

Regulator P i PI w pętli: dlaczego całka usuwa uchyb

Regulator proporcjonalny ma wrodzoną wadę: działa tylko wtedy, gdy jest uchyb, więc pod stałym obciążeniem zawsze trochę się myli. Ta lekcja zamyka pętlę wokół prędkości pionowej drona i pokazuje lekarstwo znane od XIX wieku: pamięć uchybu, czyli całkę. Napišemy regulator jako kilka linijek programu w Pythonie i C++, dobierzemy jego nastawy, korzystając z członu oscylacyjnego z poprzedniej lekcji, i pierwszy raz zderzymy się z ograniczeniem silników.

CZAS

ok. 3 h teorii + ok.
3,5 h zadań i
programowania

ZADANIA

18 zadań w treści
+ 6 do
przemyślenia, z
odpowiedziami

ŹRÓDŁA

K. J. Åström, R.
M. Murray,
Feedback Systems,
rozdz. 1–2; G. C.
Goodwin i in.,
*Control System
Design*, rozdz. 6; J.
C. Maxwell, *On
Governors* (1868)

W TEJ LEKCJI

- 0 Prolog: moderatory i gubernatory
- 1 Regulator P na prędkości drona
- 2 Całka jako pamięć uchybu
- 3 Strojenie PI przez człon oscylacyjny
- 4 Pętla jest programem
- 5 Pierwsze spotkanie z nasyceniem
- 6 Metoda i przykłady rozwiązane
- 7 Błędne intuicje
- 8 Zadania do przemyślenia
- 9 Sprawdź się
- 10 Karta wzorów i co dalej

Prolog: moderatory i gubernatory

Maszyna parowa Watta z 1788 r. miała regulator odśrodkowy: dwie kule na obracającym się wałku. Gdy maszyna przyspieszała, kule rozchodziły się na boki i przez dźwignie przymykały zawór pary. To regulator proporcjonalny: im szybciej, tym mniej pary. Miał znaną wadę. Gdy maszynę obciążono, np. włączając kolejne krosna w tkalni, ustalała się na nieco mniejszej prędkości. Kule musiały być trochę niżej, żeby zawór był trochę bardziej otwarty, a to wymagało niższej prędkości.

W 1868 r. James Clerk Maxwell w pracy *On Governors* rozróżnił dwa rodzaje urządzeń. Te, w których korekta jest proporcjonalna do odchyłki prędkości, nazwał *moderatorami*: pod obciążeniem zostawiają stały błąd. Te, w których korekta jest proporcjonalna do *całki* odchyłki, nazwał właściwymi *gubernatorami*: one doprowadzają prędkość dokładnie do zadanej. Ta lekcja wyjaśnia, dlaczego całka to potrafi.

Pytanie na start. Tempomat z regulatorem P traci pod górę 2 km/h. Jak sprawić, by nie tracił nic, jeśli regulator nie zna ani nachylenia drogi, ani masy samochodu? Odpowiedź w części 2.

1

Regulator P na prędkości drona

W lekcji 1.2 Jak reagują układy: człony podstawowe i odpowiedź skokowa rozłożyliśmy drona na człony: ciąg zmienia prędkość pionową jak człon inercyjny, a prędkość zmienia wysokość jak człon całkujący. Zaczniemy od pętli **prędkości**, bo obiekt jest tu prostszy: tylko jeden człon inercyjny. Prawdziwe drony też mają taką wewnętrzną pętlę prędkości, a pętla wysokości wydaje jej polecenia (to tzw. kaskada, lekcja 3.2 Strojenie: model z testu, reguły nastaw i regulacja kaskadowa).

Obiekt: $m\dot{v} = T - mg - bv$. Ciąg to odpowiedź z modelu plus korekta regulatora: $T = \hat{m}g + \Delta T$. Siłę, której model nie przewidział, oznaczmy $F = (\hat{m} - m)g$ plus ewentualne podmuchy. Wtedy

$$m\dot{v} = \Delta T - bv + F.$$

Regulator P: $\Delta T = K_p(v_{\text{zad}} - v)$. Po podstawieniu i przeniesieniu wyrazów z v na lewą stronę dostajemy człon inercyjny:

WYNIK · PĘTLA PRĘDKOŚCI Z REGULATOREM P

$$m\dot{v} + (b + K_p)v = K_pv_{\text{zad}} + F \quad \Rightarrow \quad \tau_z = \frac{m}{b + K_p}, \quad e_\infty = v_{\text{zad}} - v_\infty = \frac{bv_{\text{zad}} - F}{b + K_p}.$$

Regulator P przyspiesza obiekt (stała czasowa maleje z m/b do $m/(b + K_p)$) i zmniejsza uchyb, ale go nie usuwa.

