Informatyczne Systemy Sterowania

Ćwiczenie 2: Systemy regulacji - Regulator PID

Krzysztof Przybylski 239266

13 stycznia 2019

1 Wstęp

1.1 Cel ćwiczenia

Celem tego ćwiczenia jest poznanie podstawowej struktury systemu sterowania z typową formą algorytmu regulacji PID, oraz zapoznanie się z częścią pakietu MATLAB - Simulinkiem, oraz jego możliwości w zakresie modelowania i analizy systemów regulacji.

1.2 Plan badań

1. Symulacja systemu regulacji
   (a) Regulator P. Zbadać wpływ wartości wzmocnienia regulatora na działanie systemu.
   (b) Regulator PI. Zbadać wpływ wartości parametru całkowania na działanie systemu (dla ustalonej wartości parametru k).
   (c) Regulator PID. Zbadać wpływ wartości parametru różniczkowania na działanie systemu (dla ustalonych wartości parametrów k i T_i).
2. Dobór optymalnych parametrów regulatora. Sporządzenie wykresów:
   (a) dla regulatora P.
   (b) dla regulatora PI przy trzech różnych ustalonych wartościach parametru k.
   (c) dla regulatora PID przy trzech różnych wartościach parametru oraz ustalonej wartości parametru k.
3. Dobór parametrów regulatora PID według zasad Zieglera–Nicholsa
   (a) Doświadczalnie znaleźć współczynnik wzmocnienia, dla którego układ traci stabilność.
   (b) Ustalić okres oscylacji i wzmocnienie krytyczne.
   (c) Według odpowiednich rekomendacji określić wartości parametrów regulatora PID.

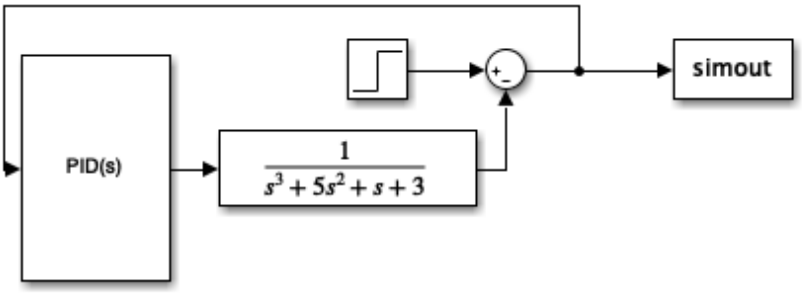
Zadania należy wykonać dla obiektu o podanej transmitancji operatorowej:

$$G(s) = \frac{b}{s^3 + a_2 s^2 + a_1 s + a_0} \tag{1}$$

2 Realizacja planu i wyniki

2.1 Symulacja systemu regulacji.

System regulacji będziemy symulować przy użyciu programu Simulink będącego częścią pakietu MATLAB. Schemat systemu służącego nam do symulacji przedstawiony został na poniższym rysunku.



Rys. 1: Schemat symulacji obiektu z regulatorem PID

Wartości błędu regulacji $\varepsilon(t)$ mogą być odczytywane zarówno na wykresie *Scope* jak i z poziomu MATLABA dzięki obiektowi *simout*.

Regulator PID jest opisany następującą transmitancją:

$$G(s) = k + \frac{T_i}{s} + T_d s, \tag{2}$$

gdzie k to współczynnik wzmocnienia, T_i to parametr członu całkującego, a T_d to parametr członu różniczkującego.

Podczas ćwiczenia przyjęliśmy następujące wartości parametrów transmitancji obiektu sterowania (4):

$$b = 1, a_2 = 5, a_1 = 1, a_0 = 3 \tag{3}$$

oraz zadanej wartości wyjścia $y^* = 1$.

2.1.1 Regulator P.

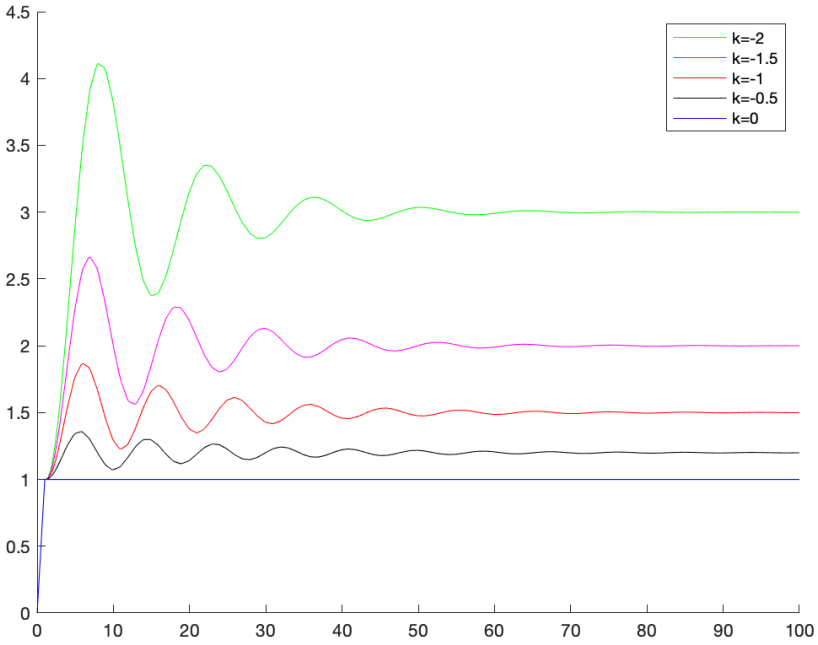
Aby zasymulować regulator P musieliśmy ustawić wartości parametru całkującego i różniczkującego na $T_i = 0$, $T_d = 0$.
Następnie wykorzystując poniższą funkcję przetestowaliśmy zachowanie regulatora P dla różnych wartości parametru k.

```
function testP(start, step, stop)
load_system('pidModel.mdl');
hold on;
k = start;
color = char('g', 'm', 'r', 'k', 'b', 'y');
set_param('pidModel/PID Controller', 'I', num2str(0));
set_param('pidModel/PID Controller', 'D', num2str(0));
i = 0;
legend('on');
while (k <= stop)
    set_param('pidModel/PID Controller', 'P', num2str(k));
    sim('pidModel.mdl');
    figure(1);
    plot(simout.time, simout.signals.values, 'Color', color(mod(i,6)+1), 'DisplayName', strcat↩
        ('k=',num2str(k)));
    i = i + 1;
    k = k + step;
end
end
```

