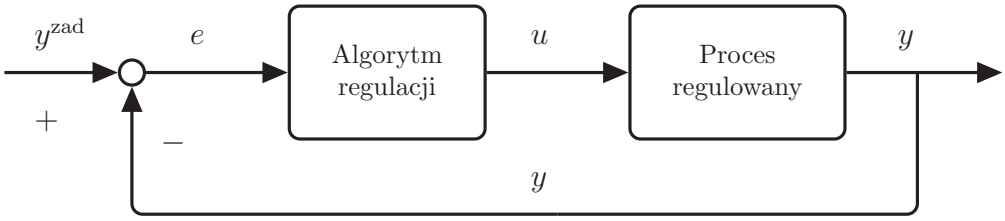


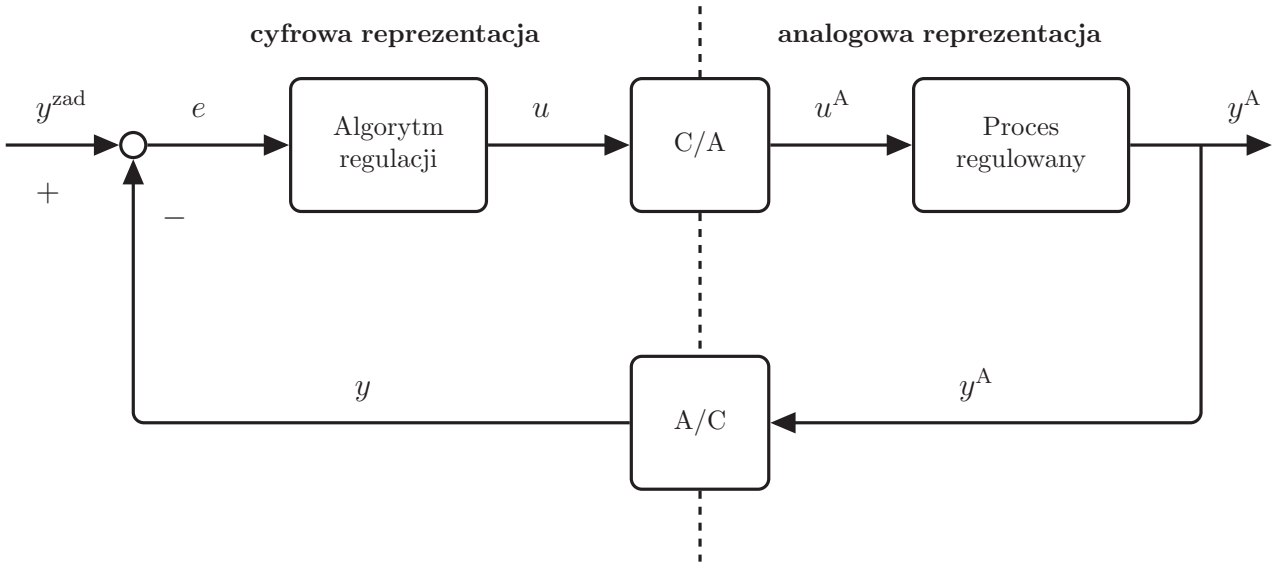
Rysunek 36: Schemat ideowy układu regulacji

5 Implementacja algorytmów regulacji PID i DMC prostego procesu dynamicznego. Interfejs użytkownika. Archiwizacja pomiarów. Dobór nastaw algorytmów. Badania porównawcze.

Cel Celem tego projektu jest implementacja dwóch algorytmów regulacji: PID oraz DMC, oraz porównanie jakości ich działania przy użyciu symulowanego obiektu. Do realizacji tego zadania konieczna jest umiejętność implementacji podanych algorytmów, umiejętność gromadzenia i analizy danych oraz rozsądne zarządzanie przerwaniami i ich priorytetami. Regulator PID jest obecny powszechnie w przemyśle, dlatego jego poprawna implementacja i strojenie są ważnymi praktycznymi umiejętnościami. Regulator DMC jest jednym z najprostszych algorytmów regulacji predykcyjnej, który wymaga wyjątkowo niewiele obliczeń oraz posiada analityczne rozwiązanie pod warunkiem nieuwzględniania ograniczeń. Ponieważ warto zdawać sobie sprawę z zalet i ograniczeń obu – celem ostatecznym tego projektu jest sprawozdanie podsumowujące ich różnice w działaniu.

Zadanie regulacji Zadaniem regulacji jest taka manipulacja sygnałem wejściowym procesu (manipulowanym) u, aby wartość sygnału wyjściowego procesu (regulowanego) y była możliwie bliska wartości zadanej y^{zad}. Często wartość uchybu $y^{\text{zad}} - y$ oznacza się symbolem e (Rys. 36). Jest to podstawowa realizacja sprzężenia zwrotnego. Algorytmy regulacji mogą być bardzo zróżnicowane – począwszy od najprostszego regulatora typu „włącz/wyłącz” (przykładowy piekarnik), aż po algorytmy wykorzystujące skomplikowane modele nieliniowe, aby na ich podstawie wyznaczać nowe sygnały sterowania. Rosnący poziom skomplikowania algorytmu przekłada się często zarówno na jakość regulacji (tj. jej dokładność, szybkość) jak i na wymagania związane z realizacją tego algorytmu. Stąd wciąż jednym z najpopularniejszych algorytmów regulacji jest prosty regulator PID, którego wymagania są minimalne a jakość regulacji w wielu przypadkach satysfakcjonująca.

Realizując to ćwiczenie warto mieć świadomość istnienia przetworników analogowo-cyfrowych i cyfrowo-analogowych, które występują pomiędzy procesem regulowanym a regulatorem. Dodatkowo koniecznie należy pamiętać, że czas wyznaczenia obliczeń nie zawsze może być pomijalny. W przypadku, gdy nowa wartość sygnału sterującego wyznaczana jest w połowie czasu między kolejnymi iteracjami algorytmu regulacji (chwilami próbkowania) warto rozważyć opóźnienie aplikacji do procesu nowej wartości sterowania do momentu rozpoczęcia nowej iteracji. W ten sposób sztucznie wprowadzone opóźnienie pozwoli na utrzymanie regularnego przekazywania nowej wartości sterowania do obiektu regulacji. Takie podejście jest szczególnie ważne w sytuacjach, gdy czas pojedynczej iteracji algorytmu regulacji nie jest stały (np. z powodu użycia



Rysunek 37: Schemat ideowy układu regulacji z uwzględnieniem przetworników analogowo-cyfrowych i cyfrowo-analogowych

funkcji do rozwiązywania zadań programowania kwadratowego).

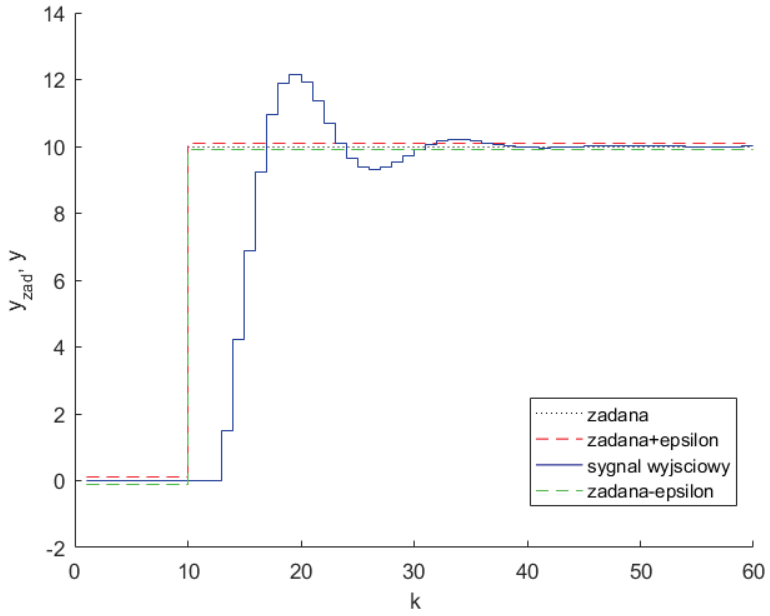
Zanim omówione zostaną algorytmy regulacji potrzebne jest wprowadzenie dwóch pojęć, które służą do oceny jakości regulacji. Są to „przesterowanie" oraz „czas ustalenia". Obie te wartości są związane z odpowiedzią obiektu na skok wartości zadanej. Aby te wartości można było ze sobą porównać, a co za tym idzie porónwać jakość regulacji różnych parametrów algorytmu regulacji, należy pamiętać o zachowaniu identycznych warunków eksperymentów. Oznacza to, że eksperymenty służące do wyznaczenia przesterowania oraz czasu ustalenia muszą rozpoczynać się zawsze w takim samym stanie początkowym, skok wartości zadanej musi być identyczny w każdym przebiegu eksperymentu a przesterowanie i czas ustalenia muszą być liczone w ten sam sposób.