Liczby: dla $m = 1 \text{ kg}$ i $K_p = 3,5 \text{ N/(m/s)}$ stała czasowa spada z 2 s do 0,25 s. Ale polecenie „wznos się 1 m/s” daje tylko 0,875 m/s. Dron z kamerą ($m = 1,1 \text{ kg}$), któremu każemy wisieć ($v_{\text{zad}} = 0$), opada z prędkością $0,981/4 \approx 0,25 \text{ m/s}$, czyli bez końca traci wysokość. Uchyb prędkości to błąd, który się całkuje.

Skąd uchyb? To ten sam mechanizm co w lekcji 1.1 Po co sprzężenie zwrotne? Pętla otwarta i zamknięta. Żeby utrzymać się w powietrzu, dron potrzebuje dodatkowej siły $(m - \hat{m})g + bv_{\text{zad}}$. Regulator P wytwarza ją tylko z uchybu: $\Delta T = K_p e$. Zerowy uchyb oznaczałby zerową korektę, a wtedy dron by opadał. Uchyb ustala się więc dokładnie na takiej wartości, przy której $K_p e$ (plus ułamek, który „dokłada” opór) pokrywa brakującą siłę.

ZADANIA · CZĘŚĆ 1

3 zadania · ok. 20 min

- 1 Dron ($m = 1 \text{ kg}$, $b = 0,5 \text{ N·s/m}$) ma regulator P o $K_p = 10 \text{ N/(m/s)}$. Jaka jest stała czasowa pętli i uchyb ustalony przy poleceniu 2 m/s (bez błędu masy)?
- 2 Wyprowadź wzór na uchyb z ramki. Wskazówka: w stanie ustalonym $\dot{v} = 0$.
- 3 Dron z kamerą ($m = 1,1 \text{ kg}$) ma wisieć, regulator P prędkości ma $K_p = 3,5 \text{ N/(m/s)}$. O ile metrów obniży się w ciągu minuty? Czy większe K_p rozwiązuje problem?

 podstawowe  średnie  feynmanowskie

2 Całka jako pamięć uchybu

Dodajmy do regulatora drugi człon: sumę wszystkich dotychczasowych uchybów, pomnożoną przez stałą K_i .

PRAWO · REGULATOR PI

$$\Delta T(t) = K_p e(t) + K_i \int_0^t e(s) ds.$$

Jeśli pętla z regulatorem PI ustali się przy stałej wartości zadanej i stałych zakłóceniach, to uchyb ustalony jest zerowy, niezależnie od masy, oporu i zakłócenia.

Dowód jest jednym zdaniem. W stanie ustalonym wszystko jest stałe, także człon całkujący $I(t) = K_i \int_0^t e ds$. Jego pochodna to $K_i e$, więc I może być stałe tylko wtedy, gdy $e = 0$. Jeśli uchyb byłby

