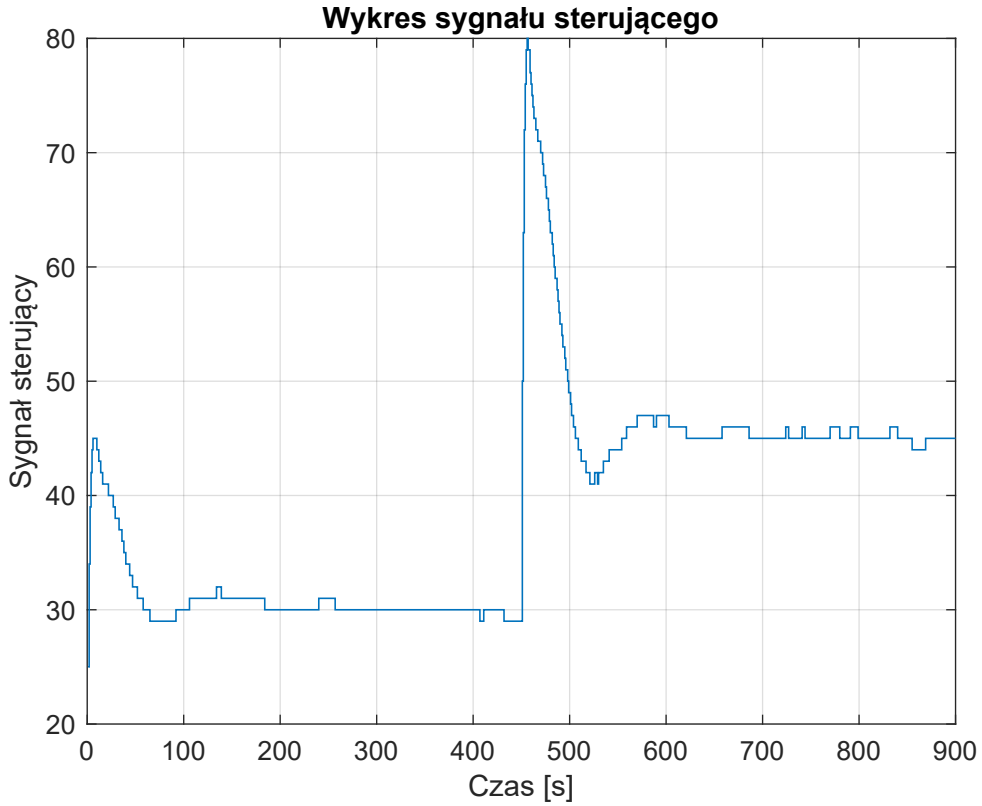
Parametr	Wartość
N	400
N_u	3
D	700
λ	0,05



