Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Systemy Mikroprocesorowe w Sterowaniu

Sprawozdanie z projektu 1

Michał Kwarciński, Kacper Marchlewicz

Warszawa, 2022

Spis treści

1. Projekt 1 2
1.1. Regulator PID 2
1.2. DMC 5
1.3. Podsumowanie 11

1. Projekt 1

1.1. Regulator PID

Regulator PID został zaimplementowany na podstawie wzoru:

$$u(k) = u_p(k) + u_I(k) + u_D(k) \tag{1.1}$$

gdzie:

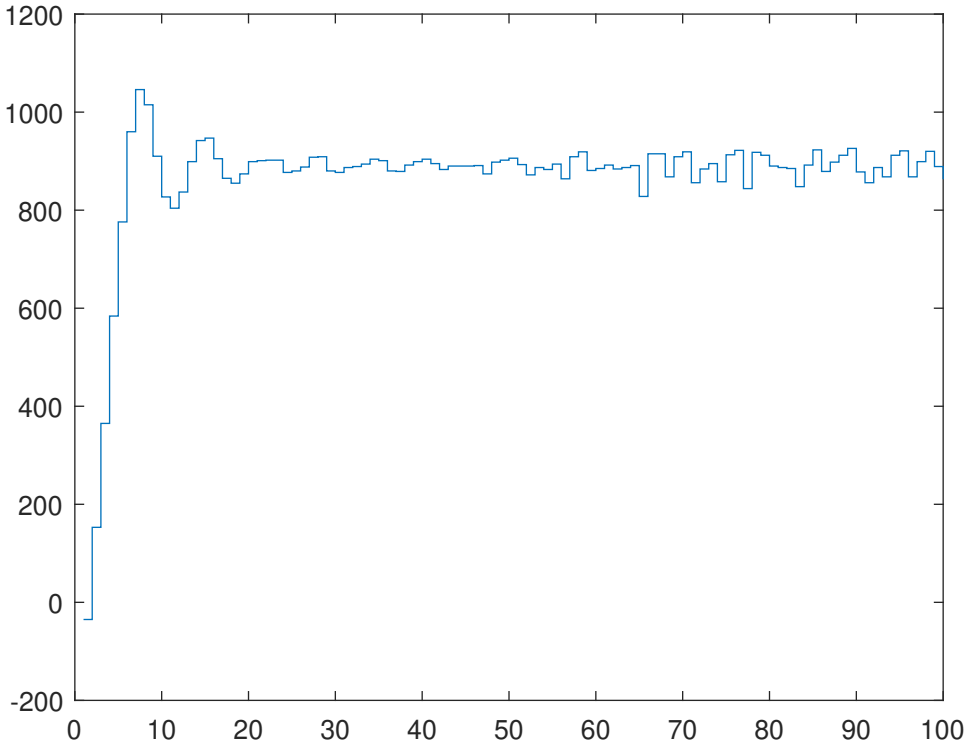
$u_P(k) = Ke(k)$
$u_I(k) = u_I(k-1) + \frac{T}{T_I}\frac{e(k-1)+e(k)}{2}$
$u_D(k) = KT_D\frac{e(k)+e(k-1)}{T}$

Potrzebne dla metod strojenia, początkowe parametry K=17 i T_u=0.6 zostały obliczone poprzez wprowadzenie układu w oscylacje niegasnące. Następnie właściwe parametry K, T_i, T_d zostały wyznaczone następującymi metodami:

1.Metoda Zieglera-Nicholsa

Polega ona na wyznaczeniu parametrów zgodnie z tabelą podaną w skrypcie. Wynoszą one: K=17, T_I=0,3; T_D=0,075. Otrzymany przebieg wyjścia (1.1) dla obiektu regulowanego tymi nastawami nie są zadowalające, lecz stanowią dobry punkt odniesienia dla następnej metody strojenia.

2.Metoda inżynierska



Rys. 1.1. Wyjście regulatora PID wyznaczonego metodą Zieglera–Nicholsa

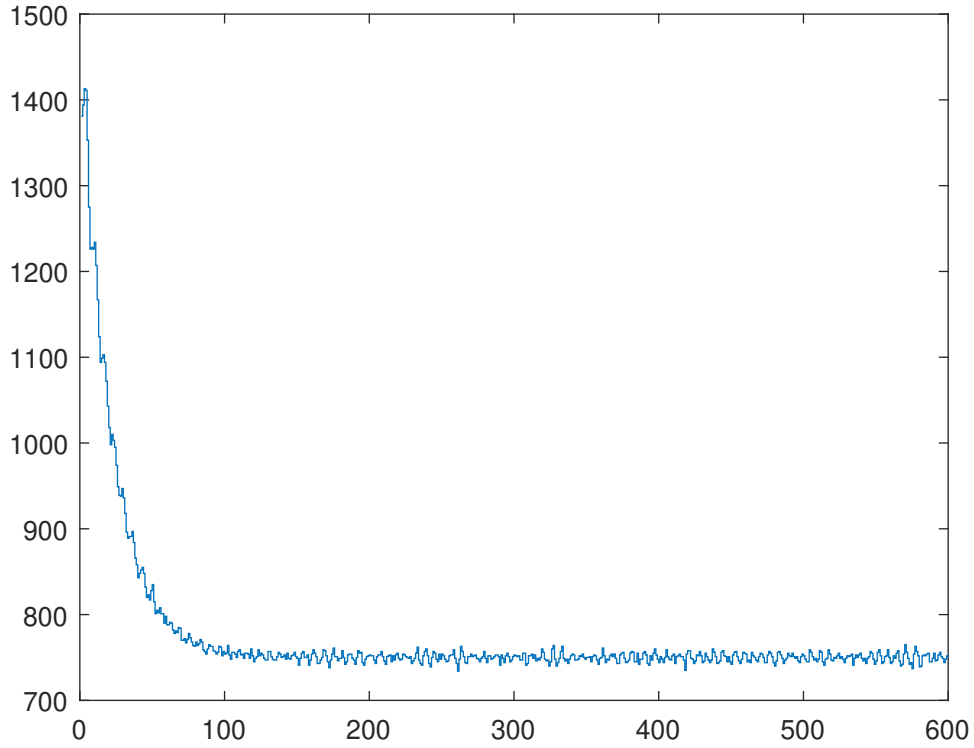
Należy przejść do doboru nastaw regulatora PI, gdzie $K = 0{,}5K_u$ natomiast parametr T_I należy dobrać metodą prób i błędów tak, aby uzyskać jak najlepsze rezultaty. Gdy osiągnięty zostanie już satysfakcjonujący czas zdwojenia, należy włączyć ostatni człon – różniczkujący. Tak samo jak dla członu całkującego należy metodą prób i błędów dobierać wartości T_D tak, aby zminimalizować wybrany wskaźnik jakości. Osiągnięcie satysfakcjonujących wyników kończy procedurę strojenia. Otrzymane nastawy wynoszą: K=8,5; T_I=0,8; T_D=0,02. Otrzymany przebieg wyjścia (1.2) potwierdza poprawność metody. Regulator lepiej trzyma się wartości zadanej, dzięki czemu jest lepszy od regulatora wyznaczonego metodą Zieglera-Nicholsa.