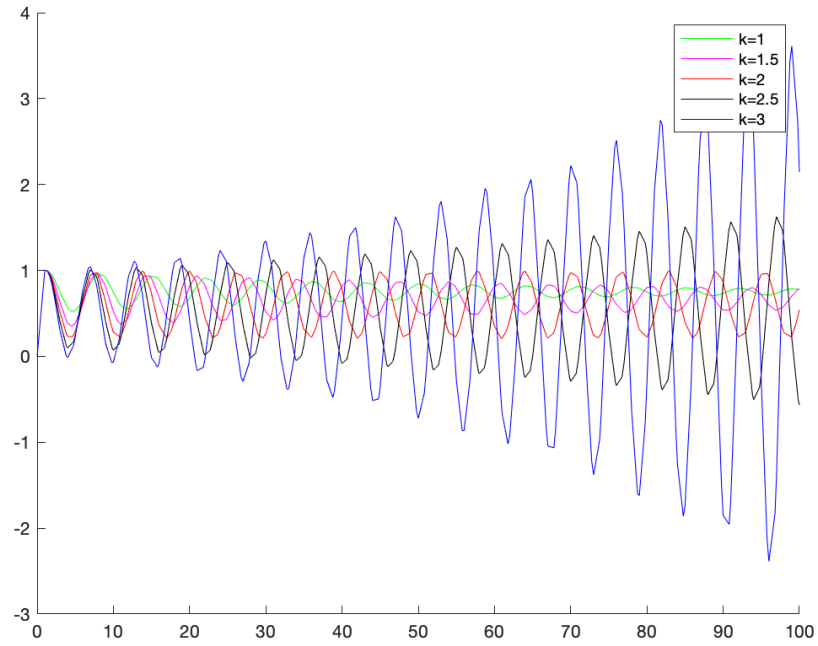
Fun. 1: Funkcja testująca regulator P.

W ten sposób otrzymalismy następujące wykresy:

Z wykresu 2, oraz 3 możemy zauważyć, że:



Rys. 2: Wykres przebiegu funkcji $\varepsilon(t)$ regulatora P dla wartości ujemnych k.



Rys. 3: Wykres przebiegu funkcji $\varepsilon(t)$ regulatora P dla wartości dodatnich k.

- Dla ujemnych wartości parametru k drgania po pewnym czasie ustają, a wykres $\varepsilon(t)$ stabilizuje się na pewnym poziomie większym od żądanej wartości $y*$, a poziom ten jest tym większy im mniejsza jest wartość parametru k.
- Dla wartości dodatnich wykres $\varepsilon(t)$ przyjmuje postać drgań oscylujących wokół wartości 0.6. Dla wartości k mniejszych od 2 amplituda tych drgań maleje, natomiast rośnie dla wartości większych od 2. Dla $k = 2$ amplituda tych drgań jest stała.
- Regulator P nie jest efektywny, ponieważ wykres funkcji $\varepsilon(t)$ oscyluje poniżej wartości $y*$, ale nie

wokół 0.

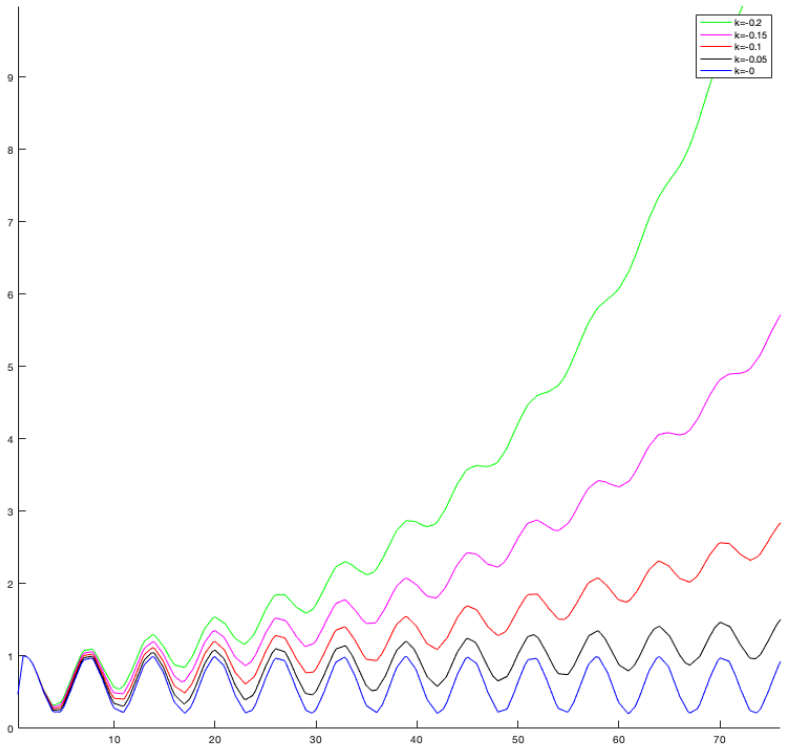
2.1.2 Regulator PI.

Aby zasymulować regulator PI musieliśmy ustawić wartości parametru proporcjonalnego i różniczkującego na $k = 2$, $T_d = 0$.
Następnie wykorzystując poniższą funkcję przetestowaliśmy zachowanie regulatora PI dla różnych wartości parametru $T_i = 0$.

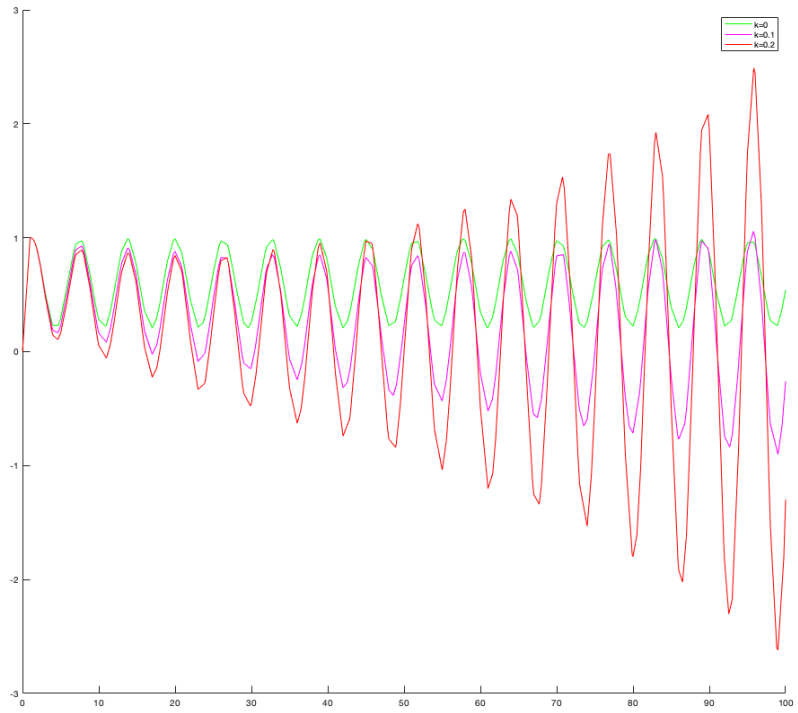
```
function testPI(start, step, stop)
load_system('pidModel.mdl');
hold on;
ti = start;
i = 0;
color = char('g', 'm', 'r', 'k', 'b', 'y');
set_param('pidModel/PID Controller', 'P', num2str(2));
set_param('pidModel/PID Controller', 'D', num2str(0));
legend('on');
while (ti <= stop)
    set_param('pidModel/PID Controller', 'I', num2str(ti));
    sim('pidModel.mdl');
    figure(1);
    plot(simout.time, simout.signals.values, 'Color', color(mod(i,6)+1), 'DisplayName', strcat↩
        ('k=',num2str(ti)));
    i = i + 1;
    ti = ti + step;
end
end
```