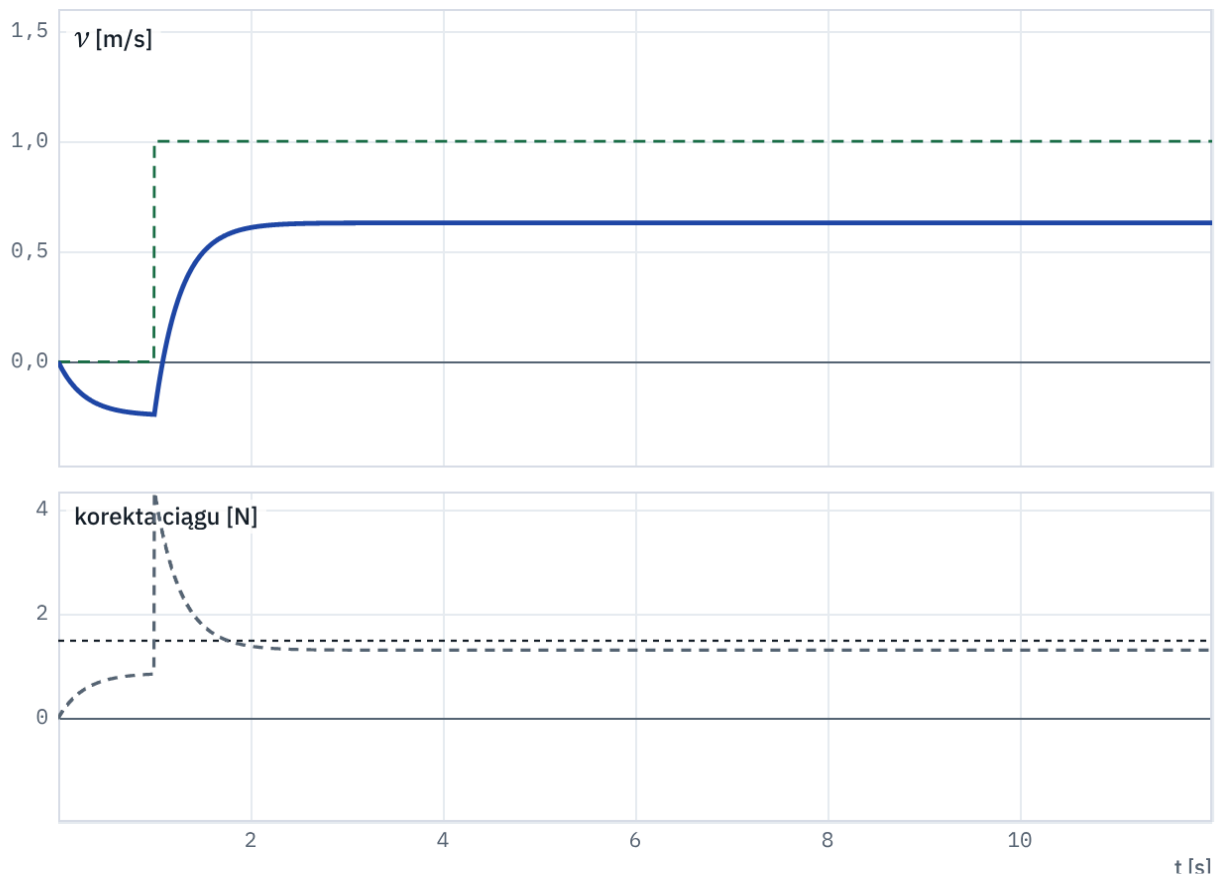
choć trochę różny od zera, całka roślaby (albo malała) liniowo, a razem z nią sterowanie. Nie byłby to więc stan ustalony.

Ważniejsze od dowodu jest to, *gdzie* ląduje całka. Gdy uchyb znika, człon P daje zero, więc całą brakującą siłę $(m - \hat{m})g + bv_{\text{zad}}$ musi dostarczać człon I. Całka zatrzymuje się dokładnie na tej wartości. Regulator PI **sam się uczy**, czego brakuje w modelu, i to „pamięta”. Odpowiedź na pytanie z prologu: tempomat z całką dodaje gazu tak długo, aż prędkość wróci do zadanej. Całka zapamiętuje, ile gazu potrzeba na tym wzniesieniu, nie znając ani nachylenia, ani masy samochodu.

Trzy obrazy pomagają w intuicji. Człon P patrzy na *teraźniejszość*: jak daleko jesteśmy od celu. Człon I patrzy na *przeszłość*: jak długo i jak bardzo już się mylimy. Gdy uchyb jest mały, ale trwa uparcie, P go ignoruje, a I powoli, cierpliwie dokręca sterowanie. W lekcji 3.1 Regulator PID: działanie P, I, D i uchyb ustalony dojdzie trzeci człon, D, który patrzy na *przyszłość*: jak szybko uchyb się zmienia.

SYMULACJA 1

Pętla prędkości pionowej drona. Górny wykres: prędkość. Dolny: dwa składniki korekty ciągu. Człon P (przerywany szary) i człon I (czerwony). Cienka przerywana linia to siła, której brakuje w modelu. Zaczynij od $K_i = 0$: uchyb zostaje. Dodaj K_i i patrz, jak człon I dochodzi do brakującej siły, a człon P gaśnie do zera. Zmieniaj masę i włącz podmuch: cała dopasowuje się do każdej nowej sytuacji.



