SMS
Sprawozdanie z Projektu I

Krystian Chachuła
Marcin Dolicher

AIR Semestr V

Spis treści

1 Zagadnienia i założenia projektowe

Postawione przed nami zadanie polegało na zaprojektowaniu regulatora PID i DMC, które sterują obiektem zrealizowanym na mikrokontrolerach z serii STM32. Powinniśmy tak manipulować sygnałem wejściowym procesu u, aby wartość sygnału wyjściowego procesu (regulowanego) y była możliwie bliska wartośći zadanej y^{zad}. Wartość uchybu $e = y^{zad} - y$ powinna być jak najmniejsza. Wyniki uzyskane podczas eksperymentów zostaną porównane i poddane krytycznej weryfikacji.

2 Algorytm PID

2.1 Omówienie implementacji

Tradycyjnie regulację za pomocą algorytmu PID realizujemy za pomocą trzech członów proporcjonalnego, całkującego i różniczkującego. Człon proporcjonalny powoduje wzrost wartości sterowania wraz z wzrostem uchybu, całkujący zwiększa wartość sygnału sterującego wraz z akumulowanym uchybem, a dla różniczkującego wraz z wzrostem uchybu, wzrasta wartość sygnału sterującego.

Implementacji algorytmu dokonaliśmy w plikach `Pid.c` i `Pid.h`. Parametry PID-a zostały w kodzie zaprezentowane jako struktura *Pid*. W pliku `main.c` zadajemy wartości odpowiednim parametrom z struktury PID. Wymagane obliczenia w algorytmie są realizowane za pomocą funkcji `float pidCe(Pid *pid, float pv)`, której argumentami są struktura z wartościami naszego PID-a i zmienna *pv - process value*, czyli naszą wartość zadaną, a funkcja zwraca nam sygnał sterujący.

W każdym wywołoniu funkcji dokonujemy następujących obliczeń:

1. Wyliczamy uchyb na podstawie wzoru: $e(k) = y^{zad}(k) - y(k)$
2. Wartość członu proporcjonalnego $u(P) = Ke(k)$
3. Wartość członu całkującego $u_I(k) = u_I(k-1) + \frac{K}{T_I} T \frac{e(k-1) + e(k)}{2}$
4. Wartość członu różniczkującego $u_D(k) = KT_D \frac{e(k) - e(k-1)}{T}$

5. Następuje zapisanie wartości z stanu k jako wartości dla stanu k-1 (w naszym kodzie zmienne z poprzedniego stanu wyrażone są za pomocą przedrostka prev)

Oprócz tych kroków do naszego algorytmu zastosowaliśmy rozwiązanie anti-windup. Rozwiązania tego używamy w przypadku gdy zmienna sterowania osiąga wartość graniczną urządzenia wykonawczego. Wiemy, że nie ma sensu zadawać większej wartości sygnału sterowania niż element wykonawczy jest w stanie zrealizować. W takiej sytuacji przerywamy pętlę sprzężenia zwrotnego i system zaczyna pracę w pętli otwartej. Takie rozwiązanie zapobiega „nawijaniu" członu całkującego, czyli osiąganiu nadzwyczaj dużych wartości członu całkującego co prowadzi do ogromnego spowolnienia działania regulatora, a w skrajnych przypadkach do jego rozregulowania.

Odp skokowa !!!!!!!!!!!!!!!!?????????????????????? Dostrajanie regulatora odbywało się na zasadzie pozyskiwania odpowiedz skokowej. Obserwując w jaki sposób sygnał sterujący generowany przez regulator osiąga wartość zadaną podczas skoku dokonywaliśmy oceny regulacji. W ten sposoób wybieraliśmy najlepsze nastawy dla regulatora.

2.2 Wyznaczanie nastawów regulatora metodą Zieglera-Nicholsa

Przbieg strojenia regulatora przy użyciu metody Zieglera-Nicholsa

1. Implementujemy regulator typu P.
2. Wartość wzmocnienia K dobieramy tak aby wyjście obiektu regulacji miało charakter oscylacyjny (nierosnący, niemalejący). Przyjmujemy wzmocnienie krytyczne $K_u = K$. Odczytujemy jeszcze okres oscylacji T_u.
3. Używając tabelki wyliczamy parametry K, T_I, T_D w zależności od regulatora który chcemy stosować P, PI, PID. My oczywiście wybiermay wzory dla PID.
4. Wyznaczone parametry powinny zapewnić niezłą jakość regulacji, gdy będziemy chcieli spróbować znaleźć lepszy regulator zaczęcie od nastawów wyznaczonych metodą Zieglera-Nicholsa będzie dobrym pomysłem.

3 Algorytm DMC

4 Porównanie najlepszych realizacji PID i DMC

4.1 f)

5 Testowanie

6 Wnioski