Fun. 2: Funkcja testująca regulator PI.

W ten sposób otrzymalismy następujące wykresy:



Rys. 4: Wykres przebiegu funkcji $\varepsilon(t)$ regulatora PI dla wartości ujemnych T_i.



Rys. 5: Wykres przebiegu funkcji $\varepsilon(t)$ regulatora PI dla wartości dodatnich T_i.

Z wykresu 4, oraz 5 możemy zauważyć, że:

- Dla ujemnych wartości parametru T_i wartości funkcji $\varepsilon(t)$ szybko rosną. Tempo w jakim wzrasta jest tym większe im mniejszą wartość ma parametr T_i.
- Dla wartości dodatnich wykres $\varepsilon(t)$ przyjmuje postać drgań, które z początku oscylują tak jak przy regulatorze P, lecz z czasem oscylują wokół wartości coraz bliższej 0. Tempo w jakim zbliża się do 0 jest tym większe im większą wartość ma parametr T_i. Amplituda drgań z czasem wzrasta.
- Regulator PI jest bardziej efektywny niż regulator P, ponieważ wykres funkcji $\varepsilon(t)$ oscyluje bliżej 0.

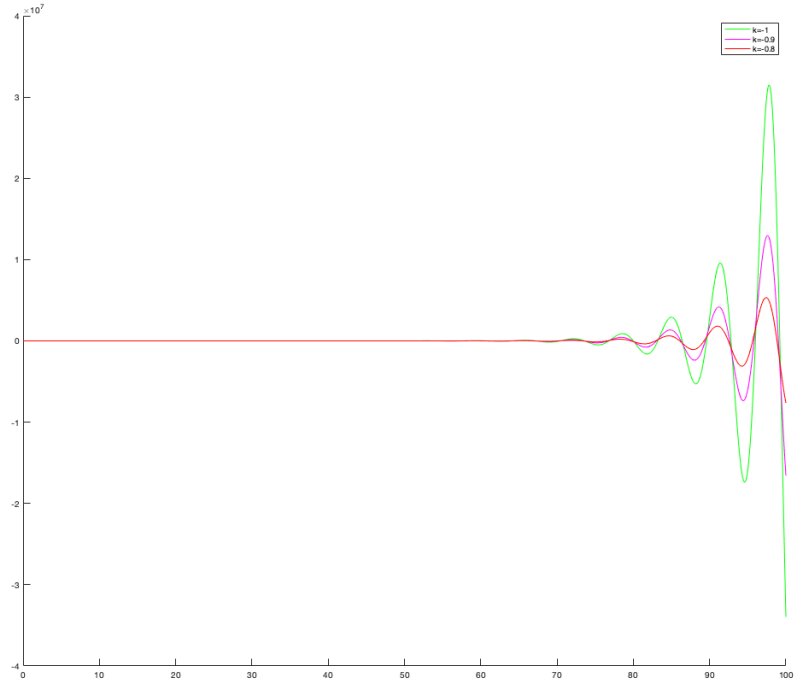
2.1.3 Regulator PID.

Aby zasymulować regulator PID musieliśmy ustawić wartości parametru proporcjonalnego i całkującego na $k = 2$, $T_i = 1$.
Następnie wykorzystując poniższą funkcję przetestowaliśmy zachowanie regulatora P dla różnych wartości parametru T_d.

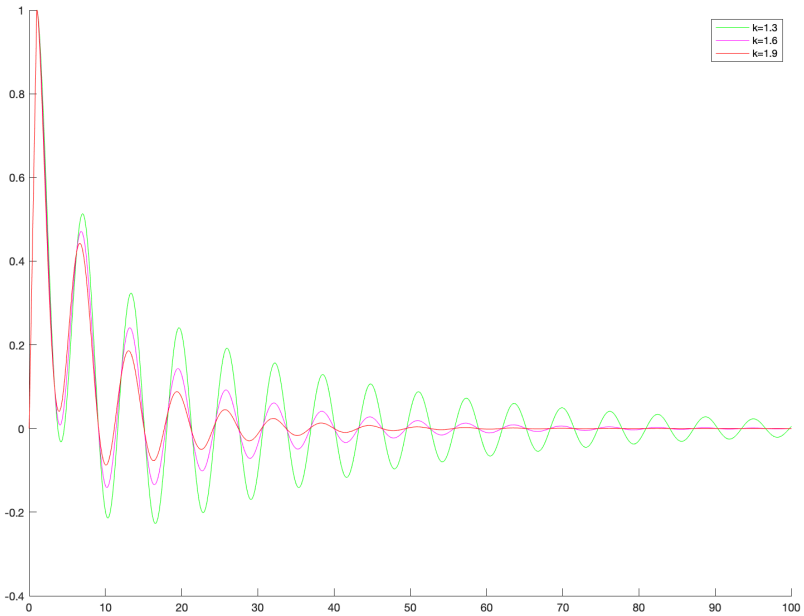
```
function testPID(start, step, stop)
load_system('pidModel.mdl');
hold on;
td = start;
i = 0;
color = char('g', 'm', 'r', 'k', 'b', 'y');
set_param('pidModel/PID Controller', 'P', num2str(2));
set_param('pidModel/PID Controller', 'I', num2str(1));
legend('on');
while (td <= stop)
    set_param('pidModel/PID Controller', 'D', num2str(td));
    sim('pidModel.mdl');
    figure(1);
    plot(simout.time, simout.signals.values, 'Color', color(mod(i,6)+1), 'DisplayName', strcat↩
        ('k=',num2str(td)));
    i = i + 1;
    td = td + step;
end
end
```

Fun. 3: Funkcja testująca regulator PID.