— prędkość zadana — prędkość $v(t)$ — — człon P: $K_p e$ — — człon I: pamięć uchybu
 — — potrzebna siła $(m - \hat{m})g + b \cdot v_{\text{zad}}$

v po 12 s 0,630 m/s

uchyb ustalony 0,370 m/s

preregulowanie 0,0 %

stała czasowa $m/(b + K_p)$ 0,275 s

Regulator P: uchyb nie znika, bo korekta $K_p e$ istnieje tylko dzięki uchybowi. Potrzebna siła to 1,48 N, więc ustala się uchyb 0,370 m/s. Dodaj K_i .

Stan na rysunku: $K_p = 3,5 \text{ N/(m/s)}$, $K_i = 0,0 \text{ N/m}$, $m = 1,10 \text{ kg}$, skok do 1,0 m/s, ciąg ograniczony.

ZADANIA · CZĘŚĆ 2

4 zadania · ok. 30 min

- 1 Dron o masie 1,1 kg (model: 1 kg) wznosi się z prędkością 1 m/s pod regulatorem PI. Na jakiej wartości ustali się człon I? A człon P?
- 2 W symulacji 1 ustaw $K_p = 3,5$, $K_i = 4$ i włącz podmuch. Co robi człon I w czasie podmuchu i po nim? Dlaczego po podmuchu musi „oddać” to, co nazbierał?
- 3 Czy regulator I bez członu P (sam K_i) usunie uchyb? Sprawdź w symulacji ($K_p = 0$) i wyjaśnij, dlaczego odpowiedź jest gorsza.
- 4 Tempomat z regulatorem PI wjeżdża na wzniesienie, które przy stałym gazie obniżyłoby prędkość o 20 km/h. Model: 3,33 (km/h)/%. O ile procent przepustnicy człon I zwiększy gaz po ustaleniu?

▢▢ podstawowe ▢▢ średnie ▢▢ feynmanowskie

3 Strojenie PI przez człon oscylacyjny

Ile powinny wynosić K_p i K_i ? Z regulatorem PI pętla jest układem drugiego rzędu, a takie układy umiemy już opisywać. Wstawmy $\Delta T = K_p e + K_i \int e$ z $e = v_{\text{zad}} - v$ do równania obiektu i zróżniczkujmy całość, żeby pozbyć się całki:

$$m\ddot{v} + (b + K_p)\dot{v} + K_i v = K_p \dot{v}_{\text{zad}} + K_i v_{\text{zad}} + \dot{F}.$$

Po lewej stronie widać człon oscylacyjny z lekcji 1.2 Jak reagują układy: człony podstawowe i odpowiedź skokowa: $\ddot{v} + 2\zeta\omega_0\dot{v} + \omega_0^2 v$. Porównując współczynniki, odczytujemy

WYNIK · PĘTLA PI Z OBIEKTEM INERCYJNYM

$$\omega_0 = \sqrt{\frac{K_i}{m}}, \quad \zeta = \frac{b + K_p}{2\sqrt{K_i m}}; \quad \text{odwrotnie: } K_i = m\omega_0^2, \quad K_p = 2\zeta\omega_0 m - b.$$

Wybierasz, jak ma się zachowywać pętla (ζ i ω_0), a wzory podają nastawy. K_i ustala szybkość, K_p tłumienie.

Przykład: $\zeta = 1$ i $\omega_0 = 2$ rad/s dla $m = 1$ kg dają $K_i = 4$ N/m i $K_p = 3,5$ N/(m/s). To nastawy domyślne symulacji. Dla drona z kamerą ($m = 1,1$ kg) te same nastawy dają $\zeta \approx 0,95$ i $\omega_0 \approx 1,9$ rad/s: prawie bez zmian. Pętla PI jest odporna na umiarkowany błąd modelu.