Są różne definicje wartości przesterowania – jedną z najczęstszych jest

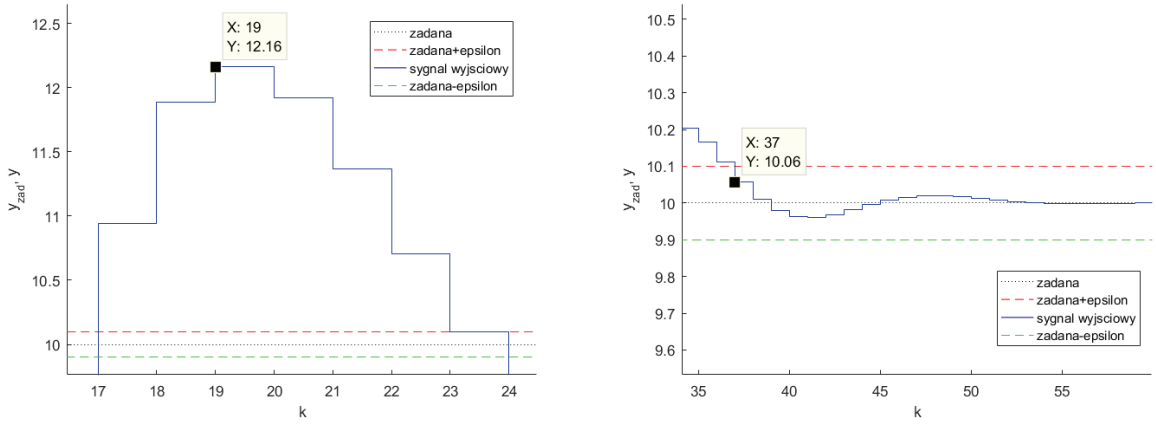
$$K_o = \frac{y_m - y^{\text{zad}}}{y^{\text{zad}}} \cdot 100\%$$

gdzie y_m jest maksymalną osiągniętą w trakcie eksperymentu wartością sygnału wyjściowego. Sygnał wyjściowy będzie w najwyższym położeniu chwilę po zmianie wartości zadanej – kolejne jego wartości powinny być od tej wyłącznie niższe. W przeciwnym razie układ może być niestabilny.

Wartość czasu ustalenia T_{ust} określa się jako czas od zmiany wartości sygnału zadanego, do chwili gdy wartość bezwzględna uchybu przestaje przekraczać pewną przyjętą niewielką wartość ε. Dla przykładu, jeśli założyć, że wartość zadana y^{zad} zmieniana jest w chwili $k = 10$ z 0 na 10, a $\varepsilon = 0{,}1$, to czasem ustalenia jest czas od zmiany wartości zadanej (10), do momentu, w którym wartość wyjściowa wchodzi w zakres wartości $y^{\text{zad}} - \varepsilon$ do $y^{\text{zad}} + \varepsilon$ i więcej go nie opuszcza. Na Rys. 38 przedstawiony został wykres wartości wyjsciowej obiektu regulacji, na podstawie którego wyznaczone mogą zostać przesterowanie oraz



Rysunek 38: Wykres będący efektem eksperymentu, mający na celu wyznaczenie wartości przesterowania oraz czasu ustalenia



Rysunek 39: Odczyt maksymalnej wartości sygnału wyjściowego obiektu regulacji będący podstawą do wyznaczenia przesterowania (lewy wykres) oraz odczyt czasu ustalenia (prawy wykres)

czas ustalenia. Rys. 39 (lewa strona) przedstawia fragment poprzedniego wykresu, gdzie widać wyraźnie wartość maksymalną sygnału wyjściowego obiektu regulacji, tj. $y_m = 12{,}16$. Na tej podstawie można wyznaczyć wartość przesterowania $K_o = \frac{12{,}16-10}{10} \cdot 100\,\% = 21{,}6\,\%$. Czas ustalenia wyznaczyć można na podstawie Rys. 39 (prawy wykres) – widać, że uchyb przestaje przekraczać wartość ε od chwili $k = 37$. Zakładając, że czas próbkowania jest równy 0,1 s, to czas ustalenia jest równy $T_{\text{ust}} = (37 - 10) \cdot 0{,}1 = 2{,}7\,\text{s}$.

Algorytm PID PID jest to algorytm regulacji składający się z trzech czynników: proporcjonalnego (*Proportional*), całkującego (*Integral*) i różniczkującego (*Derivative*). Każdy z tych członów reaguje na zmianę sygnału wyjściowego procesu regulowanego o innym charakterze. Człon propocjonalny powoduje wzrost wartości sterowania wraz ze wzrostem uchybu (tj. różnicy między wartością zadaną a wyjściową). Człon całkujący zwiększa wartość sygnału sterującego wraz z akumulowanym uchybem (tj. sumą uchybów z przeszłości) – jest to bardzo wygodny mechanizm pozwalający na niwelację uchybu ustalonego (zostanie on omówiony niżej). Ostatnim członem jest człon różniczkujący – im szybciej wzrasta uchyb, tym większa jest wartość sygnału sterującego.

Waga każdego z tych członów może być dowolnie modyfikowana, w szczególności każdy z tych członów może zostać wyłączony poprzez odpowiednią manipulację związanym z nim parametrem. Pozwala to na łatwe i jednocześnie dostosowanie właściwości regulatora do potrzeb użytkownika. W dalszej części omawiany będzie wyłącznie regulator PID w wersji dyskretnej – implementacja algorytmu PID w wersji ciągłej wymaga innego podejścia.

Implementacja algorytmu PID sprowadza się do wyznaczenia w każdej iteracji aktualnego uchybu i na tej podstawie wyznaczenia nowej wartości sygnału sterującego. Prawo regulacji (tj. wzór umożliwiający obliczenie wartości sygnału sterującego w aktualnej chwili dyskretnej k) algorytmu PID przedstawia się następująco:

$$u(k) = u_{\text{P}}(k) + u_{\text{I}}(k) + u_{\text{D}}(k) \tag{1}$$

gdzie

$$\begin{aligned} u_{\text{P}}(k) &= Ke(k) \\ u_{\text{I}}(k) &= u_{\text{I}}(k-1) + \frac{K}{T_{\text{I}}} T \frac{e(k-1) + e(k)}{2} \\ u_{\text{D}}(k) &= KT_{\text{D}} \frac{e(k) - e(k-1)}{T} \end{aligned} \tag{2}$$

Powyższe powstało poprzez zastosowanie metody Eulera oraz całkowania metodą trapezów do wzoru na ciągły w czasie regulator PID. Jak zostało wcześniej zaznaczone: $u(k)$, $y(k)$, $e(k)$ oznaczają kolejno wartości sygnału sterującego, wyjściowego procesu regulowanego i uchybu (różnicy między wyjściem procesu regulowanego a wartością zadaną) w dyskretnej chwili k. $u_{\text{P}}(k)$, $u_{\text{I}}(k)$, $u_{\text{D}}(k)$ oznaczają kolejno wartość sterowania wyznaczoną na podstawie członu proporcjonalnego, całkującego i różniczkującego. Suma sygnałów wyjściowych trzech członów składowych regulatora jest faktycznym sygnałem sterującym, który należy zaaplikować następnie do procesu regulowanego. Zmienna T oznacza tutaj czas próbkowania (tj. czas pomiędzy kolejnymi dwiema iteracjami algorytmu regulacji) wyrażony w sekundach. Każdy z wymienionych czynników jest opisany pewnym parametrem. W przypadku członu proporcjonalnego jest to parametr K – wzmocnienie. Dla członu całkującego jest to czas zdwojenia T_{I}, natomiast dla członu różniczkujacego tym

parametrem jest czas wyprzedzenia T_D. Wartym uwagi jest fakt, iż dla $K = 0$ wyłączony jest nie tylko człon proporcjonalny, ale także i pozostałe. Oczywiście algorytm PID ma wiele wersji i postaci – ta jednak posiada parametry, które reprezentują pewne rzeczywiste właściwości. Jeśli założyć, że uchyb $e(k) = 0$ dla $k < 0$ i $e(k) = \bar{e} \neq 0$ w pozostałych przypadkach, gdzie $\bar{e}$ jest pewną niezerową stałą (tj. $e(k)$ jest stałe w czasie) i człon różniczkujący jest wyłączony (tj. $T_\mathrm{D} = 0$), to T_I jest czasem (w sekundach), który musi minąć, aby $u_\mathrm{I}(k)$ osiągnęło wartość równą $u_\mathrm{P}(k)$ (które w tym przypadku jest, tak jak uchyb, stałe w czasie). Zakładając jednak, że wyłączony został człon całkujący (T_I jest nieskończenie duży), a uchyb $e(k) = 0$ dla $k < 0$ i $e(k) = \bar{e}k \neq 0$ w pozostałych przypadkach, gdzie $\bar{e}$ jest pewną niezerową stałą (tj. $e(k)$ wzrasta liniowo w czasie), to T_D jest czasem (również w sekundach), po jakim człon różniczkujący $u_\mathrm{D}(k)$ osiągnie wartość równą członowi proporcjonalnemu $u_\mathrm{P}(k)$. Czasem wartości czasów wyprzedzenia i zdwojenia zamienia się na współczynniki wzmocnienia poszczególnych członów zgodnie z poniższym:

$$K_\mathrm{I} = \frac{K}{T_\mathrm{I}}$$
$$K_\mathrm{D} = KT_\mathrm{D}$$

Tabelka z parametrami wyznaczonymi metodą Zieglera-Nicholsa operuje najczęściej takimi właśnie zmiennymi – zarówno tabelka jak i metoda podane są w dalszej części.