Algorytm anti-windup

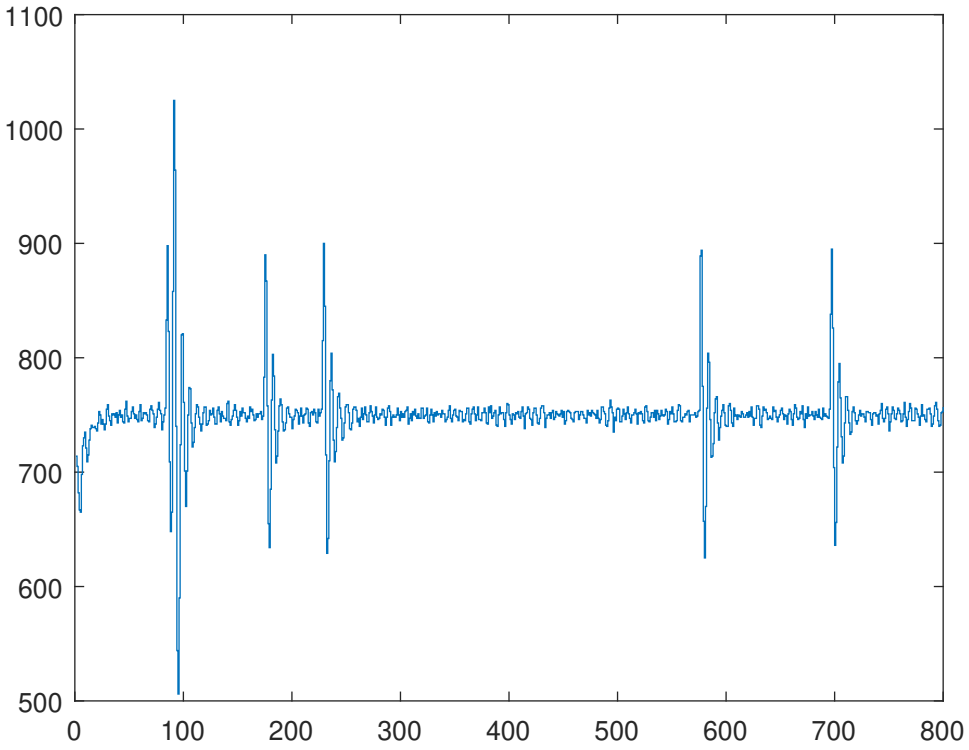
Jest to algorytm mający na celu niwelację wpływu zjawiska przesadnego całkowania, nazywanego nawijaniem (windup). Realizowany jest poprzez wprowadzenie do członu całkującego dodatkowego elementu proporcjonalnego:

$$u_I(k) = u_I(k-1) + \frac{K}{T_I} T \frac{e(k-1) + e(k)}{2} + \frac{T}{T_v}(u_w(k-1) - u(k-1)) \tag{1.2}$$

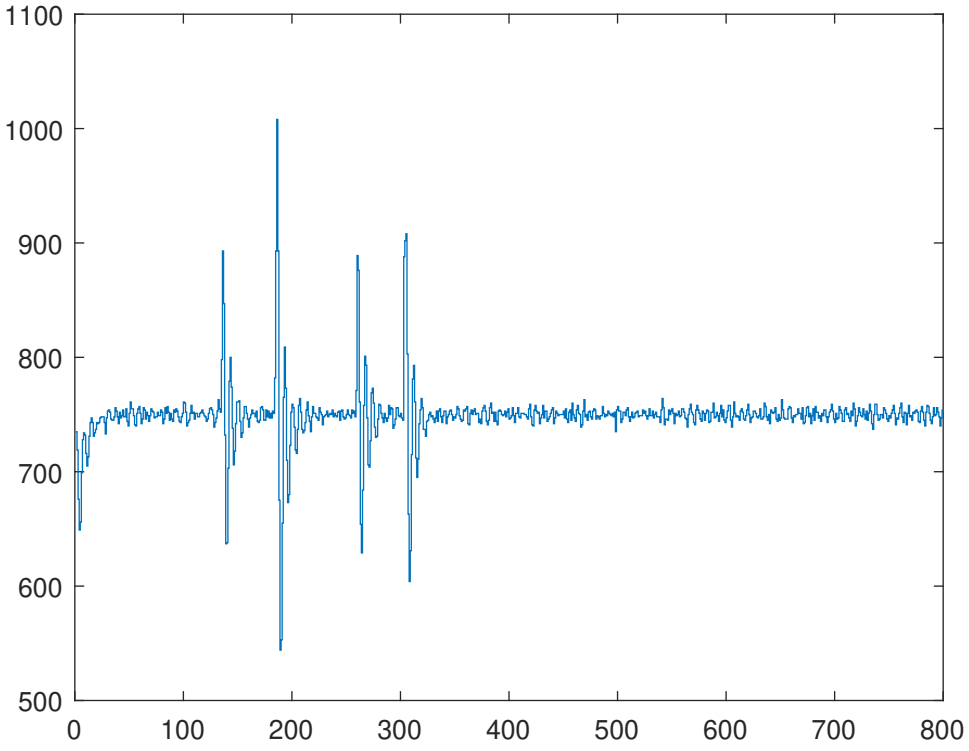
T_w jest to parametr algorytmu, a sterowanie $u_w(k-1)$ to sterowanie które zostało faktycznie zaaplikowane do procesu. Przebieg wyjścia regulatora pod wpływem zakłóceń dla parametru T_v=1 przedstawia wykres 1.3, a dla T_v=0,1 wykres 1.4.