Dwie obserwacje z symulacji zasługują na uwagę. Po pierwsze, za duże K_i przy stałym K_p zmniejsza ζ i pętla zaczyna się kołysać: pamięć jest zbyt „nerwowa” i przelatuje. Po drugie, nawet

przy $\zeta = 1$ odpowiedź na skok wartości zadanej ma przeregulowanie ok. 7%. Członu oscylacyjnego z $\zeta = 1$ nie powinno to dotyczyć. Winny jest wyraz $K_p \dot{v}_{\text{zad}}$ po prawej stronie: skok wartości zadanej daje przez człon P natychmiastowe szarpnięcie ciągu. Wyjaśnimy to w lekcji 2.2 Bieguny i zera pętli zamkniętej, linie pierwiastkowe (zera pętli). Znana praktyczna sztuczka: człon P liczy się nie od uchybu, a tylko od pomiaru, $-K_p v$. Wtedy skok wartości zadanej działa tylko przez całkę i przeregulowanie wraca do tego, co obiecuje ζ .

ZADANIA · CZĘŚĆ 3

4 zadania · ok. 30 min

- 1 Dobierz nastawy PI dla drona o masie 1 kg ($b = 0,5 \text{ N}\cdot\text{s/m}$), tak by pętla miała $\zeta = 0,7$ i $\omega_0 = 3 \text{ rad/s}$.

- 2 Przy $K_p = 3,5 \text{ N/(m/s)}$ zwiększasz K_i z 4 do 20 N/m ($m = 1 \text{ kg}$). Jak zmieni się ω_0 i ζ ? Jakie przeregulowanie przewiduje wzór z lekcji 1.2?

- 3 Pokaż, że przy $K_p = 0$ (sam człon I) tłumienie pętli wynosi $\zeta = b/(2\sqrt{K_i m})$. Ile wynosi dla $K_i = 4 \text{ N/m}$, $m = 1 \text{ kg}$? Co to mówi o roli członu P?

- 4 Wymaganie: pętla prędkości ma się ustalać ($\pm 2\%$) w czasie ok. 1 s, bez wyraźnego przelotu. Zaproponuj ζ i ω_0 , a potem nastawy dla $m = 1 \text{ kg}$. Czy silniki to wytrzymają przy skoku polecenia o 2 m/s?


 podstawowe  średnie  feynmanowskie

4 Pętla jest programem

Prawdziwy regulator to nie równanie, tylko pętla w programie mikrokontrolera, która co Δt sekund robi cztery rzeczy: czyta czujnik, liczy uchyb, aktualizuje całkę i wysyła sterowanie do silników. Całka staje się zwykłą sumą: $I \leftarrow I + K_i e \Delta t$. To metoda Eulera z lekcji 4.1 Równania różniczkowe zwyczajne, zastosowana do całki. Poniżej ta sama pętla co w symulacji 1, razem z modelem drona. W prawdziwym dronie model zastępuje fizyka.

Python · dron_pi.py

```
# Pętla prędkości pionowej drona z regulatorem PI (krok Eulera).
g, m_hat, b = 9.81, 1.0, 0.5      # model: masa 1 kg, opór 0,5 N·s/m
m = 1.1                          # prawdziwa masa (kamera +100 g)
Kp, Ki = 3.5, 4.0                 # nastawy regulatora
dt = 0.01                        # okres pętli [s]
T_max = 2 * m_hat * g             # silniki dają najwyżej 2·mg

v, integral = 0.0, 0.0
for k in range(int(12 / dt) + 1):
    t = k * dt
    v_ref = 1.0 if t >= 1.0 else 0.0
    e = v_ref - v                  # 1. pomiar i uchyb
    integral += Ki * e * dt         # 2. pamięć uchybu
    T = m_hat * g + Kp * e + integral # 3. sterowanie: model + P + I
    T = min(max(T, 0.0), T_max)    # 4. ograniczenie silników
    if k % 100 == 0:
        print(f"t = {t:4.1f} s    v = {v:6.3f} m/s    T = {T:6.2f} N    I = {integral:5.3f} N")
    v += dt * (T - m * g - b * v) / m # obiekt: w dronie robi to fizyka
```