We wzorze (2) można zobaczyć występowanie rekurencji. Aby uniknąć konieczności zapisywania dodatkowej zmiennej w pamięci mikrokontrolera, na którym realizowany będzie ten algorytm warto posłużyć się postacią przyrostową algorytmu PID

$$u(k) = r_2 e(k-2) + r_1 e(k-1) + r_0 e(k) + u(k-1) \tag{3}$$

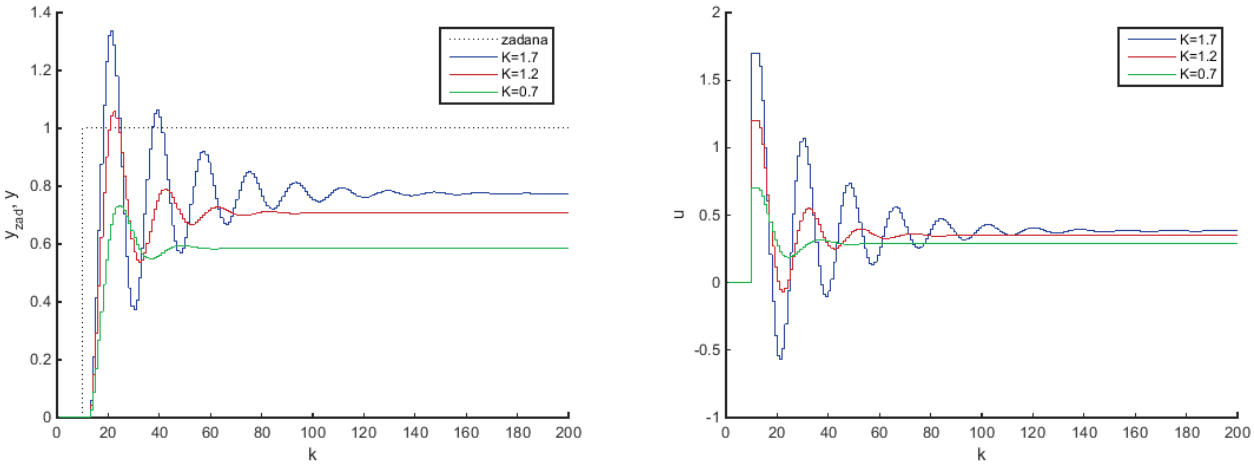
gdzie

$$r_2 = \frac{KT_\mathrm{D}}{T}$$
$$r_1 = K\left(\frac{T}{2T_\mathrm{I}} - 2\frac{T_\mathrm{D}}{T} - 1\right)$$
$$r_0 = K\left(1 + \frac{T}{2T_\mathrm{I}} + \frac{T_\mathrm{D}}{T}\right)$$

Jak widać rekurencja została zastąpiona wykorzystaniem dodatkowego uchybu z chwili $k-2$ oraz sterowania z chwili $k-1$. Wzory (1) oraz (3) są sobie równoważne, tj. można z jednego przejść do drugiego wyłącznie przy użyciu prostych przekształceń (wystarczy od każdej strony równania (1) odjąć $u(k-1)$). Wszelkie oznaczenia są identyczne jak przy omawianiu poprzednich wzorów. Wartości r_2, r_1, r_0 nie posiadają jednak ciekawszego znaczenia niż „zmienne, które stoją przy kolejnych uchybach". Do realizacji ćwiczenia warto posłużyć się wzorem (1), gdyż jest bardziej intuicyjny.

Skoro już wyznaczone zostały wzory, warto zastanowić się jak działa regulator PID w praktyce. Oczywiście wynika to wprost ze wzorów. Poniżej omawiane wykresy są efektem regulacji obiektu o następującym równaniu różnicowym:

$$y(k) = 0{,}043\,209u(k-1) + 0{,}030\,415u(k-2) + 1{,}309\,644y(k-1) - 0{,}346\,456y(k-2)$$



Rysunek 40: Wyjście oraz wejście procesu regulowanego – regulator P

gdzie k oznacza obecną chwilę dyskretną. Natomiast parametry algorytmu PID zostały dobrane arbitralnie jako następujące:

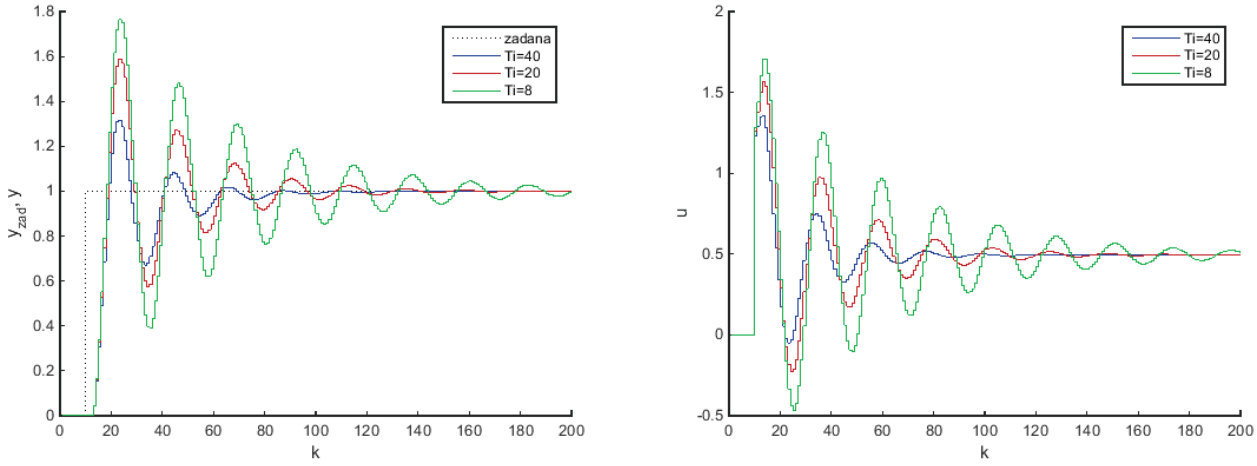
$$T = 2 \qquad K = 1{,}2 \qquad T_{\mathrm{I}} = 20 \qquad T_{\mathrm{D}} = 5;$$

uwzględniając ewentualne wyłączanie poszczególnych członów.

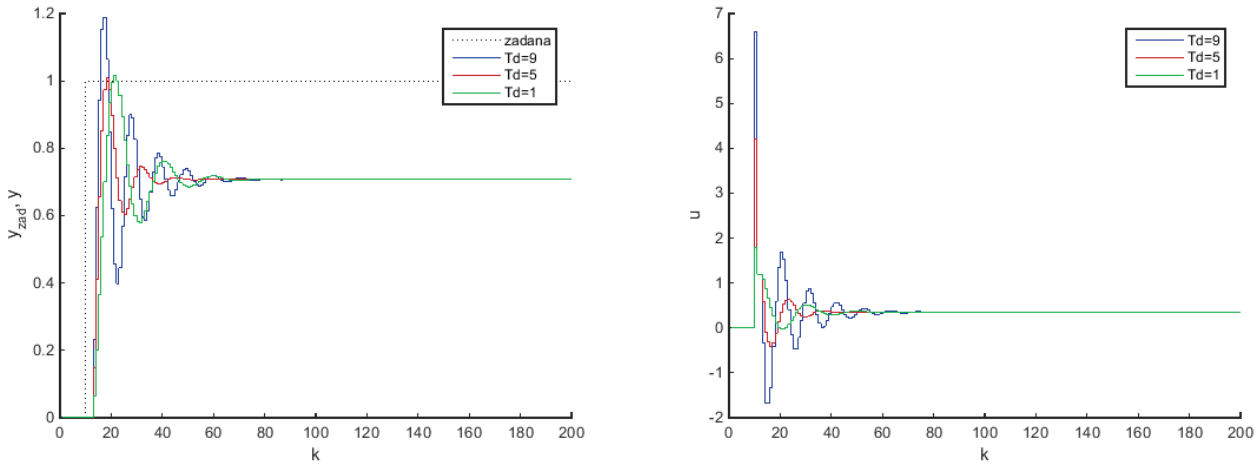
Regulator P (wyłączone człony całkujący i różniczkujący) generuje sygnał sterujący proporcjonalny do różnicy między wartością zadaną a wyjściem procesu regulowanego. Na Rys. 40 widać, że, nawet po bardzo długim czasie, wyjście procesu regulowanego nie jest równe wartości zadanej. Wręcz przeciwnie – różnica między nimi jest stała. Zjawisko takie nazywane jest uchybem ustalonym. Aby zrozumieć to zjawisko warto zastanowić się co działoby się w sytuacji gdyby jednak uchyb był zerowy. Wtedy sterowanie również byłoby zerowe, a co za tym idzie wartość wyjściowa procesu by spadła. To spowodowałoby narastanie uchybu, który powodowałoby narastanie sterowania i hamowanie opadania sygnału wyjściowego. W pewnym momencie nastąpi osiągnięcie równowagi i będzie można ponownie zaobserwować uchyb ustalony. Inną ciekawą cechą jest malejąca wartość uchybu ustalonego wraz ze wzrostem wzmocnienia (nie ma tak dużego wzmocnienia, które wyeliminuje uchyb ustalony!) – niestety jednocześnie wzrasta czas i amplituda oscylacji.

Aby pozbyć się tego zjawiska warto wprowadzić człon całkujący, który spowoduje, że kolejne błędy będą się akumulować i wraz ze wzrostem tego zsumowanego błędu regulator będzie zwiększał wartość sterowania przybliżając się do wartości zadanej. Działanie regulatora PI można zaobserwować na Rys. 41. Mimo zwiększonego czasu oscylowania sygnału wyjściowego – uchyb ustalony został wyeliminowany.