Rys. 1.2. Wyjście regulatora PID wyznaczonego metodą inżynierską



Rys. 1.3. Wyjście regulatora PID dla Tv=1



Rys. 1.4. Wyjście regulatora PID dla Tv=0,1

1.2. DMC

Na początku zaczęliśmy od otrzymania odpowiedzi skokowej. Na mikrokontrolerze zmienialiśmy cyklicznie wartość sterowania pomiędzy 0 a 1000 i mniej więcej dwa takie cykle zapisaliśmy w Matlabie. Potem otrzymane dane przycięliśmy do przedziału zawierającego odpowiedź (koło 50 pomiarów) i przeskalowaliśmy korzystając ze wzoru:

$$s = \frac{y(X : X + 50) - y(X - 1)}{1000} \tag{1.3}$$

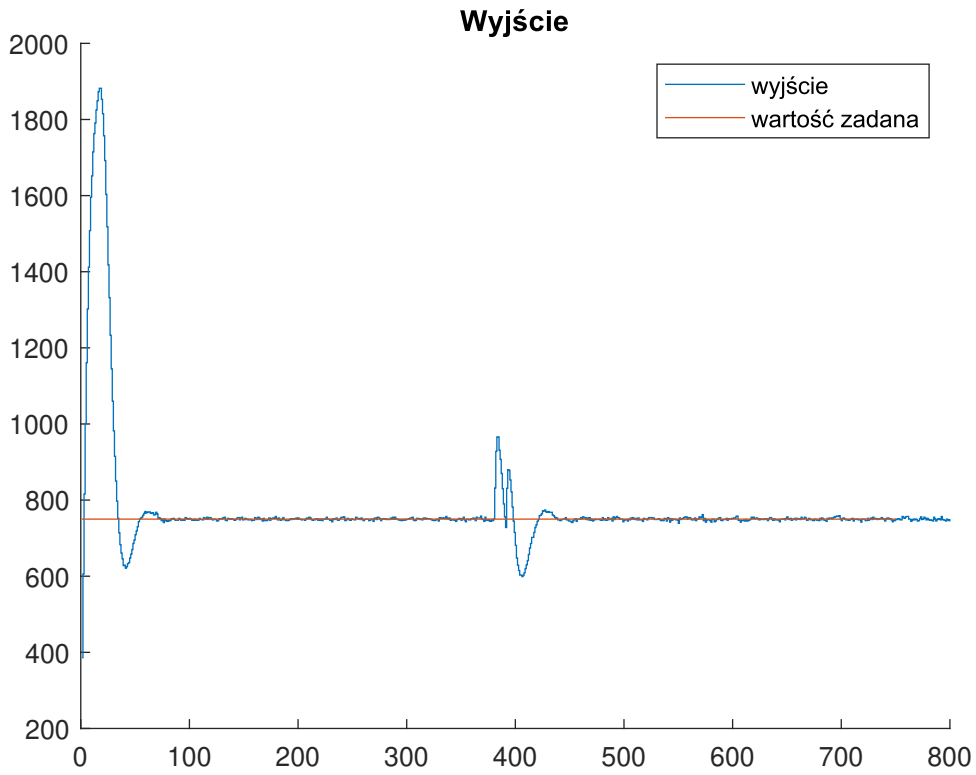
gdzie:

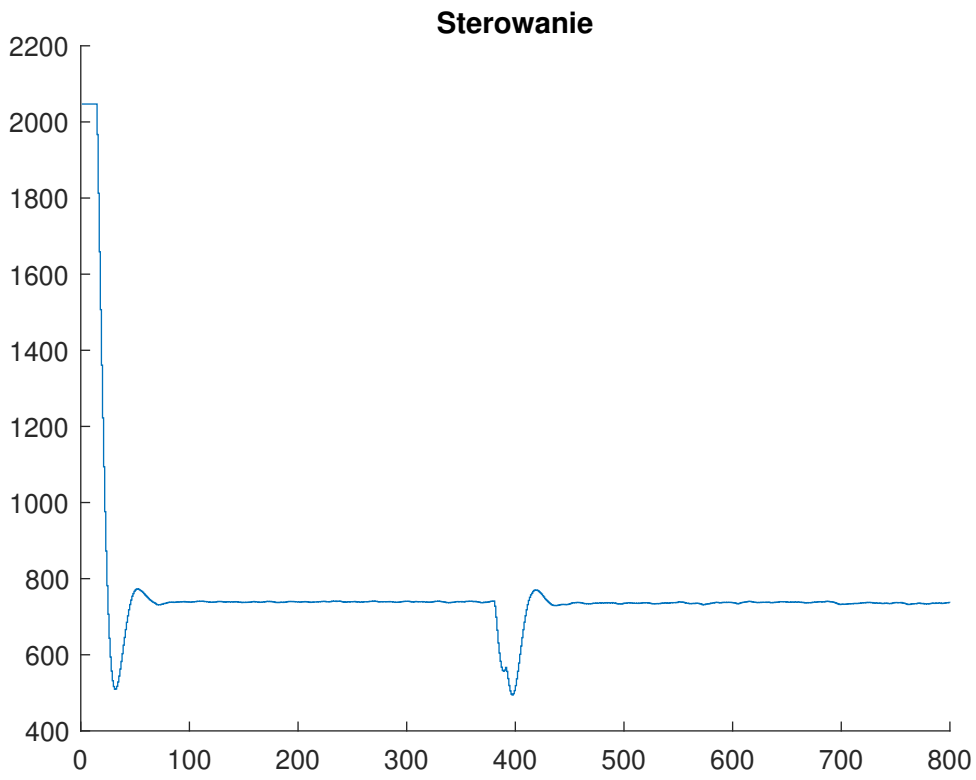
s - przeskalowana odpowiedź skokowa.
X - wartość, od której zaczyna się odpowiedź skokowa.
y - odpowiedź skokowa przed przeskalowaniem.

Na podstawie przeskalowanej odpowiedzi w Matlabie i parametrów D, λ, N i N_u stworzyliśmy skrypt w którym obliczamy macierze K (M i λ*I) i M_p, dzięki którym otrzymujemy parametr K_e i macierz K_u, które przepisujemy do kodu wgrywanego do mikrokontrolera. W kodzie w C, stworzyliśmy funkcje do obsługi algorytmu DMC, w której liczymy przyrost sterowania w danej chwili i wektor DU_p, z którego pierwszy element jest szukanym przyrostem sterowania. Na koniec do poprzedniego sterowania dodajemy otrzymany przyrost i sprawdzamy czy mieści się ono w ograniczeniach, jeśli nie to obcinamy do ograniczeń.