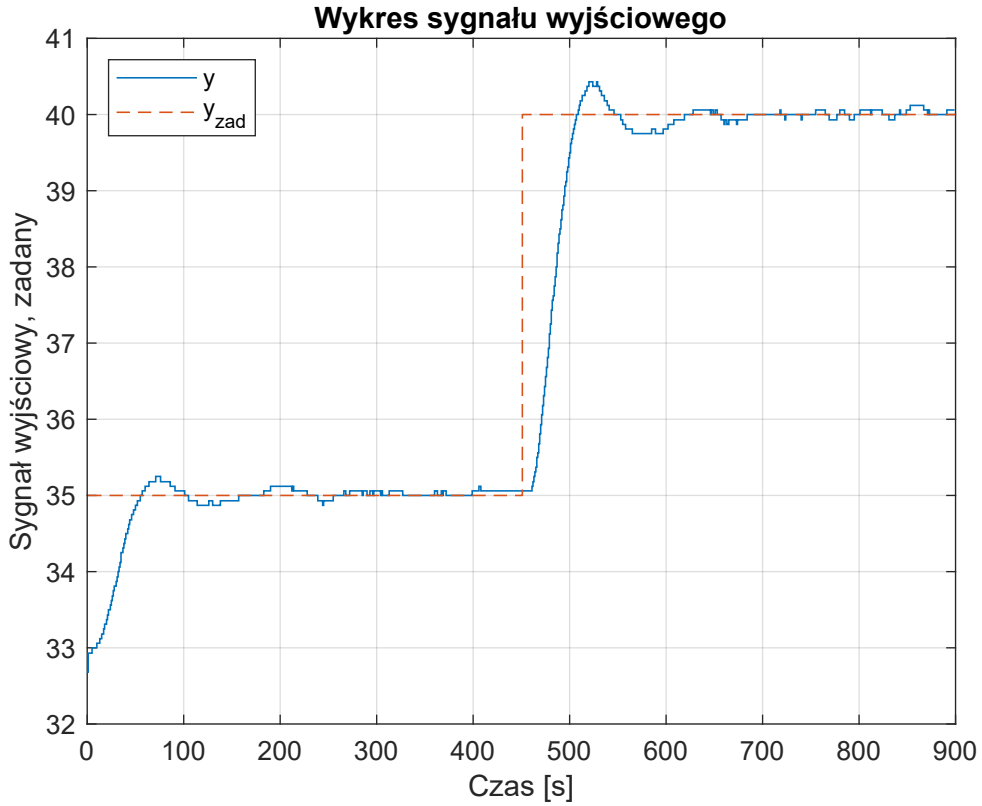
Rys. 2.13. Sygnał wejściowy symulowanego regulatora DMC w odpowiedzi na dwie różne wartości zadane



Rys. 2.14. Sygnał wyjściowy symulowanego regulatora DMC w odpowiedzi na dwie różne wartości zadane



Rys. 2.15. Sygnał sterujący regulatora DMC w odpowiedzi na dwie różne wartości zadane

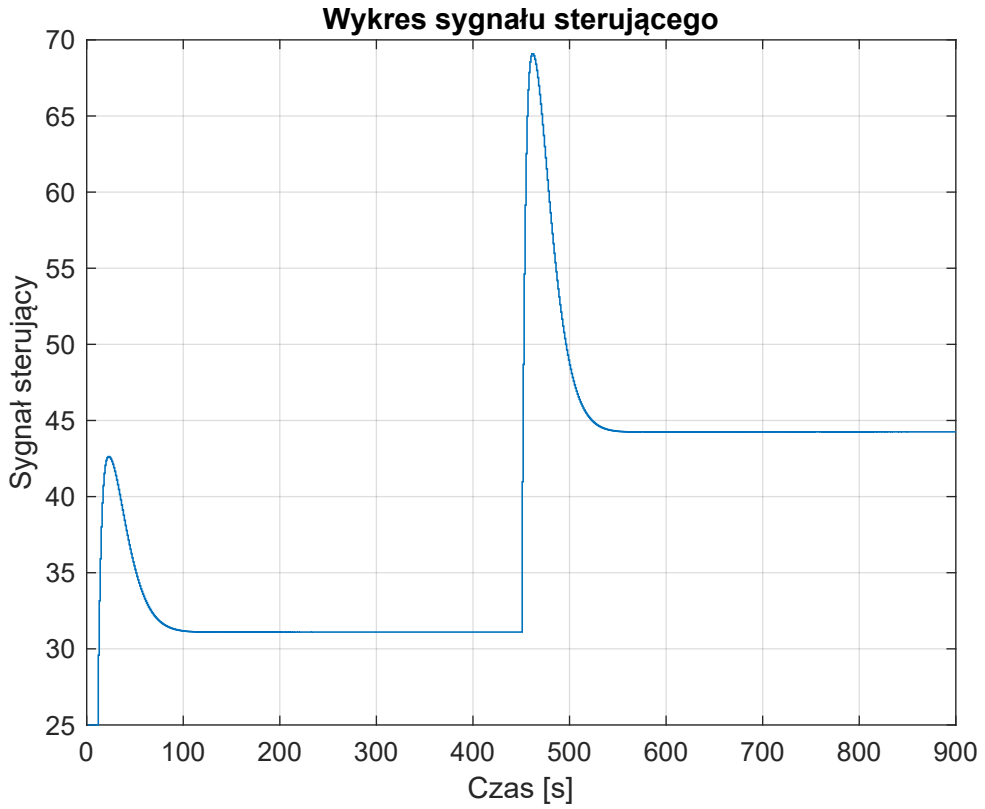


Rys. 2.16. Sygnał wyjściowy regulatora DMC w odpowiedzi na dwie różne wartości zadane