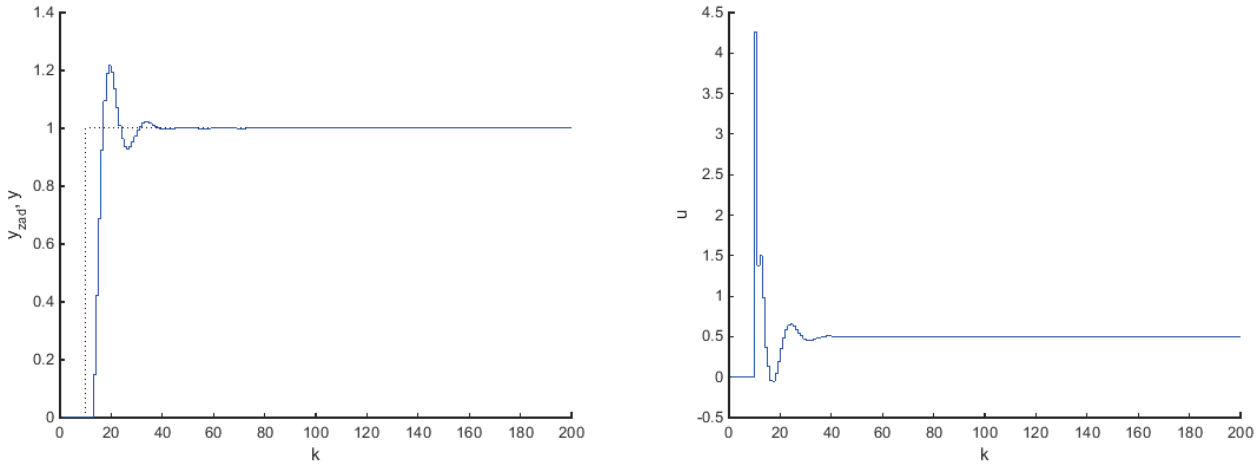
Człon różniczkujący ma za zadanie reagować na zmianę uchybu. Wraz ze wzrostem prędkości jego narastania, wzrasta wartość sterowania. Szczególnie widoczne jest to w przypadku zmiany wartości zadanej, która na Rys. 42 ma miejsce w chwili 20. W tym miejscu na wykresie sygnału sterującego zauważyć można bardzo wysoki skok wartości sterowania – jest on spowodowany właśnie przez człon różniczkujący, ponieważ różnica między uchybem obecnym, a uchybem z poprzedniej chwili jest bardzo wysoka. Ponieważ rozważanym regulatorem jest regulator PD, ponownie można zaobserwować uchyb ustalony.



Rysunek 41: Wyjście oraz wejście procesu regulowanego – regulator PI



Rysunek 42: Wyjście oraz wejście procesu regulowanego – regulator PD



Rysunek 43: Wyjście oraz wejście procesu regulowanego – regulator PID będący punktem odniesienia

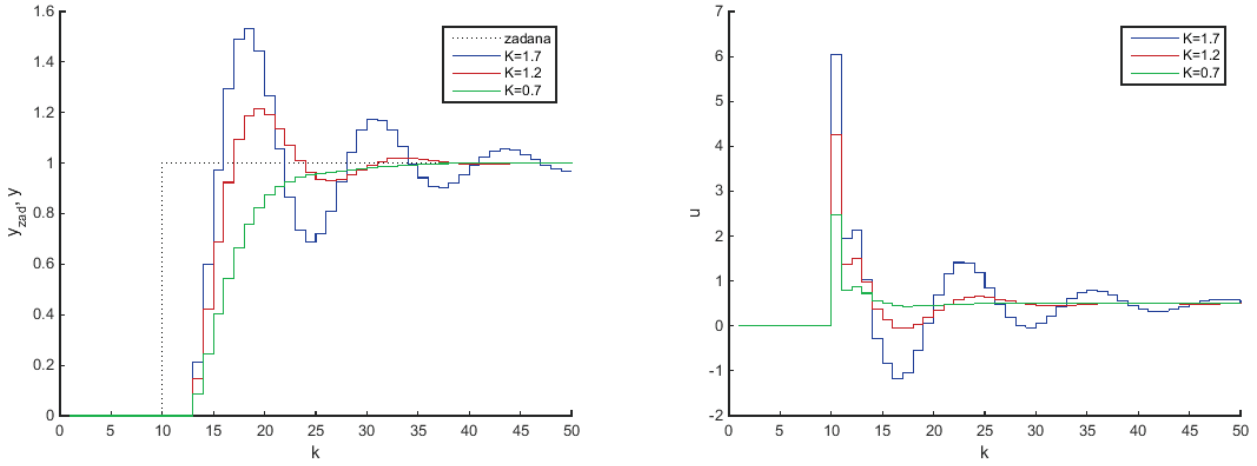
Regulator PID jest złączeniem wszystkich powyższych członów, a więc posiada wszystkie ich cechy. Najważniejszymi są brak uchybu ustalonego oraz chwilowy skok wartości sterowania w momencie zmiany wartości zadanej. Wykres pełnego regulatora PID widoczny jest na Rys. 43.

Na Rys. 44, 45 oraz 46 pokazane są przebiegi sygnałów wejściowego i wyjściowego w zależności od zmiany jednego z parametrów (nastaw) regulatora PID. Dla Rys. 45 pokazany został dłuższy przebieg aby można było zaobserwować powolne dążenie do wartości zadanej takiego algorytmu. Dla pozostałych przebiegów pokazane zostało tylko 100 pierwszych sekund, ponieważ po osiągnięciu wartości zadanej przebiegi szybko się stabilizowały i były do siebie zbliżone na przestrzeni pozostałego czasu.

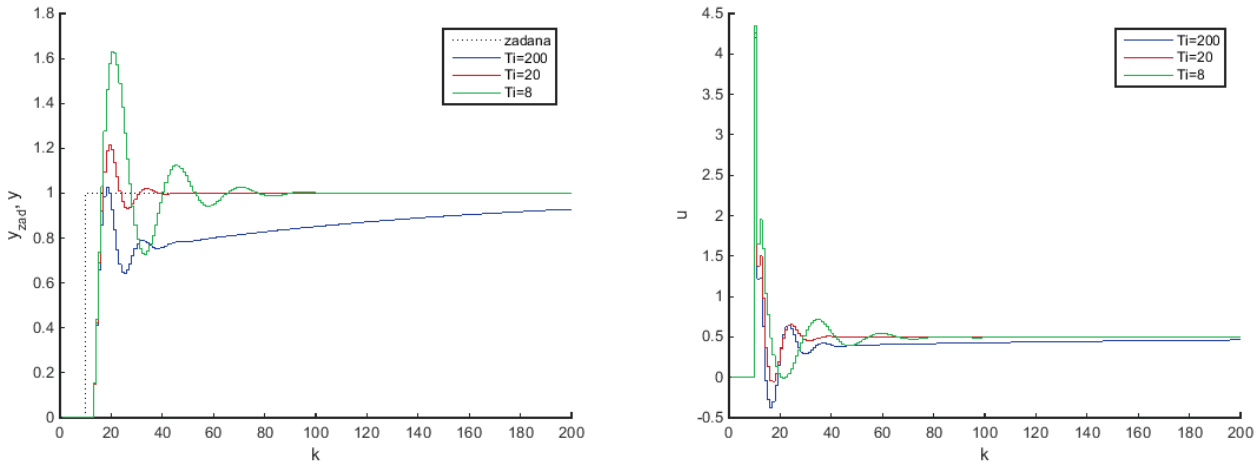
Ponieważ w prawie sterowania algorytmu PID nie ma miejsca na uwzględnienie ograniczeń na wartości sterowania, należy poradzić sobie z tym problemem osobno. Brak uwzględnienia ograniczeń powoduje, że gdy zostanie ono osiągnięte, a błąd będzie niezerowy, realizowane będzie niepotrzebne całkowanie, tj. wyznaczona wartość sygnału sterującego będzie rosła mimo, że faktyczna maksymalna wartość sygnału sterującego już została osiągnięta. Efekty tego zjawiska widoczne są szczególnie w momencie, gdy sygnał wyjściowy przekroczy wartość zadaną – wtedy przez długi czas dokonywane jest „odcałkowywanie", tj. sygnał sterujący jest nieustannie pomniejszany (przez składową całkującą), aż jego wartości będą mniejsze niż ograniczenie i regulator będzie ponownie pracować zgodnie z oczekiwaniami (pod warunkiem, że w tym momencie jeszcze obiekt nie wpadł w oscylacje). Zjawisko przesadnego całkowania jest nazywane nawijaniem (*windup*) – poniżej omówiony zostanie prosty algorytm pozwalający na niwelację jego wpływu (algorytm *anti-windup*).

Jednym z prostszych algorytmów przeciwdziałających nawijaniu jest pomniejszanie składowej całkującej wartości sterowania o pewną stała przemnożoną przez różnicę między nasyconą wartością sygnału sterującego, a wartością wyznaczoną przez regulator.

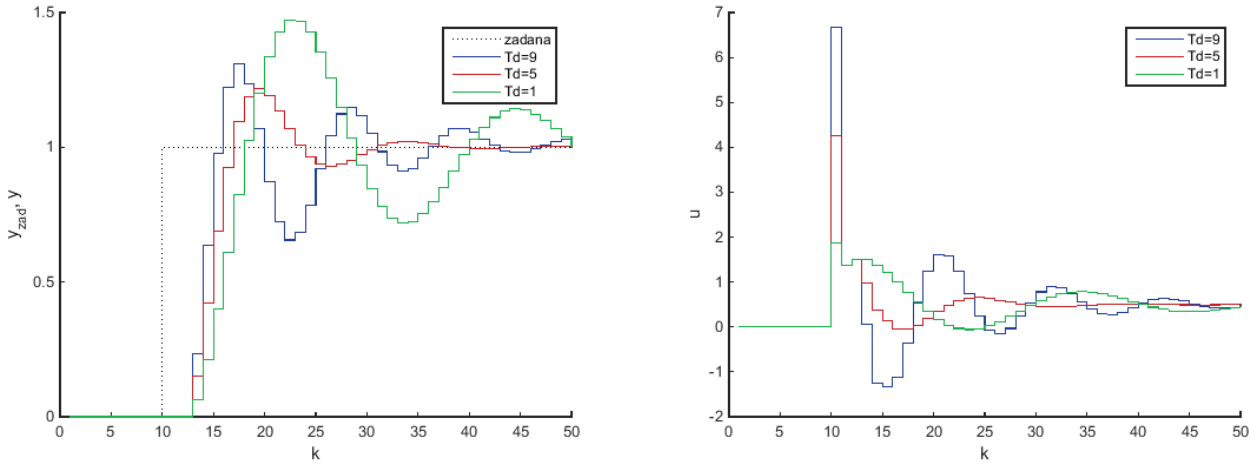
Realizowane jest to poprzez wprowadzenie do członu całkującego dodatkowego elementu proporcjonal-