W ten sposób otrzymalismy następujące wykresy:



Rys. 6: Wykres przebiegu funkcji $\varepsilon(t)$ regulatora PID dla wartości ujemnych T_d.



Rys. 7: Wykres przebiegu funkcji $\varepsilon(t)$ regulatora PID dla wartości dodatnich T_d.

Z wykresu 6, oraz 7 możemy zauważyć, że:

- Dla ujemnych wartości parametru k wykres $\varepsilon(t)$ przyjmuje postać drgań, których amplituda z czasem rośnie. Tempo tego wzrostu jest tym większe im mniejszą wartość ma parametr T_d.
- Dla wartości dodatnich wykres $\varepsilon(t)$ przyjmuje postać drgań, które oscylują tak jak przy regulatorze PI, lecz z czasem amplituda maleje. Tempo w jakim ona maleje tym większe im większą wartość ma parametr T_d.
- Regulator PID jest efektywny, ponieważ wykres funkcji $\varepsilon(t)$ oscyluje bliżej 0 niż regulator P, a amplituda drgań, która w regulatorze PI się zwiększała tutaj zanika.

2.2 Dobór optymalnych parametrów regulatora.

Przy doborze optymalnych parametrów przyjęliśmy następujące kryterium:

$$Q = \frac{1}{n}\sum_{i=1}^{n}\varepsilon(t) \tag{4}$$

Gdzie n jest ilością pomiarów wykonanych przy symulacji.

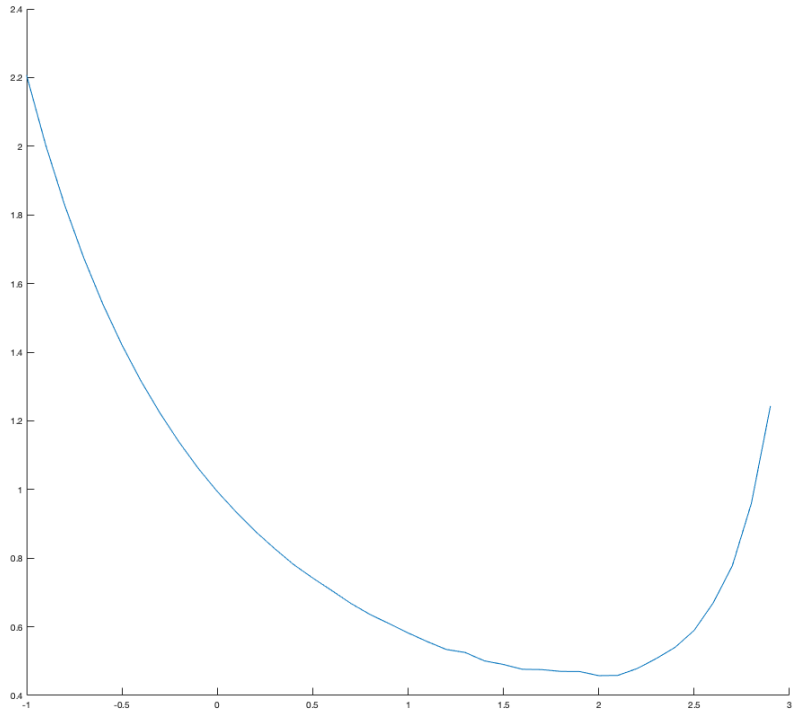
2.2.1 Regulator P.

Aby dobrać optymalną wartość parametru k ustawiliśmy parametry podobnie jak w zadaniu 2.1.1, oraz wykorzystując poniższą funkcję przetestowaliśmy zachowanie regulatora P dla wielu wartości parametru k.

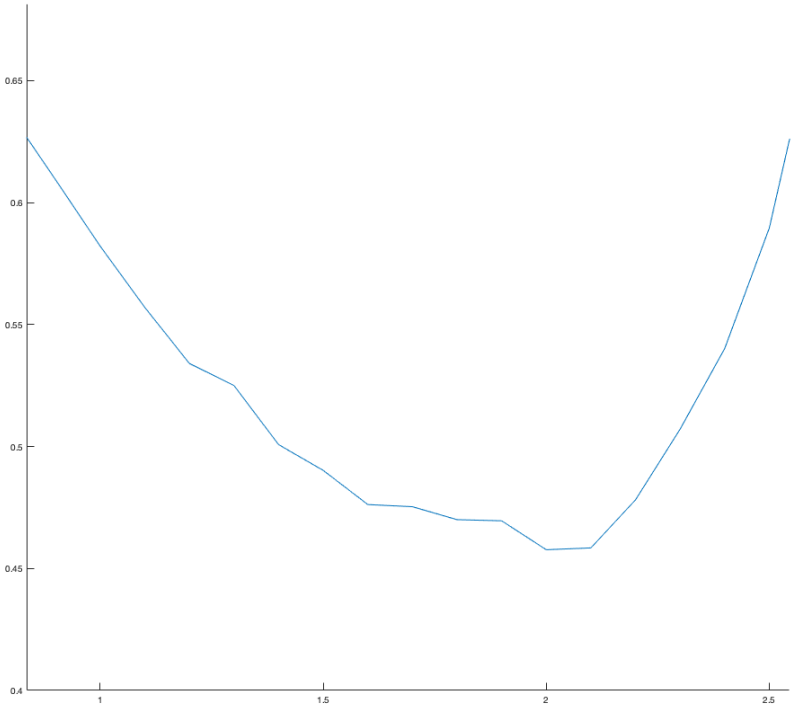
```
function optimumP(start, step, stop)
load_system('pidModel.mdl');
hold on;
k = start;
i = 0;
set_param('pidModel/PID Controller', 'I', num2str(0));
set_param('pidModel/PID Controller', 'D', num2str(0));
while (k <= stop)
    set_param('pidModel/PID Controller', 'P', num2str(k));
    sim('pidModel.mdl');
    wy = simout.signals.values;
    i = i + 1;
    q(i) = sum(wy.^2)/length(wy);
    ka(i) = k;
    k = k + step;
end
figure(1);
plot(ka, q);
end
```

Fun. 4: Funkcja testująca regulator P.

W ten sposób otrzymalismy następujące wykresy:



Rys. 8: Wykres przebiegu funkcji $Q(k)$ regulatora P.



Rys. 9: Wykres przebiegu funkcji $Q(k)$ regulatora P (powiększenie).