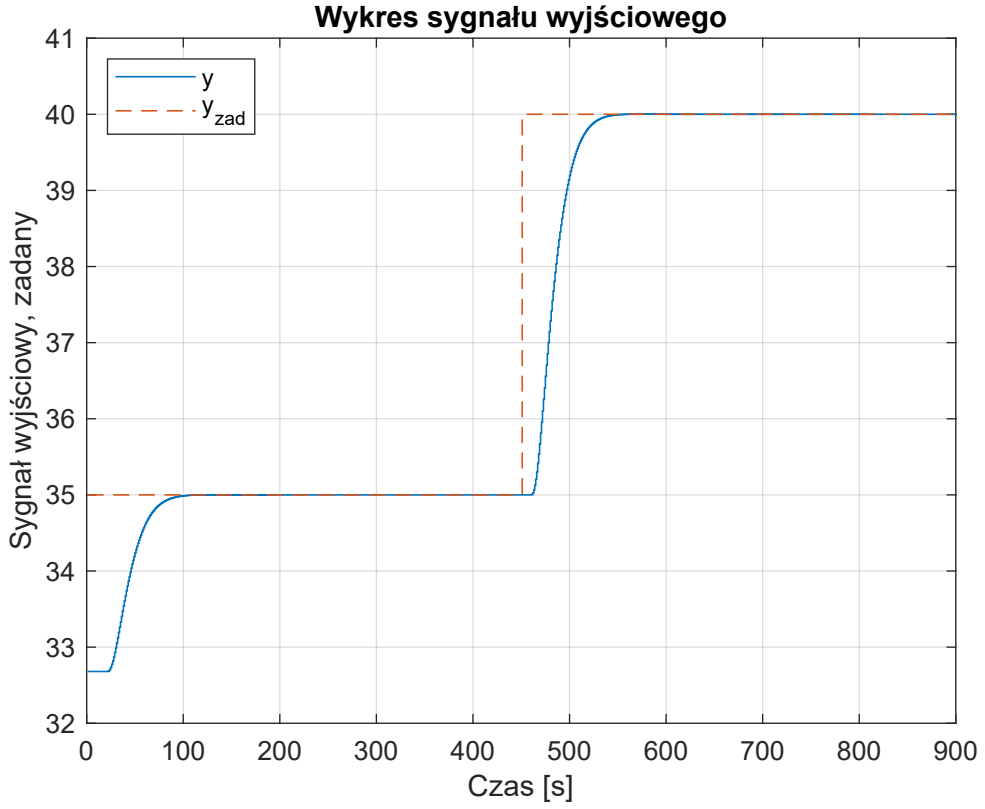
Widoczne są lekkie oscylacje oraz przeregulowanie, jednak już w tym momencie można uznać pracę regulatora za zadowalającą. Nie występuje uchyb ustalony. Regulator też poradził sobie gorzej, w kontekście powstałego przeregulowania i większego błędu E, w przypadku drugiego skoku. Na wykresie przebiegu sygnału sterującego widać nagły i duży skok, co jest wskazane, jeśli zależy nam na szybkości regulacji, jednak po nim widoczna też jest dolina - przez zauważalne przeregulowanie regulator musiał "odbić", co stworzyło niepożądane oscylacje na wyjściu.
Aby poprawić jakość regulacji zmniejszyliśmy wartość parametru N, aby ograniczyć wpływ przyszłych wartości sygnałów na jego działanie, co powinno zmniejszyć nagłe skoki sygnału sterującego. Zwiększyliśmy także parametry N_u - w celu zwiększenia wpływu przeszłych sterowań oraz λ - w celu zwiększenia kary za nagłe przyrosty sterowania, co powinno pomóc w przypadku wyżej opisanej "doliny" po nagłym skoku sygnału sterującego. Naturalnie parametr D - a więc parametr wynikający z modelu, określający po jakim czasie model się ustabilizował, został niezmieniony. Poniżej znajduje się tabela z ostatecznymi parametrami oraz przebiegi sygnałów z dobranymi nastawami:

Tab. 2.7. Ostateczne parametry regulatora DMC

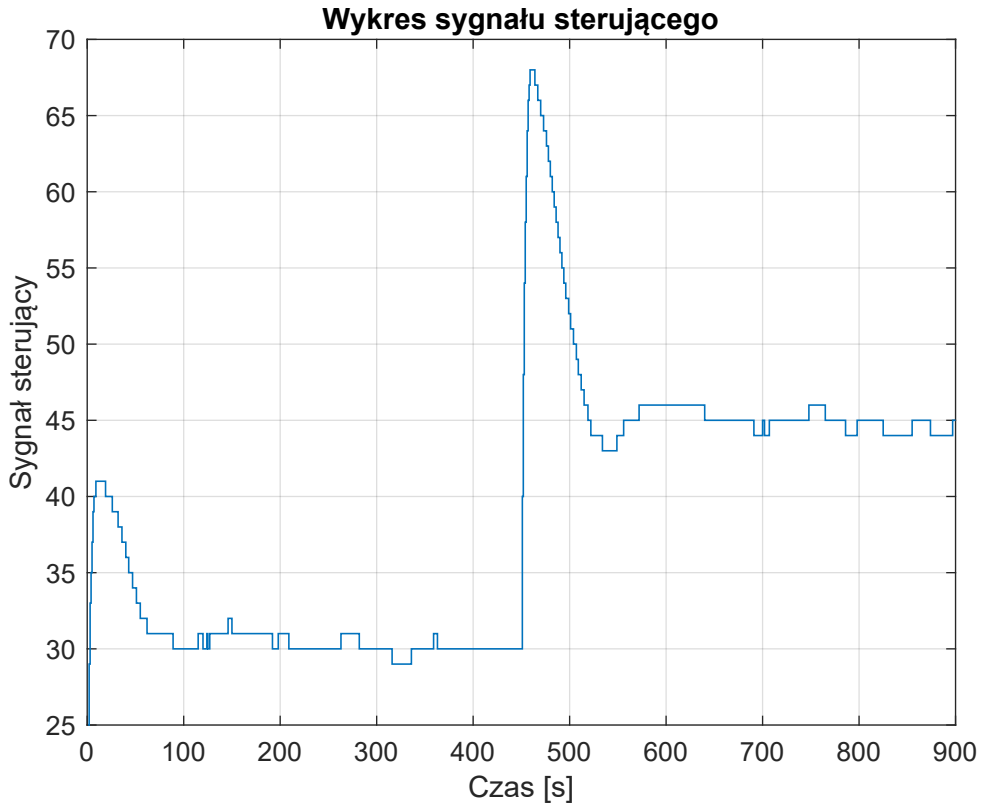
Parametr	Wartość
N	200
N_u	5
D	700
λ	0,2