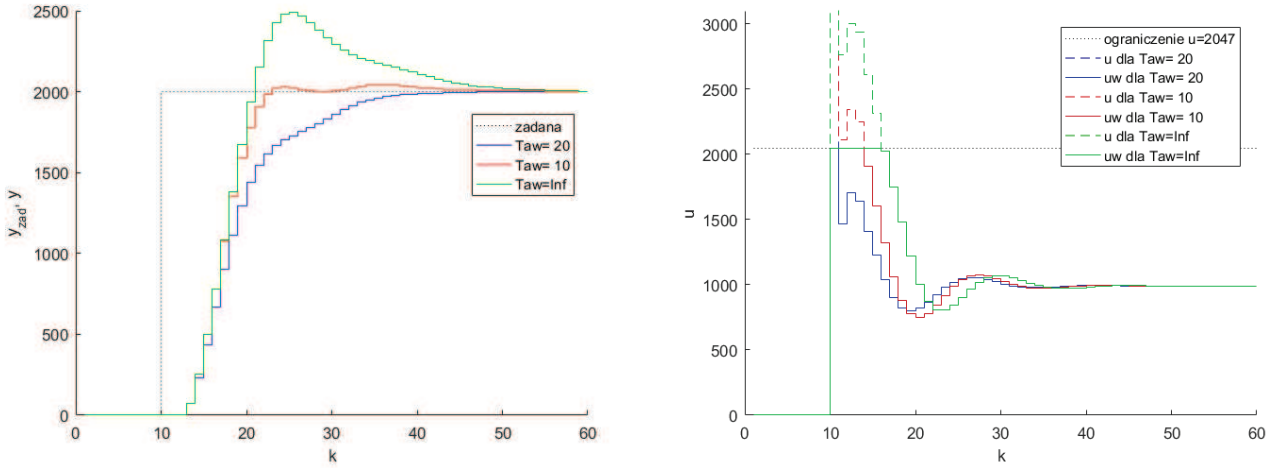
Rysunek 44: Wyjście oraz wejście procesu regulowanego w zależności od wzmocnienia – regulator PID



Rysunek 45: Wyjście oraz wejście procesu regulowanego w zależności od czasu zdwojenia – regulator PID



Rysunek 46: Wyjście oraz wejście procesu regulowanego w zależności od czasu wyprzedzenia – regulator PID



Rysunek 47: Wyjście oraz wejście procesu regulowanego – algorytm PID z oraz bez algorytmu *anti-windup* (pokazane wartości sygnału sterującego ograniczone są do 3100 dla zwiększenia czytelności)

nego (2) do różnicy między faktycznym sygnałem sterującym a oczekiwanym

$$u_{\mathrm{I}}(k) = u_{\mathrm{I}}(k-1) + \frac{K}{T_{\mathrm{I}}} T \frac{e(k-1)+e(k)}{2} + \frac{T}{T_{\mathrm{v}}}(u_w(k-1) - u(k-1))$$

gdzie T_{v} jest parametrem algorytmu *anti-windup*. Sterowanie $u_w(k-1)$ jest to sterowanie, które zostało faktycznie zaaplikowane do procesu, a więc sygnał, który jest ograniczony najczęściej fizycznymi właściwościami obiektu regulowanego. Natomiast $u(k-1)$ oznacza wartość sygnału sterowania, która została wyznaczona poprzez zastosowanie prawa regulacji algorytmu PID. Na Rys. 47 przedstawione zostało porównanie przebiegu sygnału wyjsciowego i sterowania przy wykorzystaniu ($T_{\mathrm{v}} = 10$ i $T_{\mathrm{v}} = 20$)i bez zastosowania algorytmu *anti-windup* ($T_{\mathrm{v}} = \infty$). Na wykresach widoczne jest ucięcie sygnału sterującego wynikające z ograniczeń procesu ($u = 2000$) – dla wyłączonego algorytmu *anti-windup* występuje długie przesterowanie wynikające z przekroczenia przez sygnał sterujący wartości ograniczenia. Następuje wtedy ($k = 11$) akmulacja członu całkującego, a następnie dopiero w chwili $k = 13$ następuje odcałkowywanie, które sprowadza sygnał sterujący do dozwolonych wartości po kolejnych dwóch iteracjach. Korzystając z mechanizmu *anti-windup* można skrócić ten czas (w zależności od parametru T_{v}), co oznacza jednocześnie szybszą reakcję algorytmu regulacji w przypadku osiągnięcia ograniczeń i najczęściej również ograniczenie przesterowania.

Reguły Zieglera-Nicholsa Jedną z metod wyznaczenia parametrów algorytmu PID jest zastosowanie reguł Zieglera-Nicholsa. Reguły te wyznaczone zostały na podstawie wielu eksperymentów praktycznych, co pozwala zaliczyć tę metodę do metod o charakterze heurystycznym. Ta metoda wyznaczania parametrów regulatora PID nie gwarantuje optymalności rozwiązania, a nawet nie daje pewności, że wyznaczone nastawy nie zdestabilizują układu.

Zastosowanie reguł Zieglera-Nicholsa wymaga przeprowadzenia eksperymentu. Należy zaimplementować regulator typu P dla rozważanego obiektu regulacji. Następnie należy tak dobrać wartość wzmocnienia regulatora K, aby sygnał wyjściowy obiektu regulacji miał charakter oscylacyjny. Trzeba tak dobrać wzmocnienie, aby amplituda oscylacji nie malała i nie rosła – taki układ będzie więc na skraju stabilności. K, które pozwala uzyskać niegasnące i nierosnące oscylacje nazwane zostanie wzmocnieniem krytycznym K_u. Drugim parametrem jaki jest potrzebny do dalszej pracy jest okres drgań w stanie skrajnej stabilności – parametr ten będzie oznaczany jako T_u. Posiadając takie dwie wartości można wyznaczyć parametry dla algorytmów P, PI oraz PID – wystarczy skorzystać z poniższej tabelki

Regulator	K	T_{I}	T_{D}
P	$0{,}5K_u$	–	–
PI	$0{,}45K_u$	$T_u/1{,}2$	–
PID	$0{,}6K_u$	$T_u/2{,}0$	$T_u/8$

Tak wyznaczone parametry powinny dawać rozsądną jakość regulacji. Niestety reguły te nie gwarantują ani najlepszych wyników, ani nawet poprawnych. Jest to jednak jedna z najstarszych metod wyznaczania nastaw regulatorów PID i jest to metoda przede wszystkim wygodna – nie ma potrzeby tworzenia żadnego modelu procesu. Dodatkowo nawet jeśli te parametry będą niesatysfakcjonujące – jest to dobry zestaw parametrów, aby od niego rozpocząć poszukiwanie takich nastaw, które będą spełniały wszystkie założone wymagania.

Z drugiej strony wprowadzanie obiektu regulacji w stan skrajnej stabilności nie brzmi jak najlepszy pomysł, szczególnie jeśli rozważanym obiektem regulacji jest np. reaktor jądrowy. Oczywiście taki eksperyment może być nie tylko niebezpieczny, ale również szkodliwy dla urządzeń wykonawczych – może nastąpić ich szybsze zużycie lub przesunięcie ich punktu pracy. W ramach projektu wykorzystany został obiekt symulacyjny, w związku z czym użycie tej metody jest jak najbardziej bezpieczne.

Metoda „inżynierska" Inną metodą jest metoda przypominająca zachowaniem metodę minimalizacji gradientowej. Metoda ta będzie podzielona na trzy etapy – dobór parametru K, dobór parametru T_I oraz dobór parametru T_D. Rozpocząć należy od wprowadzenia obiektu w stan skrajnej stabilności przy użyciu regulatora typu P – tak jak w metodzie Zieglera-Nicholsa. Tak wyznaczony parametr K zostanie nazwany wzmocnieniem krytycznym K_u. 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 – oczywiście należy wcześniej przyjąć pewne kryterium oceny, jak np. minimalny czas ustalenia sygnału wyjściowego procesu. 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.

Algorytm DMC Algorytm DMC (*Dynamic Matrix Control*) jest jednym z najpopularniejszych algorytmów regulacji predykcyjnej. Omówiony zostanie algorytm wyłącznie dla procesu regulacji o jednym wejściu i jednym wyjściu, a do opisu równań będzie głównie wykorzystany zapis macierzowo-wektorowy.