C++ · pi.cpp (g++ -O2 -std=c++17 pi.cpp -o pi)

```
#include <algorithm>
#include <cstdio>
// Regulator PI z ograniczeniem sterowania i całkowaniem warunkowym (anti-windup).
struct PI {
    double Kp, Ki, dt, u_min, u_max;
    double integral = 0.0;
    double step(double ref, double meas, double u_ff) {
        double e = ref - meas;
        double i_new = integral + Ki * e * dt;
        double u = u_ff + Kp * e + i_new;
        if (u >= u_min && u <= u_max) integral = i_new;    // całkuj tylko poza nasyceniem
        u = u_ff + Kp * e + integral;
        return std::clamp(u, u_min, u_max);
    }
};

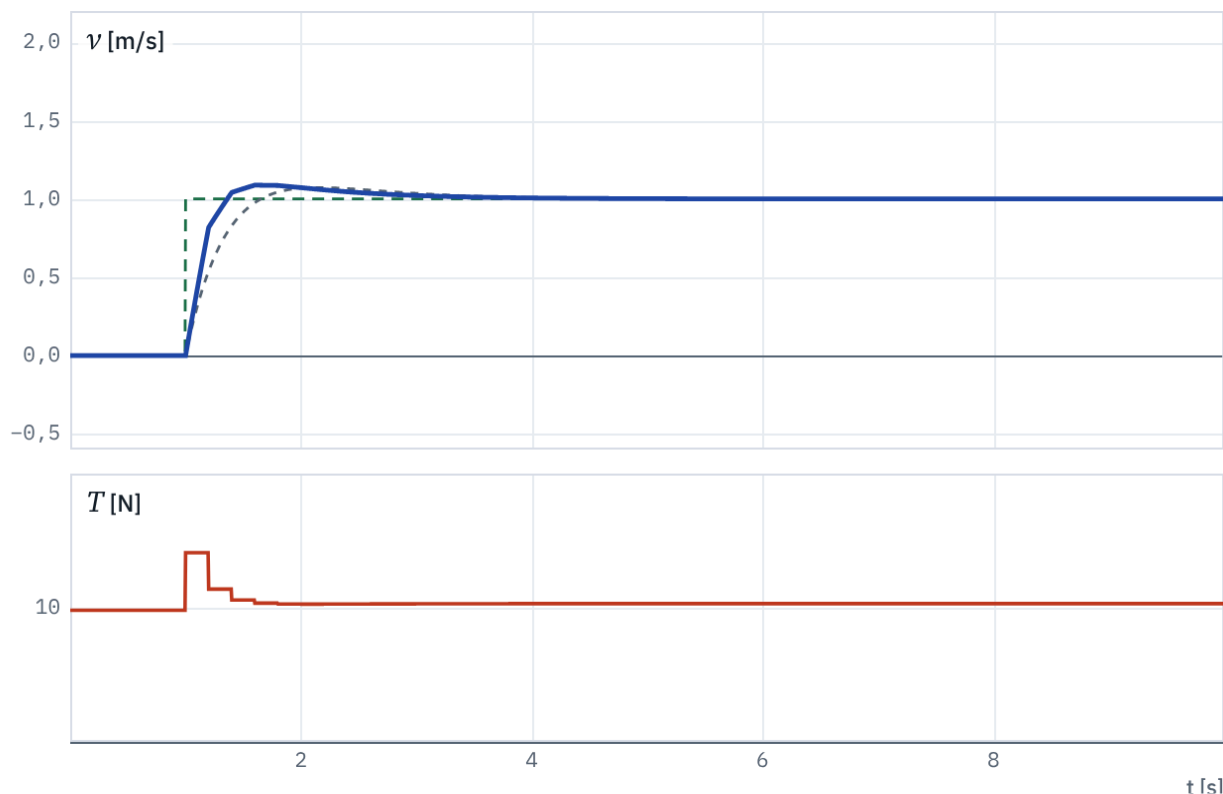
int main() {
    const double g = 9.81, m_hat = 1.0, b = 0.5, m = 1.1, dt = 0.01;
    PI pi{3.5, 4.0, dt, 0.0, 2 * m_hat * g};
    double v = 0.0;
    for (int k = 0; k <= 1200; ++k) {
        double t = k * dt, v_ref = t >= 1.0 ? 1.0 : 0.0;
        double T = pi.step(v_ref, v, m_hat * g);
        if (k % 200 == 0) std::printf("t = %4.1f s    v = %6.3f m/s    T = %6.2f N\n", t, v, T);
        v += dt * (T - m * g - b * v) / m;
    }
}
```

Oba programy wypisują po kilku sekundach $v = 1,000$ m/s i $T = 11,29$ N, czyli dokładnie ciężar drona z kamerą (10,79 N) plus opór przy 1 m/s (0,5 N). Wersja w Pythonie wypisuje też człon całkujący: 1,481 N. Program w C++ ma już gotową strukturę regulatora, którą można przenieść do mikrokontrolera, razem z ochroną przed nawijaniem całki z części 5.