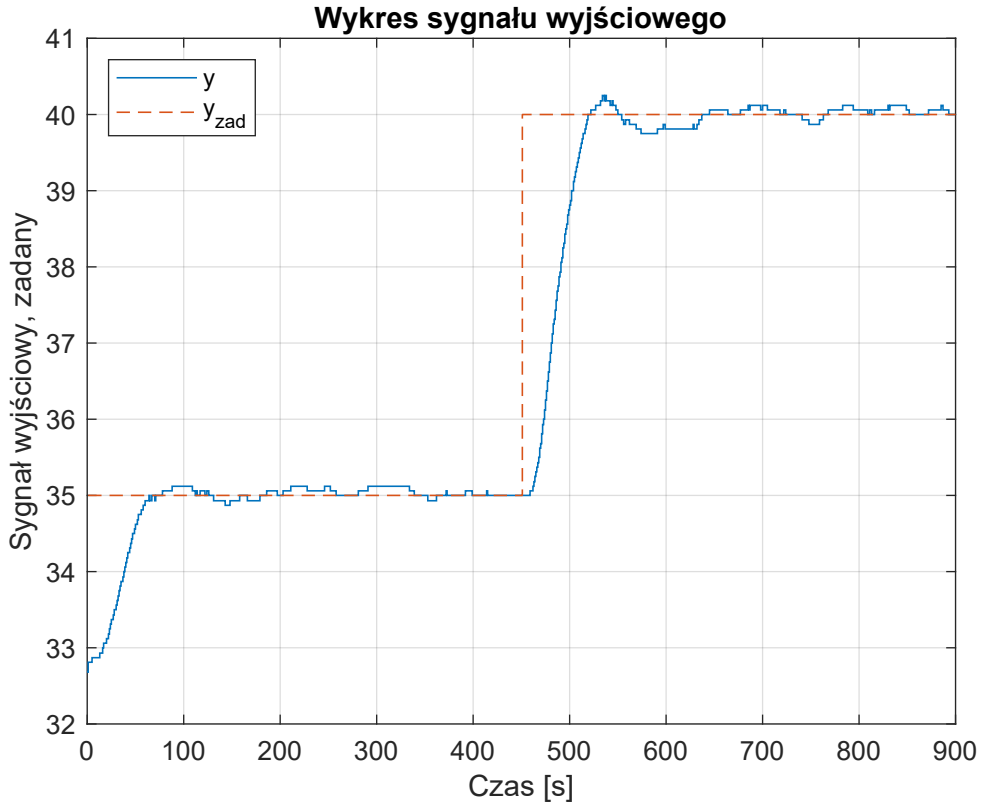
Rys. 2.17. Sygnał wejściowy symulowanego regulatora DMC w odpowiedzi na dwie różne wartości zadane



Rys. 2.18. Sygnał wyjściowy symulowanego regulatora DMC w odpowiedzi na dwie różne wartości zadane



Rys. 2.19. Ostateczny sygnał sterujący regulatora DMC w odpowiedzi na dwie różne wartości zadane



Rys. 2.20. Ostateczny sygnał wyjściowy regulatora DMC w odpowiedzi na dwie różne wartości zadane

W ten sposób uzyskaliśmy przede wszystkim mniejsze przeregulowanie, jak widać jest ono

pomijalnie małe. Dzięki temu sygnał szybciej się stabilizował. Ponadto sygnał sterujący był także bardziej stabilny - nie był poddawany aż tak nagłym skokom. Ostatecznie końcowe nastawy pozwoliły na dosyć szybką regulację bez uchybu oraz widocznego przeregulowania.

Tab. 2.8. Wskaźnik jakości regulatorów DMC

Regulator	Pierwszy	Poprawiony
Symulowany	726,98	843,05
Rzeczywisty	754,84	866,45

Porównując błędy widać pogorszenie współczynnika jakości po poprawieniu parametrów regulatora. Spowodowane jest to chęcią poprawienia stabilności sygnału wejściowego kosztem szybkości regulacji. Warto podkreślić tu fakt, że błędy symulowanych oraz rzeczywistych regulatorów są porównywalne. Również sygnały sterujące (symulowane i rzeczywiste) były do siebie bardzo podobne zarówno kształtami jak i wartościami.

2.6. Wnioski

Algorytm PID był zdecydowanie prostszy w implementacji, ale jego strojenie wymagało czasu, który na laboratorium był towarem deficytowym. DMC dzięki swojej idei stroi się samoczynnie, projektant ma wpływ jedynie na λ oraz złożoność obliczeniową poprzez określenie horyzontów. Wykresy regulatora DMC wyróżnia smukły i umiarkowany przebieg sygnału sterującego, który w porównaniu do wykresów regulatora PID jest gwałtowny i szybkozmienny.