W algorytmie DMC, w każdej kolejnej dyskretnej chwili (iteracji algorytmu) wyznaczany jest wektor przyszłych przyrostów wartości sterowania

$$\triangle U(k) = \begin{bmatrix} \triangle u(k|k) \\ \vdots \\ \triangle u(k+N_\mathrm{u}-1|k) \end{bmatrix} \tag{4}$$

Wektor ten jest długości N_u. Zakłada się, że $\triangle u(k+p|k) = 0$ dla $p \geqslant N_\mathrm{u}$, gdzie N_u jest horyzontem sterowania. Celem działania algorytmu jest minimalizacja różnic między trajektorią zadaną $y^{\mathrm{zad}}(k+p|k)$ (tj. wektorem kolejnych zadanych wartości sygnału wyjściowego) a predykowaną trajektorią sygnału wyjściowego $\hat{y}(k+p|k)$ (tj. wektorem jego kolejnych prognozowanych wartości) na horyzoncie predykcji N (tj. długość predykowanej trajektorii jest równa N). Rozwiązywane jest zatem następujące zadanie minimalizacji

$$\min_{\triangle U(k)} \left\{ \sum_{p=1}^{N} (y^{\mathrm{zad}}(k+p|k) - \hat{y}(k+p|k))^2 + \sum_{p=0}^{N_u-1} \lambda_p (\Delta u(k+p|k))^2 \right\}$$

co korzystając z zapisu wektorowo-macierzowego można zapisać jako

$$\min_{\triangle U(k)} \left\{ \|Y^{\mathrm{zad}}(k) - \hat{Y}(k)\|^2_{\boldsymbol{I}} + \|\Delta U(k)\|^2_{\Lambda} \right\} \tag{5}$$

gdzie $\Lambda > 0$, a $\|x\|_Y^2 = x^{\mathrm{T}} Y x$. Macierz $\boldsymbol{I}$ jest macierzą identycznościową (tj. diagonalną z samymi jedynkami na diagonali i zerami w pozostałych miejscach) Macierz Λ jest macierzą diagonalną

$$\Lambda = \begin{bmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{N_u-1} \end{bmatrix}$$

i służy do parametryzowania wpływu poszczególnych czynników na wartość minimalizowanej funkcji. W praktyce często ustala się elementy diagonali macierzy Λ, jako równe sobie i oznacza jako λ (tj. $\lambda_0 = \lambda_1 = \ldots = \lambda_{N_u-1} = \lambda$), a więc $\Lambda = \lambda \boldsymbol{I}$. Wektory $Y^{\mathrm{zad}}(k)$ oraz $Y(k)$ mają postać

$$Y^{\mathrm{zad}}(k) = \begin{bmatrix} y^{\mathrm{zad}}(k) \\ \vdots \\ y^{\mathrm{zad}}(k) \end{bmatrix}, Y(k) = \begin{bmatrix} y(k) \\ \vdots \\ y(k) \end{bmatrix}$$

Wektory $Y^0(k)$, $Y^{\mathrm{zad}}(k)$ oraz $Y(k)$ są długości N.

Mimo, że wyznaczony wektor minimalizujący wartość wyżej przedstawionej funkcji zawiera przyrosty sterowania na iteracje od 0 do $N_{\mathrm{u}} - 1$ w przód, to aplikowany do procesu jest jedynie przyrost wartości sygnału wyznaczony na chwilę obecną, tj. $\Delta u(k|k)$. Ponieważ wyznaczone zostały przyrosty, należy wyznaczoną wartość obecnego sterowania dodać do ostatniej wartości sygnału sterowania, tj. $u(k) = \triangle u(k|k) + u(k-1)$. W następnej iteracji algorytmu regulacji, uaktualniane są pomiary, a następnie ponownie wyznaczany jest cały optymalny wektor $\triangle U(k)$ i aplikowany jest jedynie pierwszy element.

Prognozy wartości sygnału wyjściowego $\hat{y}(k+p|k)$ na horyzoncie predykcji N obliczane są na podstawie modelu w postaci odpowiedzi skokowej procesu. Odpowiedź skokowa, jest to seria kolejnych wartości sygnału wyjściowego procesu będących reakcją na nagłą, jednostkową zmianę wartości sterowania. A więc jeśli zmiana sygnału sterującego nastąpiła w chwili k, to odpowiedź skokowa będzie się składała z pomiarów $\{s_1, s_2, s_3, \ldots\} = \{\triangle y(k+1), \triangle y(k+2), \triangle y(k+3), \ldots\}$, gdzie $\triangle y(k+p)$ dla $p = 1,2,\ldots$ oznacza przyrost wartości sygnału wejściowego w stosunku do stanu sprzed zmiany wartości sterowania. Ponieważ algorytm DMC jest przeznaczony dla procesów asymptotycznie stabilnych, można zapisać tylko pewną liczbę początkowych wartości odpowiedzi skokowej przyjmując założenie, że dalsze elementy miałyby wartość tę samą lub zbliżoną co ostatni element odpowiedzi skokowej. Wprowadzić więc warto wielkość D – horyzont dynamiki, taką, że $s_l = s_D$ dla $l \geqslant D$.

Jak widać eksperyment służący do pozyskania odpowiedzi skokowej jest wyjątkowo prosty, a jego liniowość pozwala na wyznaczenie analitycznego prawa regulacji (pod warunkiem nie uwzględniania ograniczeń). Uwzględnienie ograniczeń w takim wypadku może być realizowane dopiero po wyznaczeniu nowych wartości sterowania poprzez ich rzutowanie na zbiór dozwolonych wartości. Szczegóły zostaną omówione niżej.

Wektor będący rozwiązaniem zadania (5) można wyznaczyć wprost jako

$$\triangle U(k) = \boldsymbol{K} \left[Y^{\mathrm{zad}}(k) - Y^0(k) \right] \tag{6}$$

Macierz $\boldsymbol{K}$ (o wymiarach $N_u \times N$) zdefiniowana jest jako

$$\boldsymbol{K} = \left[\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} + \lambda\boldsymbol{I}\right]^{-1}\boldsymbol{M}^{\mathrm{T}} = \begin{bmatrix} \overline{\boldsymbol{K}}_1 \\ \overline{\boldsymbol{K}}_2 \\ \vdots \\ \overline{\boldsymbol{K}}_{N_{\mathrm{u}}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{K}_{1,1} & \boldsymbol{K}_{1,2} & \dots & \boldsymbol{K}_{1,N} \\ \boldsymbol{K}_{2,1} & \boldsymbol{K}_{2,2} & \dots & \boldsymbol{K}_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{K}_{N_{\mathrm{u}},1} & \boldsymbol{K}_{N_{\mathrm{u}},2} & \dots & \boldsymbol{K}_{N_{\mathrm{u}},N} \end{bmatrix}$$

gdzie trajektoria swobodna $Y^0(k)$ obliczana jest jako $Y^0(k) = Y(k) + \boldsymbol{M}^{\mathrm{P}}\triangle U^{\mathrm{P}}(k)$. Do wyznaczenia macierzy $\boldsymbol{K}$ wymagana jest znajomość macierzy $\boldsymbol{M}$ (zwanej macierzą współczynników odpowiedzi skokowej)

$$\boldsymbol{M} = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ s_2 & s_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N & s_{N-1} & \cdots & s_{N-N_{\mathrm{u}}+1} \end{bmatrix}$$

Macierz $\boldsymbol{M}$ ma wymiary $N \times N_{\mathrm{u}}$.

Macierze służące do wyznaczenia trajektorii swobodnej $Y^0(k)$ określone są jako

$$\boldsymbol{M}^{\mathrm{P}} = \begin{bmatrix} s_2 - s_1 & \cdots & s_D - s_{D-1} \\ s_3 - s_1 & \cdots & s_{D+1} - s_{D-1} \\ \vdots & \ddots & \vdots \\ s_{N+1} - s_1 & \cdots & s_{N+D-1} - s_{D-1} \end{bmatrix},\ \triangle U^{\mathrm{P}}(k) = \begin{bmatrix} \triangle u(k-1) \\ \vdots \\ \triangle u(k-(D-1)) \end{bmatrix}$$

Macierz $\boldsymbol{M}^{\mathrm{P}}$ ma wymiary $N \times (D-1)$, natomiast wektor $\triangle U^{\mathrm{P}}(k)$ jest długości D-1.

Na podstawie powyższych wzorów można wyznaczyć prawo regulacji, tj. wyznaczony zostanie wyłącznie najbliższy przyrost sterowania (tj. $\triangle u(k|k)$), który od razu posłuży do wyznaczenia wartości przyszłego sterowania $u(k|k)$

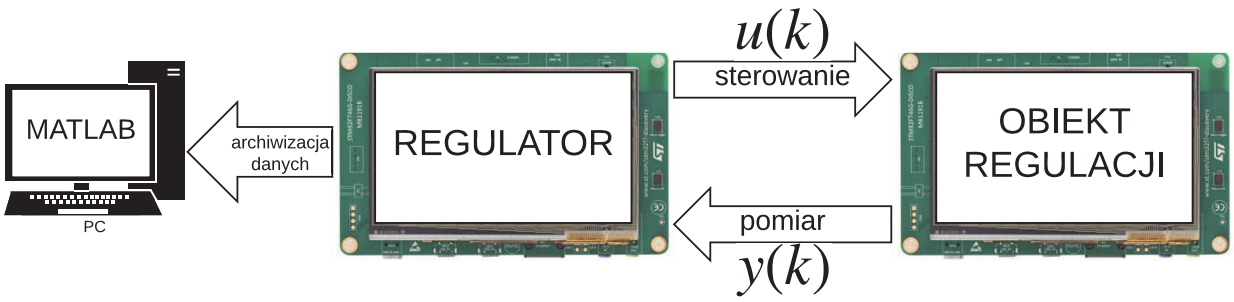
$$\begin{aligned} u(k|k) = u(k-1) + \triangle u(k|k) &= u(k-1) + \overline{\boldsymbol{K}}_1\left[Y^{\mathrm{zad}}(k) - Y^0(k)\right] \\ &= u(k-1) + \overline{\boldsymbol{K}}_1\left[Y^{\mathrm{zad}}(k) - Y(k) - \boldsymbol{M}^{\mathrm{P}}\triangle U^{\mathrm{P}}(k)\right] \\ &= u(k-1) + \overline{\boldsymbol{K}}_1\left[Y^{\mathrm{zad}}(k) - Y(k)\right] - \overline{\boldsymbol{K}}_1\boldsymbol{M}^{\mathrm{P}}\triangle U^{\mathrm{P}}(k) \\ &= u(k-1) + \sum_{i=1}^{N}\boldsymbol{K}_{1,i}(y^{\mathrm{zad}}(k) - y(k)) - \overline{\boldsymbol{K}}_1\boldsymbol{M}^{\mathrm{P}}\triangle U^{\mathrm{P}}(k) \end{aligned}$$