Okres pętli Δt ma znaczenie. Między obiegami pętli sterowanie stoi w miejscu, a regulator nie wie, co się dzieje. Schodki sterowania działają mniej więcej jak opóźnienie o $\Delta t/2$, a opóźnienie w pętli już znamy z przysznica w lekcji 1.1 Po co sprzężenie zwrotne? Pętla otwarta i zamknięta. Praktyczna reguła: pętla powinna obieć się co najmniej kilka, a lepiej ok. dziesięć razy w czasie

narastania odpowiedzi. Szczegóły, w tym transformata Z, w lekcji 3.4 Sterowanie cyfrowe: próbkowanie, transformata Z i regulator w kodzie.

SYMULACJA 2 Ta sama pętla PI ($K_p = 3,5$, $K_i = 4$, $m = 1$ kg) z regulatorem wykonywanym co Δt . Dolny wykres pokazuje, że ciąg zmienia się schodkami. Szara przerywana linia to odpowiedź pętli z $\Delta t = 0,01$ s. Wydłużaj okres pętli: do ok. 0,2 s zmiana jest niewielka, przy 0,3–0,4 s pojawia się wyraźne przeregulowanie, a od ok. 0,46 s pętla traci stabilność.



— — prędkość zadana — prędkość $v(t)$ — ciąg T (stały między obiegami pętli)
— — $\Delta t = 0,01$ s dla porównania

Δt 0,20 s

obiegów pętli na sekundę 5,0

przeregulowanie 8,8 %

czas ustalania ± 2 % 2,08 s

Ciąg zmienia się schodkami i odpowiedź zaczyna się różnić od ciągłej. Schodki działają jak opóźnienie o ok. $\Delta t/2$.

Stan na rysunku: $K_p = 3,5$ N/(m/s), $K_i = 4$ N/m, $m = 1$ kg, $\Delta t = 0,20$ s.

ZADANIA · CZĘŚĆ 4

4 zadania · ok. 45 min (z programowaniem)

- 1 Uruchom `dron_pi.py`. Zmień K_i na 0 i sprawdź, czy wynik zgadza się z wzorem na uchyb regulatora P z części 1.
- 2 W programie sterowanie wyliczone z pomiaru w chwili t_k działa od razu, przez cały następny okres. W prawdziwym mikrokontrolerze obliczenia trwają, i czasem sterowanie trafia do silników dopiero w chwili t_{k+1} . Jak zmienić program, żeby to uwzględnić? Czego się spodziewasz?
- 3 W symulacji 2 znajdź okres pętli, przy którym przeregulowanie przekracza 30%. Ile obiegów pętli przypada wtedy na czas narastania pętli ciągłej (od 10% do 90%: ok. 0,46 s)?
- 4 Dodaj do `dron_pi.py` szum pomiaru prędkości: `e = v_ref - (v + random.gauss(0, 0.05))`. Jak zmieniają się przebiegi v i T ? Który z członów, P czy I, przenosi szum na silniki?

podstawowe średnie feynmanowskie

5 Pierwsze spotkanie z nasyceniem

Silniki drona dają ciąg od zera do pewnego maksimum, tu $2\hat{m}g \approx 19,6$ N. Element wykonawczy, który dostaje polecenie spoza swojego zakresu, robi tyle, ile może: **nasycy się**. Wtedy pętla przestaje być liniowa, a wzory z części 3 przestają obowiązywać.

Regulator PI ma z nasyceniem szczególny kłopot. Gdy silniki dają już maksimum, a uchyb wciąż jest duży, całka dalej rośnie, choć więcej ciągu i tak nie będzie. To **nawijanie całki** (ang. *integrator windup*). Gdy dron w końcu dogoni zadaną prędkość, całka jest „nawinięta” za daleko i przez dłuższą chwilę trzyma ciąg za wysoko. Dron przelatuje, aż całka się „rozwinie”.