Wartością optymalną parametru k jest minimum funkcji $Q(k)$, więc z wykresu 8, oraz 9 możemy odczytać, że $k = 2$.

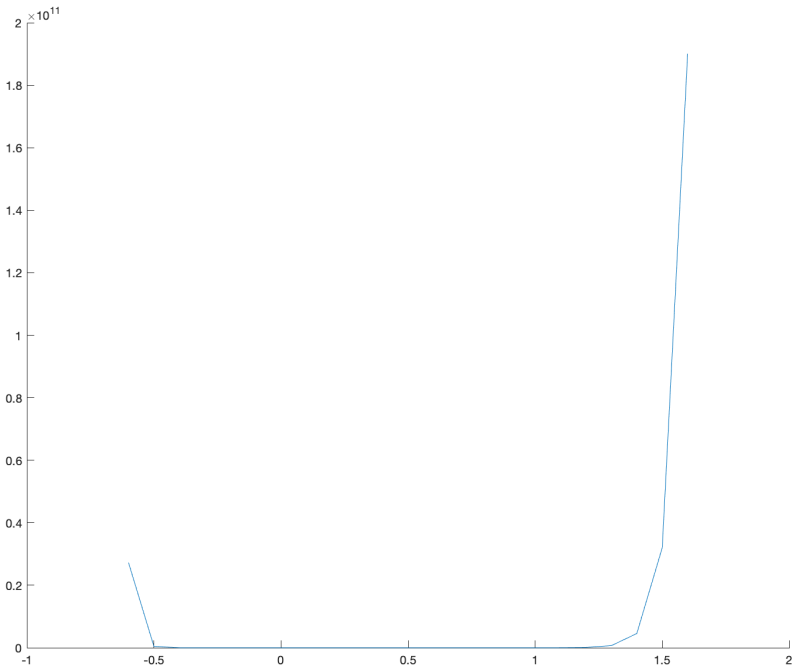
2.2.2 Regulator PI.

Aby dobrać optymalną wartość parametru T_i ustawiliśmy parametry podobnie jak w zadaniu 2.1.2, oraz wykorzystując poniższą funkcję przetestowaliśmy zachowanie regulatora P dla wielu wartości parametru T_i i trzech różnych wartości k.

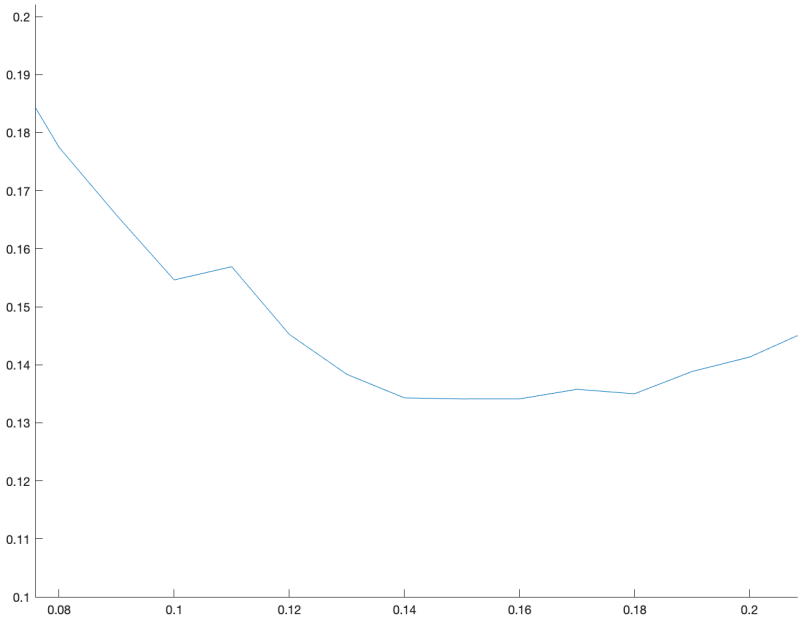
```
function optimumPI(start, step, stop)
load_system('pidModel.mdl');
hold on;
ti = start;
i = 0;
set_param('pidModel/PID Controller', 'D', num2str(0));
set_param('pidModel/PID Controller', 'P', num2str(4));
while (ti <= stop)
    set_param('pidModel/PID Controller', 'I', num2str(ti));
    sim('pidModel.mdl');
    wy = simout.signals.values;
    i = i + 1;
    q(i) = sum(wy.^2)/length(wy);
    tei(i) = ti;
    ti = ti + step;
end
figure(1);
plot(tei, q);
end
```

Fun. 5: Funkcja testująca regulator PI.

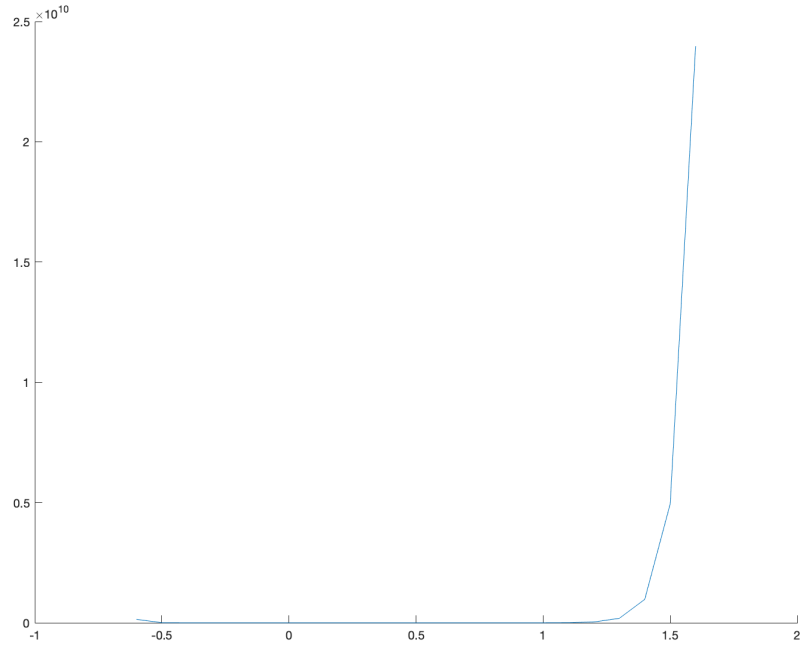
W ten sposób otrzymalismy następujące wykresy:



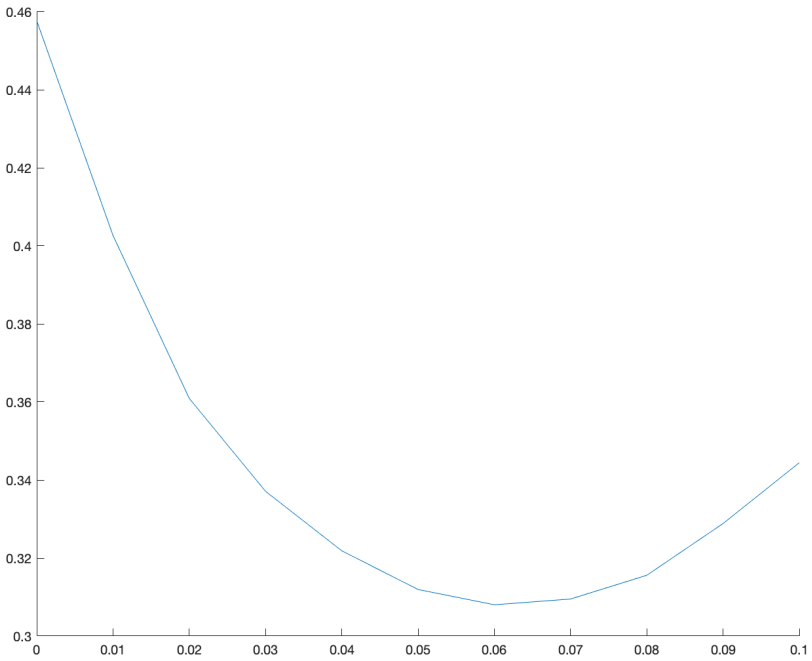
Rys. 10: Wykres przebiegu funkcji $Q(T_i)$ regulatora PI dla $k = 1$.



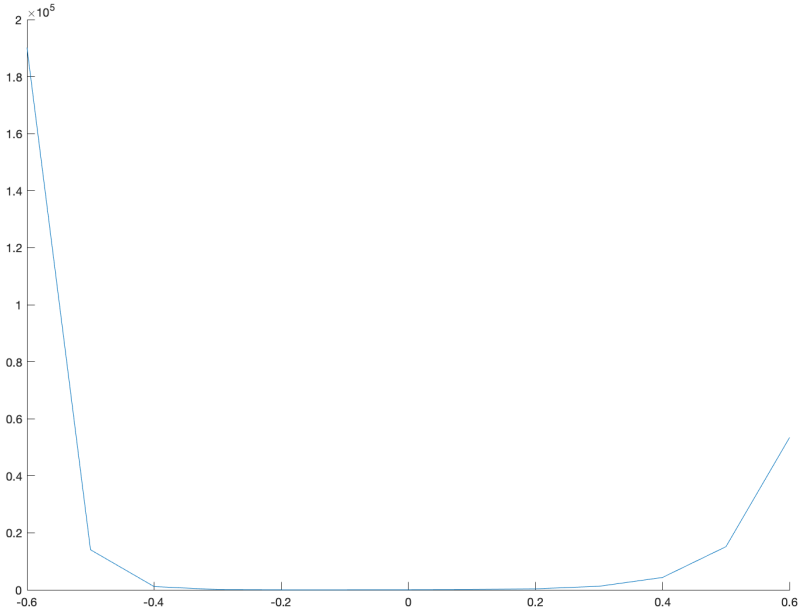
Rys. 11: Wykres przebiegu funkcji $Q(T_i)$ regulatora PI dla $k = 1$ (powiększenie).



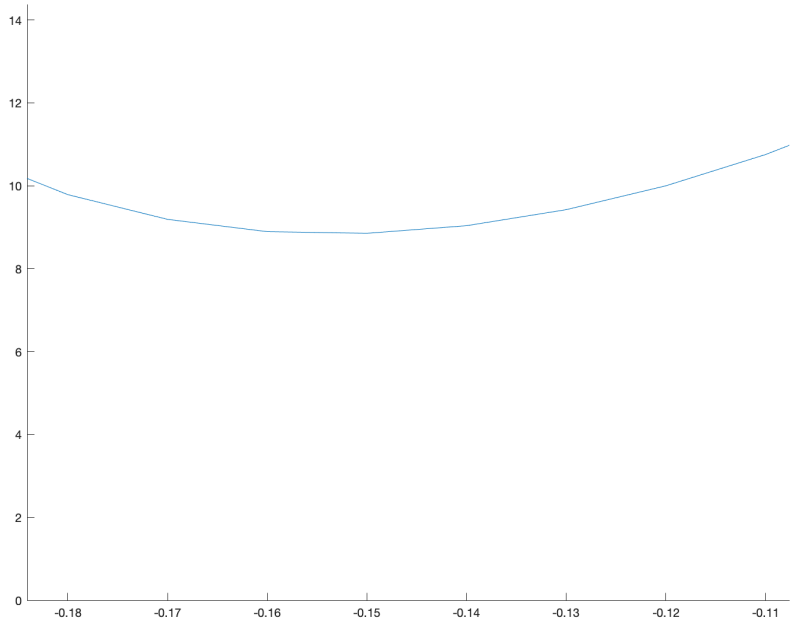
Rys. 12: Wykres przebiegu funkcji $Q(T_i)$ regulatora PI dla $k = 2$.



Rys. 13: Wykres przebiegu funkcji $Q(T_i)$ regulatora PI dla $k = 2$ (powiększenie).



Rys. 14: Wykres przebiegu funkcji $Q(T_i)$ regulatora PI dla $k = 4$.



Rys. 15: Wykres przebiegu funkcji $Q(T_i)$ regulatora PI dla $k = 4$ (powiększenie).

Wartością optymalną parametru T_i jest minimum funkcji $Q(T_i)$, więc z wykresów 10, 11, 12, 13, 14, oraz 15 możemy odczytać, że dla $k = 1, T_i \approx 0.14$, dla $k = 2, T_i \approx 0.06$, natomiast dla $k = 4, T_i \approx -0.15$.