gdzie po zdefiniowaniu symboli $e(k) = y^{\mathrm{zad}}(k) - y(k)$, $K_e = \sum_{i=1}^{N}\boldsymbol{K}_{1,i}$ i wektora o długości $D-1$ $\boldsymbol{K}_u = \overline{\boldsymbol{K}}_1\boldsymbol{M}^{\mathrm{P}}$ można zapisać

$$u(k|k) = u(k-1) + K_e e(k) - \boldsymbol{K}_u\triangle U^{\mathrm{P}}(k)$$

W tym momencie można dokonać rzutowania anlitycznie wyznaczonego sterowania na zbiór dopuszczalnych rozwiązań. Realizowane jest to poprzez implementację poniższych warunków

$$\begin{aligned}
&\text{jeżeli } u(k|k) < u^{\min} \text{ wtedy } u(k|k) = u^{\min} \\
&\text{jeżeli } u(k|k) > u^{\max} \text{ wtedy } u(k|k) = u^{\max} \\
&u(k) = u(k|k)
\end{aligned}$$



Rysunek 48: Schemat połączenia obiektu symulowanego i regulatora

Przykład działania algorytmu DMC *Najpierw muszę opisać DMC sam w sobie...*

Symulowany obiekt Obiekt regulacji jest symulowany przy użyciu płytki rozwojowej STM32F746G-DISCOVERY. Jest ona tak samo wyposażona jak płytka działająca jako regulator, którą programować będzie student. Schemat połączenia jest przedstawiony na Rys. 48. Obiekt ten jest liniowy, asymptotycznie stabilny i posiada dwie intercje. Dokładne stałe czasowe oraz wzmocnienie, które opisują ten obiekt nie będą podawane, gdyż inaczej zadanie identyfikacji (tj. pozyskiwanie modelu w postaci odpowiedzi skokowej) mijałoby się z celem (znając równania obiektu możnaby taką odpowiedź wygenerować).

Sterowanie obiektem odbywa się poprzez zmianę wartości napięcia (w zakresie od 0 do 3,3 V) na jednym z jego pinów. Wyjściem obiektu jest również wartość napięcia (także w zakresie od 0 do 3,3 V), które można zmierzyć na jednym z jego pinów. Na wyświetlaczu obiektu rysowane są na bieżąco wartości sygnału wyjściowego obiektu (`Y1`) oraz sygnału wejściowego obiektu, tj. sterowania (`U1`). Wartości te są skalowane do przedziału od -1 do 1, gdzie -1 odpowiada napięciu równemu 0 V (na wejściu lub wyjściu obiektu), natomiast 1 odpowiada napięciu 3,3 V (zarówno na wjeściu jak i wyjsciu obiektu). Połączenie obiektu z regulatorem (tj. dwóch płytek STM32F746G-DISCOVERY) realizowane jest przez prowadzącego.

Płytka z przetwornikami Do sterowania obiektem regulacji wykorzystana została płytka zawierająca przetworniki cyfrowo-analogowe (DAC – *Digital-Analog Converter*). Mikrokontroler STM32F746G-DISCOVERY wyposażony jest we własne DAC, lecz zostały już one wykorzystane do generacji dźwięku stereo (są bezpośrednio podłączone do odpowiednich elementów na płytce rozwojowej), co powoduje niemożność ich wykorzystania do innych zadań. Wspomniane przetworniki są dołączane do do płytki rozwojowej poprzez płytkę wyposażoną w złącza zgodne ze złączem Arguino Uno.

Przetworniki te, o nazwie MCP4725A0T-E/CH, są 12-bitowe (tak jak te co są na płycie rozwojowej) a komunikacja z nimi odbywa się przy użyciu protokołu I^2C. Funkcja służąca do wysłania nowej wartości cyfrowej do przetwornika wygląda następująco

```
void updateControlSignalValue(uint16_t out){
static uint8_t buffertx[] = {0x0F, 0xFF};
buffertx[0] = (out>>8)&0x0F;
buffertx[1] = (out>>0)&0xFF;
```

```
HAL_I2C_Master_Transmit(&hi2c1, 0xC2, (uint8_t*)buffertx, 2,1000);
}
```

Kolejno wartość `out` zapisywana jest na dwóch osobnych bajtach, po czym przesyłana jest do przetwornika za pomocą funkcji z biblioteki HAL. Nie zostało wykorzystane DMA ani przerwania, lecz jednokierunkowa komunikacja oraz znikomy czas wykonania są doskonałymi argumentami, aby dopuścić się takiego zabiegu.

Komunikacja na przykładzie pozyskiwania odpowiedzi skokowej Komunikacja z komputerem następuje przy użyciu kabla USB, który równocześnie służy do programowania oraz zasilania mikrokontrolera. Jest to możliwe dzięki oprogramowaniu zawartym na programatorze ST-LINK/V2-1 (zaimplementowanym na mikrokontrolerze STM32F103CBT6), który po zakończeniu programowania mikrokontrolera może być wykorzystany jako *Virtual Com Port*, z czego właśnie korzystamy. Do komunikacji została wykorzystana prędkość 115200, 8 bitów na treść, brak bitów parzystości oraz 1 bit stopu. Dodatkowo należy pamiętać o tym, aby nie korzystać ze sprzętowej kontroli przepływu.

Poniżej znajduje się **przykładowy** kod w języku MATLAB, który pozwala na wyświetlanie tego, co wysłał mikrokontroler:

```
delete(instrfindall); % zamkniecie wszystkich polaczen szeregowych
clear all;
close all;
s = serial('COM9'); % COM9 to jest port utworzony przez mikrokontroler
set(s,'BaudRate',115200);
set(s,'StopBits',1);
set(s,'Parity','none');
set(s,'DataBits',8);
set(s,'Timeout',1);
set(s,'InputBufferSize',1000);
set(s,'Terminator',13);
fopen(s); % otwarcie kanalu komunikacyjnego

y(1:100) = -1; % mikrokontroler zwraca tylko dodatnie wartosci, wiec kazde -1
% oznacza, ze nie otrzymalismy jeszcze wartosci o tym indeksie
while true
        txt = fread(s,16);  % odczytanie z portu szeregowego
        eval(char(txt'));   % wykonajmy to co otrzymalismy
        if(y(1) ~= -1)
                break;
        end
end
y(2:100) = -1; % ignorujemy wszystko co odczytalismy poza pierwszym elementem
while true
        txt = fread(s,16);  % odczytanie z portu szeregowego
        eval(char(txt'));   % wykonajmy to co otrzymalismy
```

```
        if(min(y) ~= -1)    % jesli najmniejszym elementem nie jest -1, to znaczy ze
                break;          % nie ma brakujacych elementow, a wiec dane sa gotowe
        end
end
figure;
plot(0:length(y),y); % w tym momencie mozna juz wyrysowac dane
```

W kodzie tym zakładamy, że mikrokontroler przesyła wiadomości o treści `"y(%3d) = %4d;\n\r"`, gdzie pierwszą wartością w tym formacie jest numer chwili począwszy od zmiany wartości sterowania, której dotyczy pomiar, natomiast druga wartość to właśnie pomiar bezpośrednio odczytany z ADC (a więc o wartościach od 0 do 4095). Założone zostało, że po uruchomieniu mikrokontroler wysyła do DAC wartość 1500, następnie po odczekaniu 1s (aby wyjście obiektu zdążyło się ustabilizować) do DAC wysyłana jest wartość 2500 a kolejne pomiary przesyłane są do komputera zgodnie z powyższym formatem. Po przesłaniu 100 kolejnych wartości program rozpoczyna działanie od początku. W ten sposób udało się uzyskać serię pomiarów o długości 100.

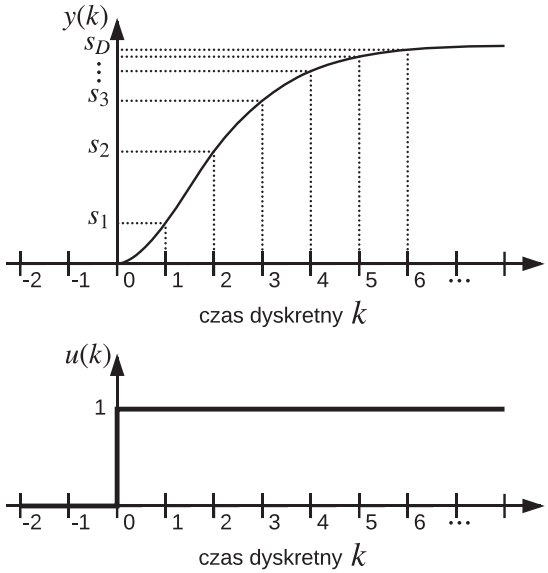
Tutaj warto zwrócić uwagę na kwestię komunikacji mikrokontrolera z komputerem – czy uruchomić najpierw skrypt MATLAB'a czy mikrokontroler jest bardzo ważnym pytaniem, które należy sobie zadać. Jeśli mikrokontroler chodzi w pętli, bez przerwy wykonując pewne operacje i wysyłający wiadomości, i uruchomiony zostanie skrypt je odbierający, pojawia się pewne ryzyko. Otóż odczytana może zostać nie pierwsza wiadomość, którą planowaliśmy odebrać, a jedna „ze środka". W powyższym skrypcie przyjęte zostało, że właśnie taka sytuacja ma miejsce. W skrypcie tym najpierw oczekujemy na konkretną wiadomość (dokładniej, tę, która zmieni wartość elementowi `y(1)`, a następnie wykonywane jest odbieranie kolejnych wiadomości. Wynika to z faktu, iż skrypt z programem mikrokontrolera zostały zsynchronizowane, tj. od tej pory wiemy jakie wiadomości kolejno będą wysyłane.