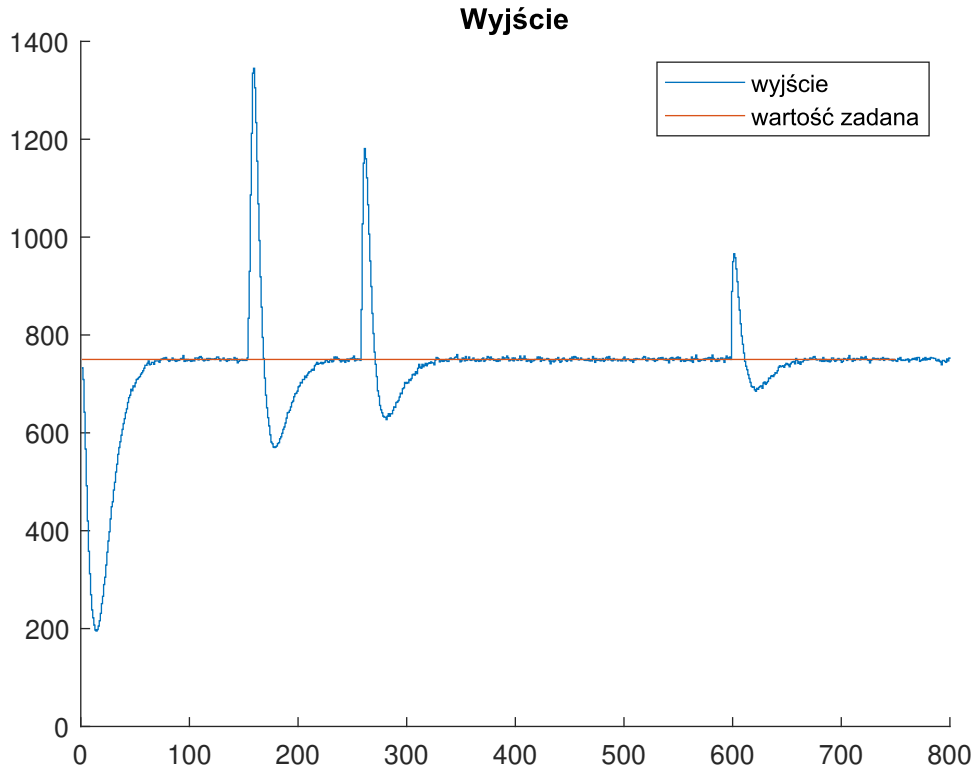
Wyniki:

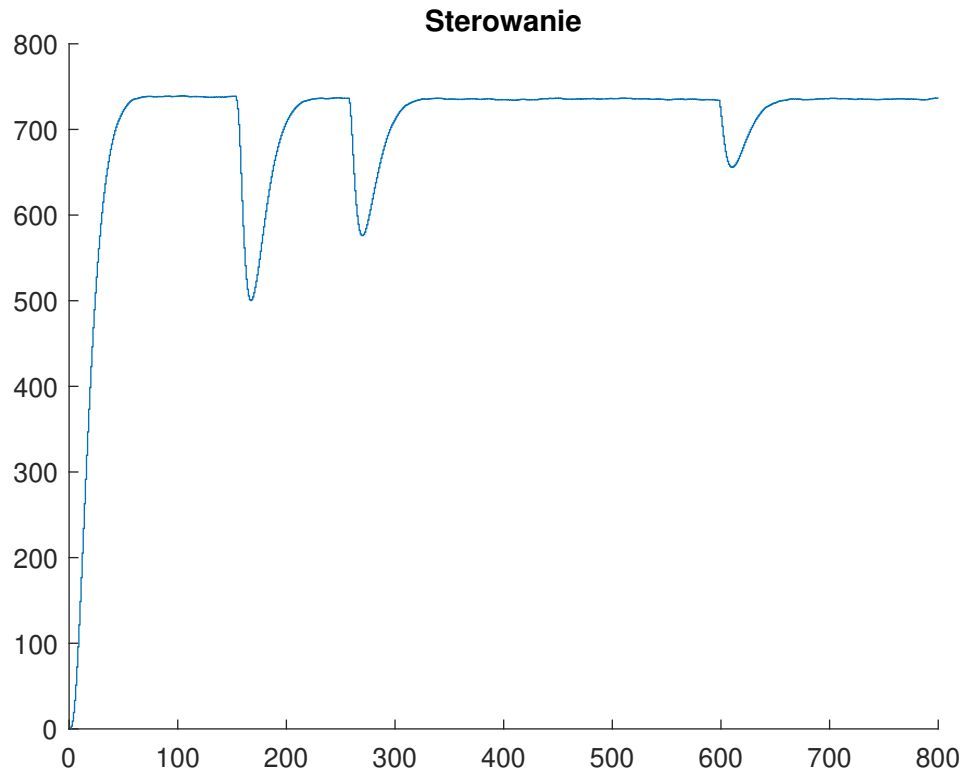
Na podstawie odpowiedzi skokowej założyliśmy, że za horyzont dynamiki przyjmiemy wartość 30. Widoczne skoki, po początkowym ustabilizowaniu się wyjścia, to efekt testowania zakłócenia. Ustawiliśmy bazowe parametry: $N = 2$, $N_u = 10$ i $\lambda = 1$. Otrzymaliśmy:



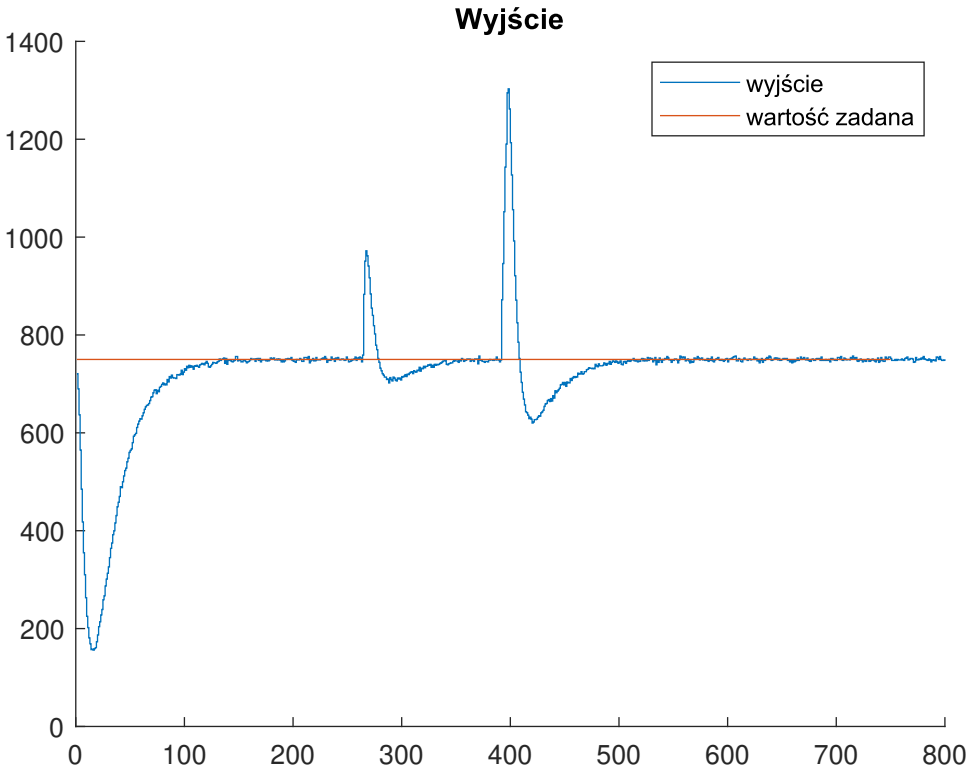


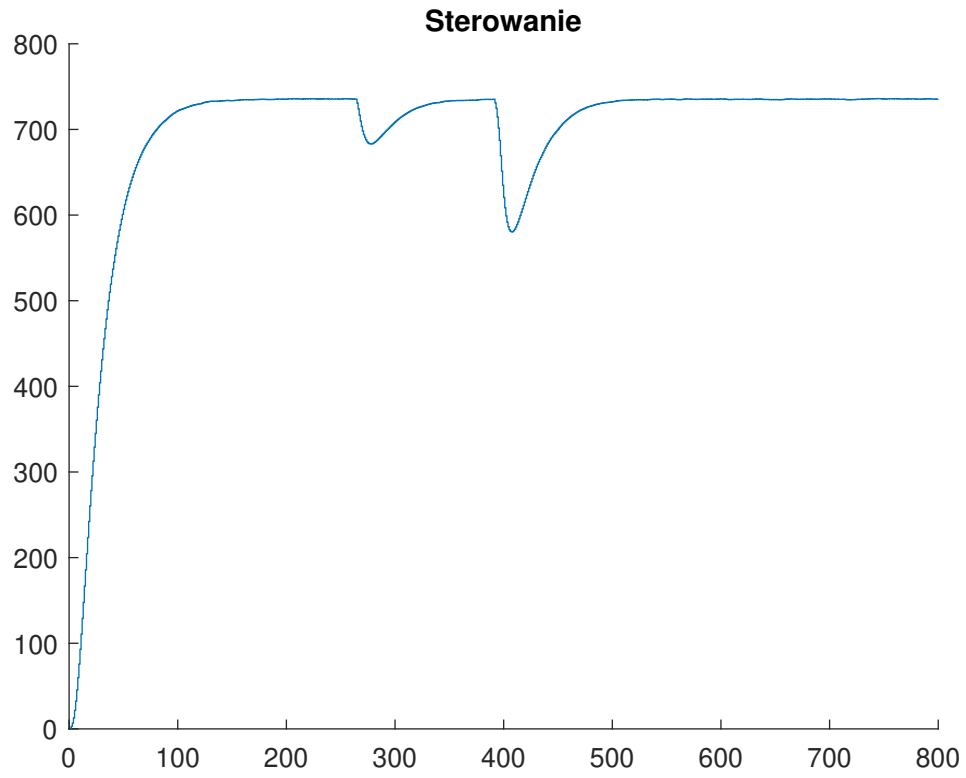
Widać niewielkie przeregulowanie. Dla zmiany parametru $\lambda = 2$:





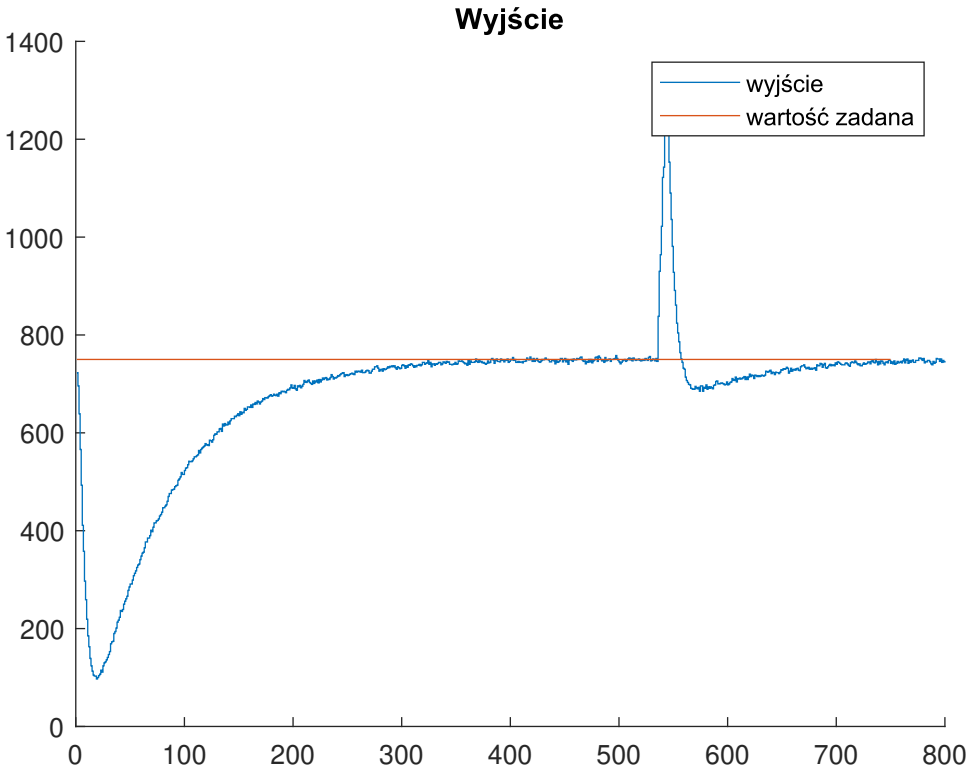
Przeregulowanie już nie występuje. Dla zmiany parametru $\lambda = 5$:

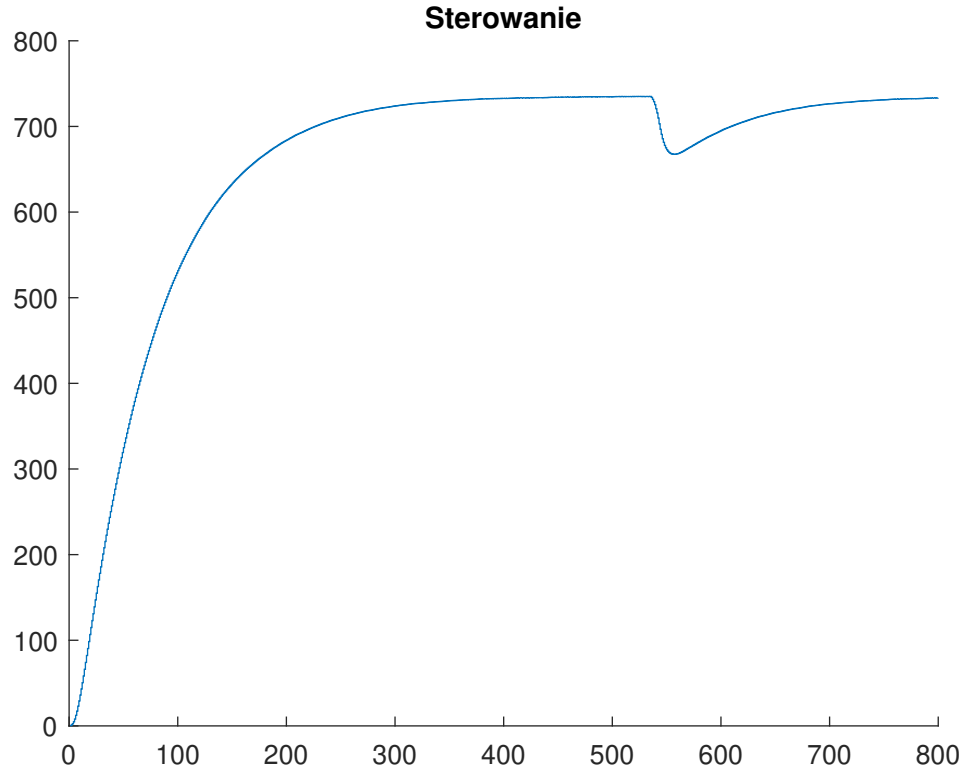




Można zauważyć widoczne wydłużenie czasu, który jest potrzebny, żeby doprowadzić obiekt do wartości zadanej. Zauważyliśmy, że optymalnym parametrem będzie $\lambda = 2$.

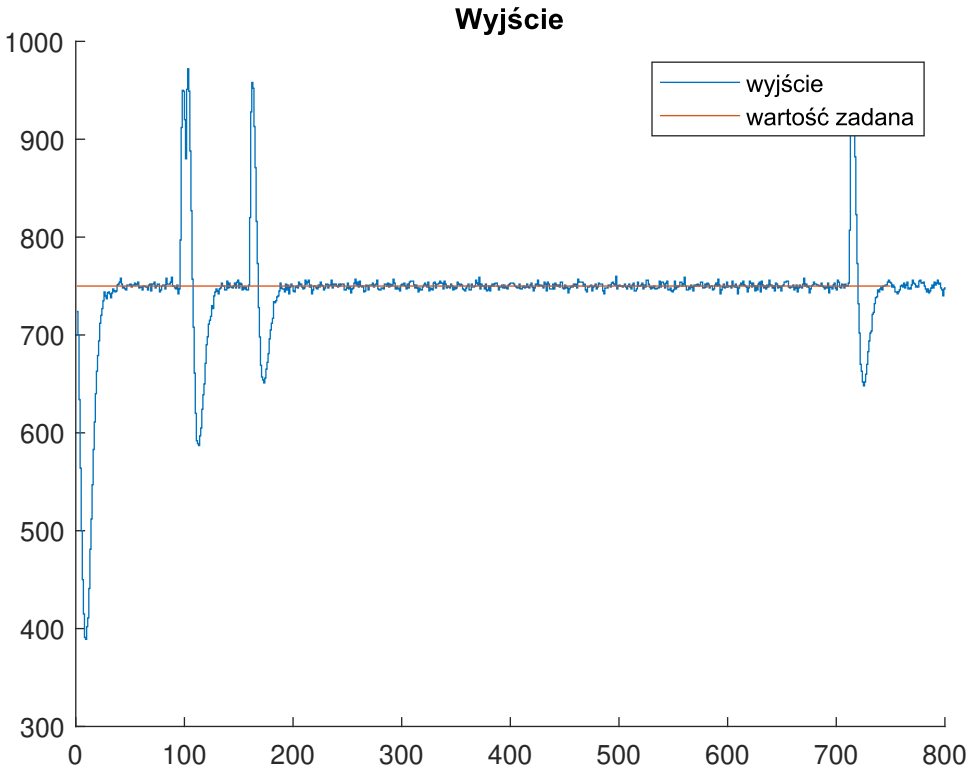
Dla zmiany parametru $N = 1$:

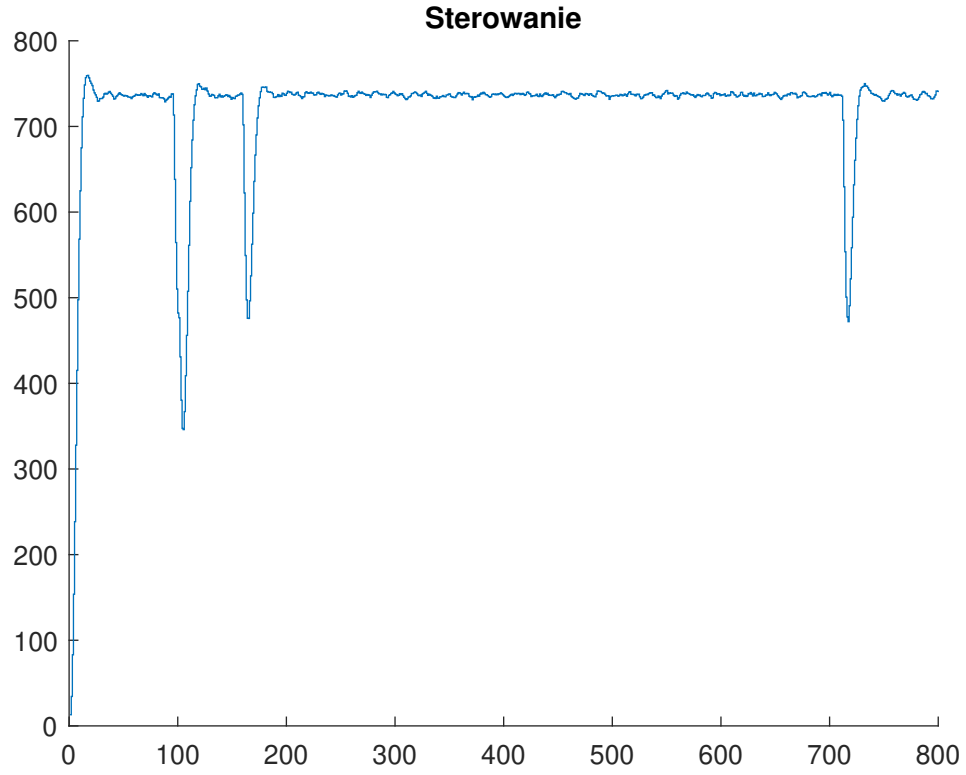




Widać zdecydowany spadek jakości regulacji. Nastąpiło duży wzrost czasu osiągnięcia wartości zadanej i przy powrocie z zakłóceń widoczne jest przeregulowanie. Widać, że horyzont predykcji ma za małą wartość.

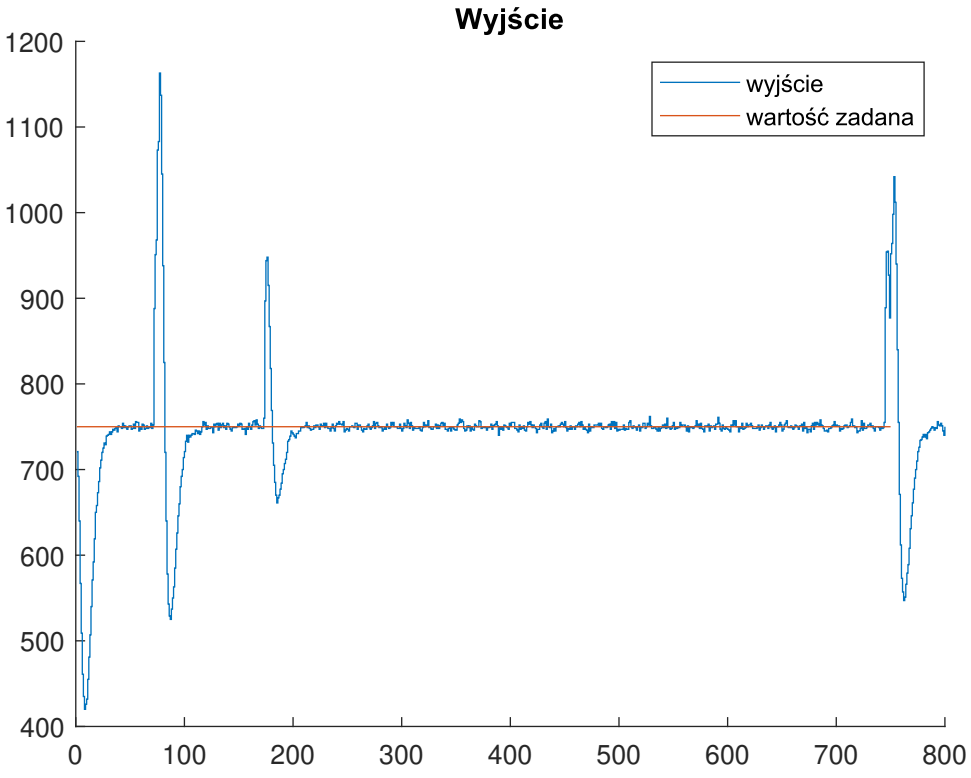
Dla zmiany parametru $N = 10$:

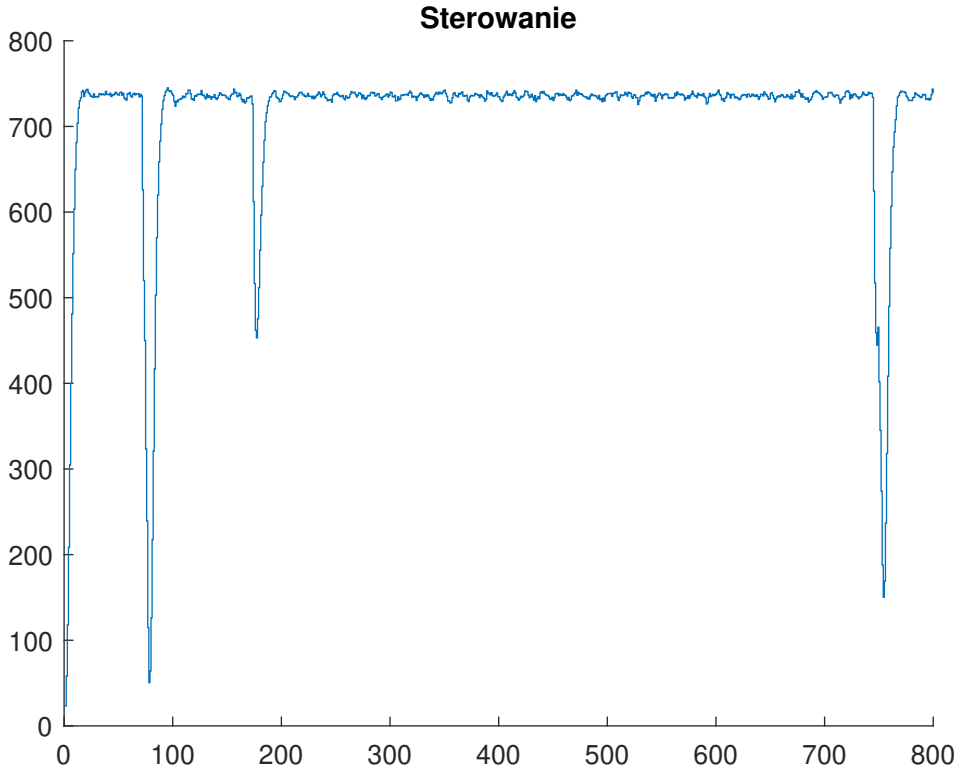




Obiekt widocznie przyśpieszył. Przy stabilizowaniu po zakłóceniach występuje przeregulowanie. Widać również wzrost małych oscylacji, w czasie laboratoriów myśleliśmy, że wynika to z zakłóceń mikrokontrolera, lecz przy porównywaniu testów na DMC widać, że prawdopodobnie mogą zwiększyć ich wpływ również nastawy regulatora.