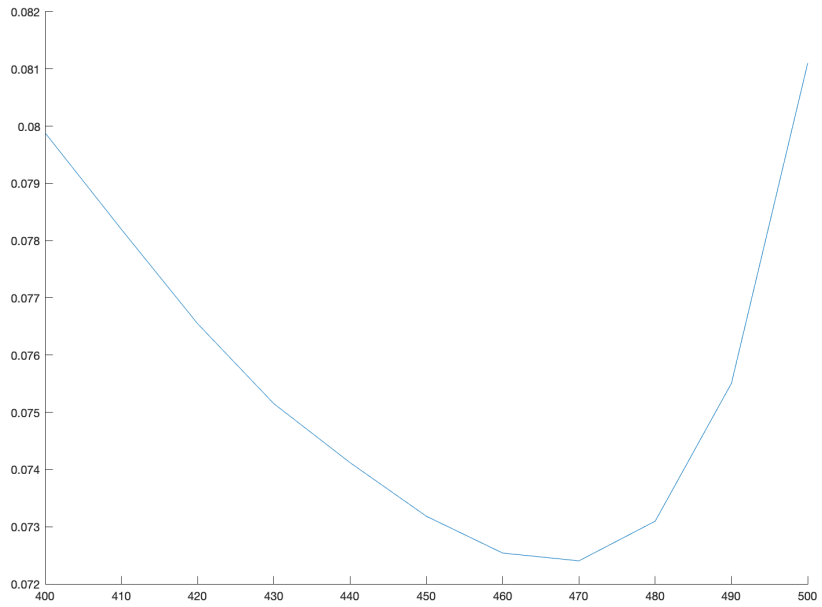
2.2.3 Regulator PID.

Aby dobrać optymalną wartość parametru T_d ustawiliśmy parametry podobnie jak w zadaniu 2.1.3, oraz wykorzystując poniższą funkcję przetestowaliśmy zachowanie regulatora P dla wielu wartości parametru T_d i trzech różnych wartości T_i.

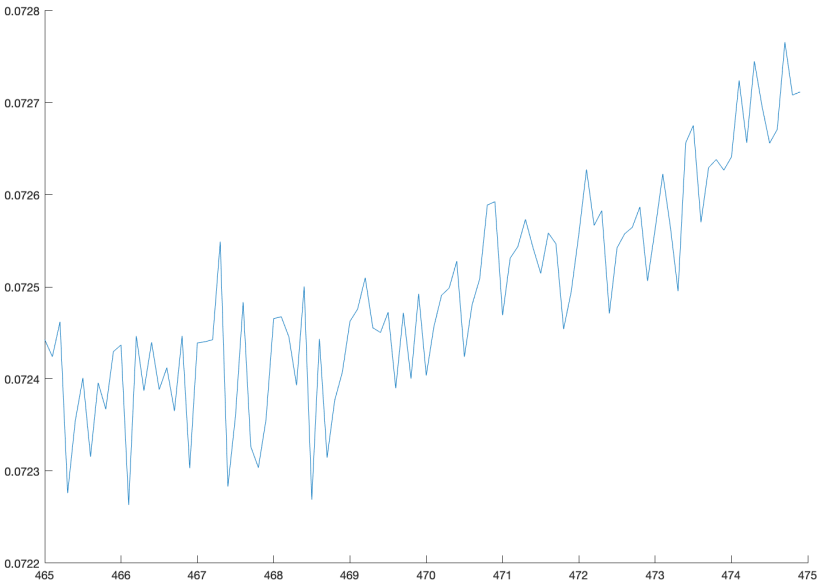
```
function optimumPID(start, step, stop)
load_system('pidModel.mdl');
hold on;
td = start;
i = 0;
set_param('pidModel/PID Controller', 'P', num2str(2));
set_param('pidModel/PID Controller', 'I', num2str(0.06));
while (td <= stop)
    set_param('pidModel/PID Controller', 'D', num2str(td));
    sim('pidModel.mdl');
    wy = simout.signals.values;
    i = i + 1;
    q(i) = sum(wy.^2)/length(wy);
    tede(i) = td;
    td = td + step;
end
figure(1);
plot(tede, q);
end
```

Fun. 6: Funkcja testująca regulator PID.

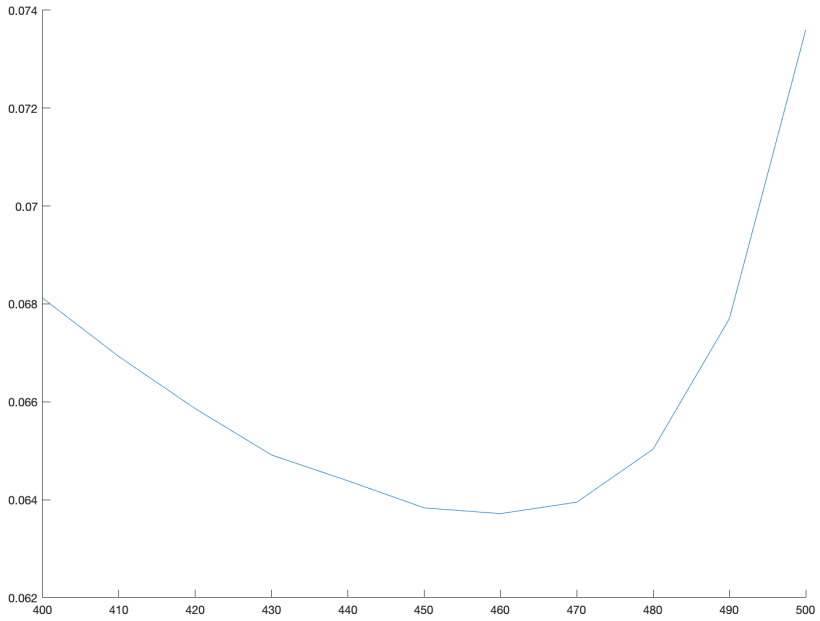
W ten sposób otrzymalismy następujące wykresy:



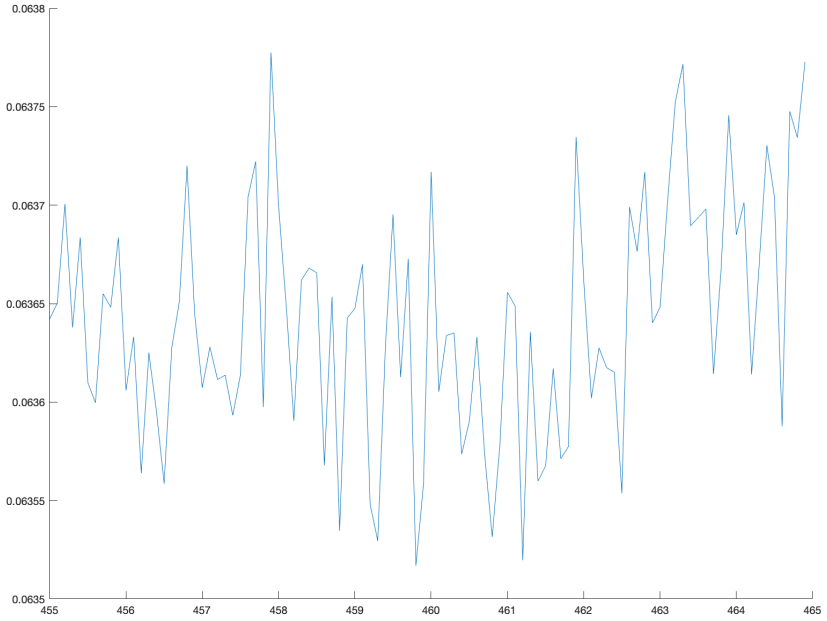
Rys. 16: Wykres przebiegu funkcji $Q(T_d)$ regulatora PID dla $T_i = 0.02$.