Taka odpowiedź skokowa jest jeszcze nieużyteczna – wymaga ona przeskalowania. W algorytmie DMC korzystamy z odpowiedzi będącej reakcją na **jednostkową** zmianę wartości sygnału sterowania (tj. zmianę z 0 na 1). Należy więc sprowadzić posiadaną odpowiedź do takiej postaci poprzez pomniejszenie każdego jej elementu o wartość pomiaru wyjścia obiektu regulowanego w stanie sprzed skoku wartości sterowania. Zależność między czasem aplikacji nowej wartości sterowania a chwilą pomiaru wyjścia obiektu regulacji przedstawia Rys. 49.

Skalowanie pomiarów tak, aby uzyskać odpowiedź skokową polega na odjęciu od wszystkich pomiarów wartości pomiaru wyjścia obiektu regulowanego sprzed zmiany wartości sterowania i podzieleniu wynikowych elementów przez przyrost sterowania. W powyższym przykładzie wartości odczytane z mikrokontrolera zostały zapisane w wektorze `y`, gdzie wartość `y(1)` odpowiada pomiarowi w chwili zmiany wartości sterowania. Oznacza to, że przejście z pomiarów na odpowiedź skokową wymaga następującego przekształcenia w MATLABie:

```
s=(y(2:100)-y(1))/1000; % przeskalowane pomiary = jednostkowa odpowiedz skokowa
```

Przy czym należy pamiętać, że element s_1 jest to przyrost wartości wyjściowej obiektu regulacji w **następnej** chwili po zmianie wartości sterowania (wyraźnie to widać na Rys.49). Z tego właśnie powodu



Rysunek 49: Schemat zależności czasowych między zmianą sygnału sterowania a pomiarem wyjścia obiektu regulowanego

wykorzystany został wektor `y(2:100)` a nie `y(1:100)`.

UWAGA! Tak uzyskana odpowiedź skokowa jest odpowiednia tylko i wyłącznie wtedy, gdy czas obliczeń algorytmu DMC jest znikomy w stosunku do czasu między wykonaniem kolejnych pomiarów (a co za tym idzie między kolejnymi iteracjami algorytmu DMC). Jeśli czas ten nie jest wystarczająco krótki i wprowadzone zostało sztuczne opóźnienie aplikacji do procesu nowej wartości sygnału sterującego, także i odpowiedź skokową należy opóźnić o jeden takt. Sprowadza się to do dodania jednego zera na początku wektora odpowiedzi skokowych. Bez tego obiekt oraz model nie będą ze sobą spójne, co będzie powodować niepoprawne działanie algorytmu. W zależności od tego czy zostało zastosowane sztuczne opóźnienie uzyskana odpowiedź skokowa powinna rozpoczynać się od wartości różnej od zera (w przypadku braku opóźnienia) lub równej zero (w przypadku jego zastosowania).

Komunikacja na przykładzie nieustannego odczytu pomiarów Aby przekazać pomiar sygnału wyjściowego obiektu regulowanego oraz wartość sterowania do komputera, warto rozważyć przesyłanie wiadomości o treści np. `"Y=%4d;U=%4d;"`. Gdzie oczywiście pierwszą liczbą w tym formacie jest odczyt wartości sygnału wyjściowego obiektu regulacji, natomiast drugim jest wartość sygnału sterującego. Taka wiadomość przesyłana by była po każdej iteracji algorytmu regulacji. Ponieważ algorytm regulacji działać będzie w pętli nieskończonej – nie jest ważne, którą wiadomość odbierzemy jako pierwszą. W związku z tym kod skryptu, w którym dokonywane będzie zbieranie danych wygląda następująco:

```
delete(instrfindall); % zamkniecie wszystkich polaczen szeregowych
clear all;
```

```
close all;
s = serial('COM9'); % COM9 to jest port utworzony przez mikrokontroler
set(s,'BaudRate',115200);
set(s,'StopBits',1);
set(s,'Parity','none');
set(s,'DataBits',8);
set(s,'Timeout',1);
set(s,'InputBufferSize',1000);
set(s,'Terminator',13);
fopen(s); % otwarcie kanalu komunikacyjnego

Tp = 0.01; % czas z jakim probkuje regulator
y = [];    % wektor wyjsc obiektu
u = [];    % wektor wejsc (sterowan) obiektu
while length(y)~=100    % zbieramy 100 pomiarow
txt = fread(s,14);  % odczytanie z portu szeregowego
                    % txt powinien zawierać Y=%4d;U=%4d;
                    % czyli np. Y=1234;U=3232;
eval(char(txt'));   % wykonajmy to co otrzymalismy
y=[y;Y];            % powiekszamy wektor y o element Y
u=[u;U];            % powiekszamy wektor u o element U
end

figure; plot((0:(length(y)-1))*Tp,y); % wyswietlamy y w czasie
figure; plot((0:(length(u)-1))*Tp,u); % wyswietlamy u w czasie
```

Czyli jak widać utworzone przez wywołanie funkcji `eval` zmienne `Y` oraz `U` są dodawane do odpowiednich tablic zgodnie z zamysłem użytkownika skryptu. Poprzedni skrypt zawierał już informację o tym, który element wektora `y` należy uzupełnić odczytem z ADC, co pozwalało na synchronizację. Ponieważ tutaj synchronizacja nie jest potrzebna/konieczna – każdą wiadomość traktujemy identycznie.

5.1 Zadanie do wykonania

Zadaniem, które będzie podlegało późniejszej ocenie jest implementacja na mikrokontrolerze rodziny STM32 dwóch algorytmów regulacji. Pierwszym jest algorytm regulacji PID (*Proportional-Integral-Derivative*), drugim DMC (*Dynamic Matrix Control*) w wersji analitycznej. Zrealizowana implementacja będzie podstawą do realizacji następnego projektu, tak więc musi być bezbłędnie zrealizowana.

Projekt kończy się sprawozdaniem (wysłać je należy do niedzieli następującej po ostatnich zajęciach z tego projektu, do godziny 23:59). W sprawozdaniu należy poruszyć następujące kwestie:

- Algorytm PID:
  - Porównanie trajektorii sygnału wyjściowego procesu regulowanego przy wyznaczeniu nastaw

metodą Zieglera-Nicholsa.

  - Porównanie trajektorii sygnału wyjściowego procesu regulowanego przy wyznaczeniu nastaw metodą „inżynierską".
  - Porównanie trajektorii sygnału wyjściowego procesu regulowanego w zależności od parametru T_v (tj. parametru związanego z algorytmem *anti-windup*)

- Algorytm DMC:
  - Porównanie trajektorii sygnału wyjściowego procesu regulowanego przy zastosowaniu różnych wartości Λ.
  - Porównanie trajektorii sygnału wyjściowego procesu regulowanego przy zastosowaniu różnych wartości horyzontu sterowania.
  - Porównanie trajektorii sygnału wyjściowego procesu regulowanego przy zastosowaniu różnych wartości horyzontu predykcji.

- Porównanie najlepszej realizacji algorytmu PID i DMC:
  - Odporność na zakłócenia (realizowane poprzez wciśnięcie klawisza na płycie z obiektem symulowanym).
  - Przesterowanie.
  - Czas ustalenia.
  - Oscylacje (ocena „na oko").

Oceniane będzie zarówno sprawozdanie jak i kod algorytmów. Błędny kod algorytmu regulacji będzie powodował wyzerowanie oceny za opis dotyczący tego algorytmu. Pod warunkiem poprawności implementacji algorytmów punkty będą przyznawane następująco:

- 1pkt za omówienie wpływu parametrów algorytmu PID na jakość regulacji,
- 2pkt za omówienie wpływu parametrów algorytmu DMC na jakość regulacji,
- 2pkt za porównanie jakości regulacji algorytmów PID i DMC z najlepiej sprawdzającymi się dla nich parametrami.

Przewidywane są oceny cząstkowe z krokiem 0,25 pkt. Określenie „omówienie" oznacza w powyższym tekście zarówno opis przeprowadzonych eksperymentów, jak i wnioski wynikające z wyników – mogą być one zaskakująco różne od oczekiwanych. Warto w takiej sytuacji zastanowić się co może być przyczyną i również to opisać. W razie większych wątpliwości – warto skonsultować się z prowadzącym.

5.2 Sugerowany przebieg projektu

Sugerowanym podejściem do projektu jest implementacja na pierwszych zajęciach algorytmu PID oraz akwizycja odpowiedzi skokowej. Między zajęciami warto zaimplementować w środowisku MATLAB (a najlepiej w C) algorytm DMC, aby na drugie zajęcia przyjść już z gotowym, działającym i przetestowanym

kodem. Jest to również dobry czas na zaplanowanie sprawozdania i wykonanych do niego eksperymentów. Drugie zajęcia warto spędzić na pozyskiwaniu przebiegów eksperymentów (dla PID zmiany członów K, T_I, T_D, T_v, a dla DMC zmiany λ, N, N_u, reakcja na zakłócenie wyjścia itp.). Sprawozdanie warto dokończyć po drugich zajęciach.

Powyższa kolejność jest wyłącznie sugestią – poczynania studentów w ramach projektu są uzależnione tylko i wyłącznie od decyzji uczestników projektu.