Dla zmiany parametru $N_u = 1$:

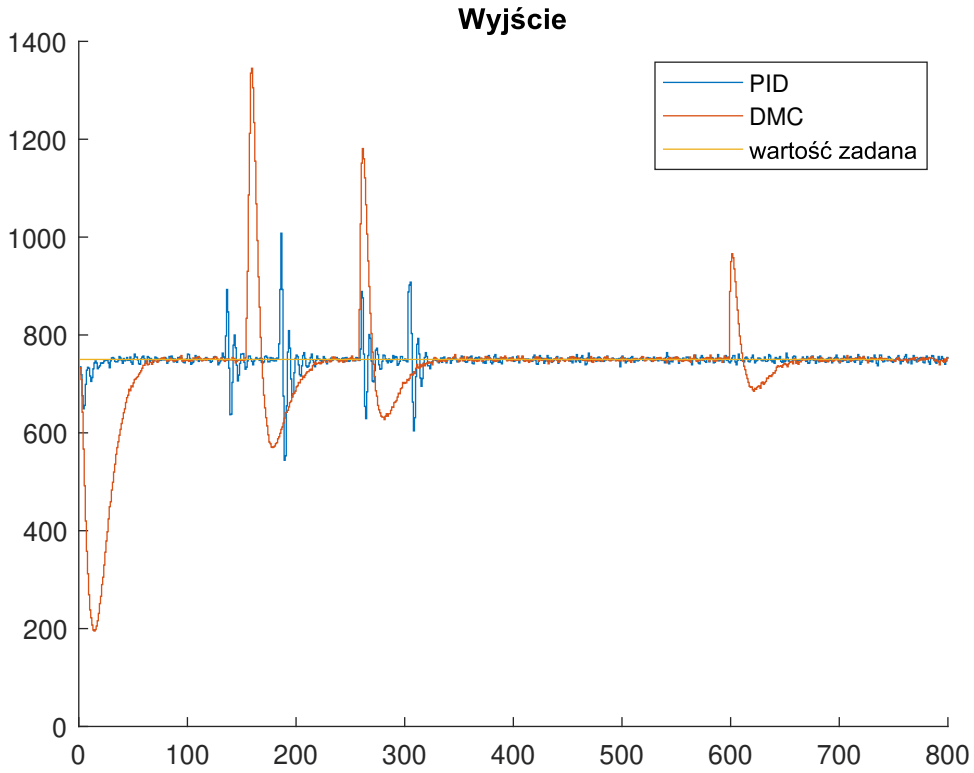


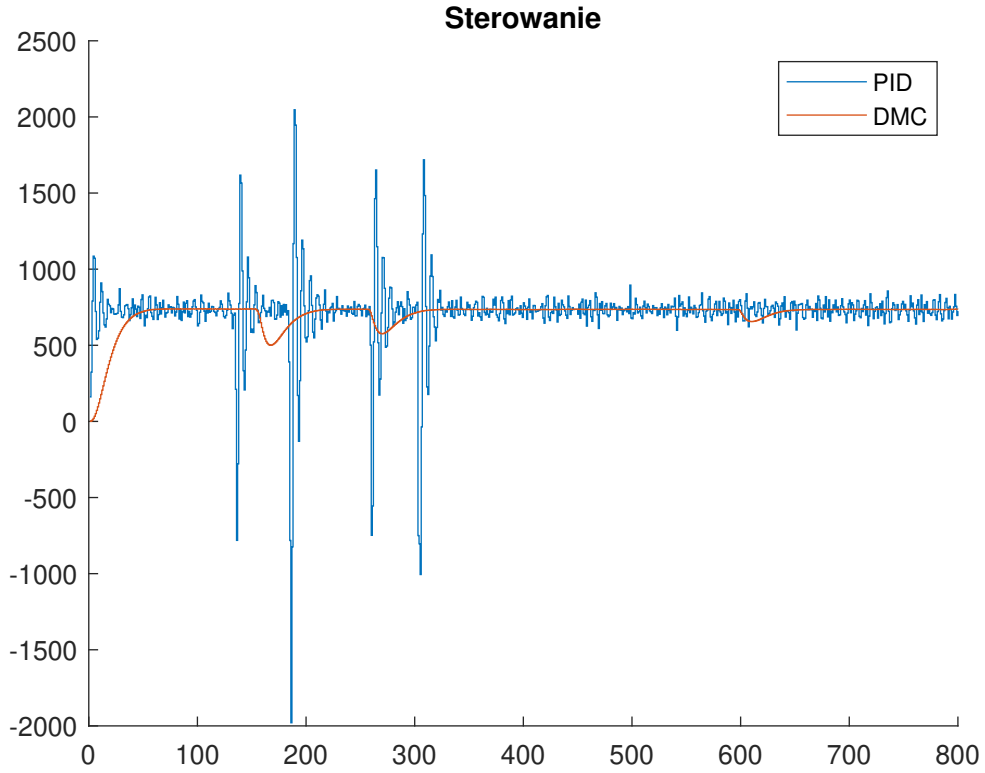


Jedyną zmianą jaka rzuca się w oczy, w porównaniu z testem dla $N = 10$ 1.2, jest "wyostrzenie" się sterowania, widoczne przy działaniu zakłóceń.

1.3. Podsumowanie

Za najlepszy algorytm DMC uznaliśmy: 1.2 o parametrach : $D = 30$, $N = 2$, $N_u = 10$ i $\lambda = 2$. Za najlepszego PID uznaliśmy: 1.4 o parametrach $K = 8,5$; $T_I = 0,8$; $T_D = 0,02$; $T_v = 0,1$.





Widać, że DMC działa lepiej. W przypadku PIDa widać duże oscylacje i wyjścia jak i sterowania (pod próg ograniczeń). DMC dochodzi do wartości zadanej łagodniej i mniej gwałtownie, jest bardziej odporny na szumy co przekłada się na lepsze osiągi w regulowaniu obiektu.