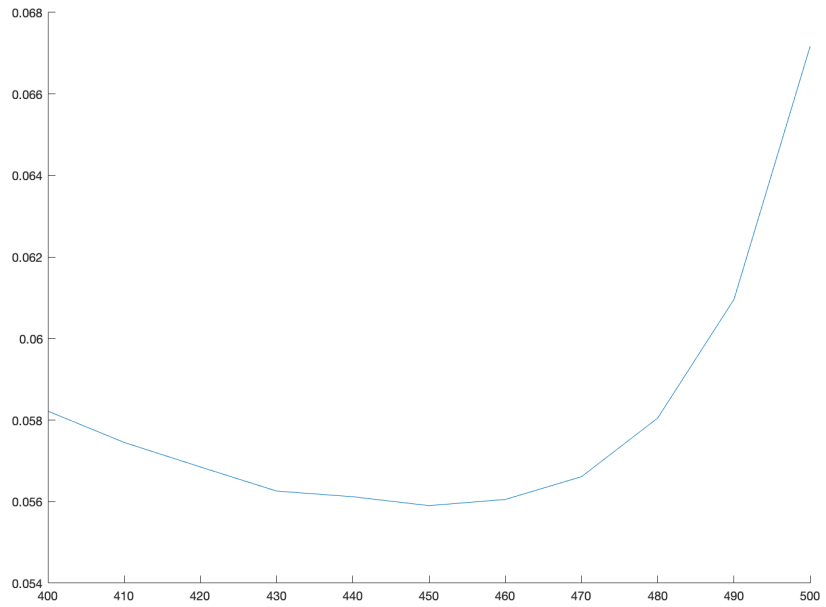
Rys. 17: Wykres przebiegu funkcji$Q(T_d)$ regulatora PID dla $T_i = 0.02$ (powiększenie).



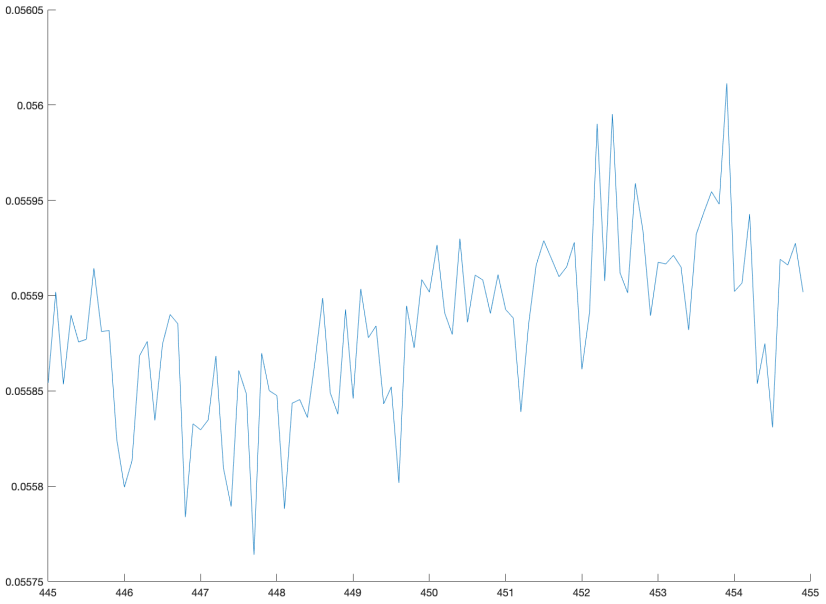
Rys. 18: Wykres przebiegu funkcji $Q(T_d)$ regulatora PID dla $T_i = 0.06$.



Rys. 19: Wykres przebiegu funkcji $Q(T_d)$ regulatora PID dla $T_i = 0.06$ (powiększenie).



Rys. 20: Wykres przebiegu funkcji $Q(T_d)$ regulatora PID dla $T_i = 0.1$.



Rys. 21: Wykres przebiegu funkcji $Q(T_d)$ regulatora PID dla $T_i = 0.1$ (powiększenie).

Wartością optymalną parametru T_d jest minimum funkcji $Q(T_d)$, więc z wykresu 16, 17, 18, 19, 20, oraz 21 możemy odczytać, że dla $T_i = 0.02$, $T_d \approx 466.1$, dla $T_i = 0.06$, $T_d \approx 459.8$, oraz dla $T_i = 0.1, T_d \approx 447, 7$.

2.3 Zastosowanie zasad Zieglera-Nicholsa.

Doświadczalnie znalezienie współczynnika wzmocnienia dla którego układ traci stabilność. Ustalenie Okresu oscylacji oraz wzmocnienia krytycznego. Określić wartości parametrów regulatora PID.

a. Doświadczalnie znaleźć współczynnik wzmocnienia, dla którego układ traci stabilność.

$p = 2$

b. Ustalić okres oscylacji i wzmocnienie krytyczne.

$k_p = 2$, $k_u = 6$

c. Według odpowiednich rekomendacji określić wartości parametrów regulatora PID.

$P = k_p * 0.6 = 1, 2$
$I = k_u * 0.5 = 3$
$D = k_u * 0.125 = 0.75$

3 Wnioski

Podczas wykonywania ćwiczenia zauważyłem pewne właściwości parametrów oraz w jaki sposób ich zmiana wpływa na regulator PID. Sterowanie proporcjonalne (k) ma wpływ na czas narastania (zmniejsza ten czas) oraz błąd sterowania (zmniejsza go), ale nie daje dobrego efektu. Natomiast sterowanie całkujące (Ti) ma wpływ na eliminowanie błędu sterowania w stanie ustalonym, jednak pogarsza odpowiedź w czasie narastania i przed czasem ustalonym, co niewątliwie jest wadą. Sterowanie różniczujące (Td) zwiększa stabilność układu poprzez zmniejszenie przeregulowania i poprawe odpowiedzi między stanami. Odpowiedni odbór tych paramatrów jest bardzo ważny dla układu regulacji, gdyż nieodpowiednie ich dobranie może powodować nieprawidłowe działanie lub w szczególych przypadkach destabilizację układu.