Najprostsze lekarstwo widać w programie C++: **nie całkuj, gdy sterowanie jest nasycone** (całkowanie warunkowe, ang. *clamping*). Dla skoku polecenia z 0 do 5 m/s ($m = 1$ kg, $K_p = 3,5$, $K_i = 4$) przeregulowanie maleje wtedy z ok. 12% do ok. 2%, a czas ustalania z ok. 2,8 s do ok. 1 s. Inne metody i sprzężenie w przód, które pomaga omijać nasycenie, poznasz w lekcji 3.3
Nieliniowa rzeczywistość: nasycenie, windup, feedforward i gain scheduling.

ZADANIA · CZĘŚĆ 5

3 zadania · ok. 25 min

- 1 W symulacji 1 ustaw $m = 1$ kg, skok do 5 m/s, $K_p = 3,5$, $K_i = 4$. Porównaj przebieg z ograniczeniem ciągu i bez. Który jest „prawdziwy”?

2 Włącz anti-windup. Porównaj przebieg członu I z wersją bez niego. Dlaczego w wersji z anti-windupem całka rośnie wolniej na początku?

3 Czy regulator P (bez całki) ma problem nawijania? Co mu szkodzi nasycenie?

podstawowe średnie feynmanowskie

☰ METODA: REGULATOR PI DLA OBIEKTU INERCYJNEGO

1. **Model obiektu:** $m\dot{v} + bv = \Delta T + F$ (albo $\tau\dot{y} + y = Ku$; wtedy $m \rightarrow \tau/K$, $b \rightarrow 1/K$).
2. **Wymagania:** przeregulowanie $\rightarrow \zeta$, czas ustalania $\rightarrow \zeta\omega_0$ (lekcja 1.2).
3. **Nastawy:** $K_i = m\omega_0^2$, $K_p = 2\zeta\omega_0 m - b$; jeśli $K_p < 0$, wymagania są wolniejsze niż sam obiekt.
4. **Zakres sterowania:** czy typowy skok wartości zadanej razy K_p mieści się w zapasie elementu wykonawczego? Jeśli nie, dodaj anti-windup i licz się z wolniejszą odpowiedzią.
5. **Okres pętli:** co najmniej kilka, lepiej ok. dziesięciu obiegów na czas narastania.
6. **Sprawdzenie:** symulacja z ograniczeniem, z błędem modelu (np. $\pm 10\%$ masy) i z zakłóceniem.

Przykład 6.1

Tempomat, który pamięta wzniesienie

Tempomat z przykładu 6.1 lekcji 1.1 Po co sprzężenie zwrotne? Pętla otwarta i zamknięta (podpowiedź z modelu 30% gazu przy 100 km/h, model 3,33 (km/h)/%) dostaje człon I. O ile wzrośnie gaz na wzniesieniu, które bez regulacji zabrałoby 20 km/h? A gdy samochód jest obciążony i naprawdę ma 3,0 (km/h)/%?

- 1 W stanie ustalonym prędkość wynosi dokładnie 100 km/h, więc człon P daje 0, a całą korektę daje człon I.
- 2 Samochód zgodny z modelem: potrzeba $20/3,33 = 6,0$ punktów procentowych.
- 3 Obciążony: na płaskim potrzeba $100/3,0 = 33,3\%$, czyli o 3,3 pp więcej niż podpowiedź; na wzniesieniu jeszcze $20/3,0 = 6,7$ pp. Razem człon I: 10,0 pp.

Wynik: człon I ustala się na 6,0 pp (zgodnie z modelem) albo 10,0 pp (obciążony), a prędkość wraca do 100 km/h.

Sprawdzenie: regulator P z lekcji 1.1 zostawiał 1,8 km/h i 3,0 km/h uchybu, a te same przypadki z całką dają zero ✓. Całka „wie” o obciążeniu i nachyleniu, choć nikt jej tego nie powiedział.

Przykład 6.2

Nastawy z wymagań i niespodzianka

Dla drona o masie 1 kg dobierz PI tak, by $\zeta = 0,7$, $\omega_0 = 3 \text{ rad/s}$. Sprawdź symulacją przeregulowanie przy skoku do 1 m/s i przy masie 1,1 kg.

- 1 $K_i = 9 \text{ N/m}$, $K_p = 2 \cdot 0,7 \cdot 3 - 0,5 = 3,7 \text{ N/(m/s)}$.
- 2 Wzór z lekcji 1.2 przewiduje 4,6% przeregulowania. Symulacja pokazuje 16,6% (czas ustalania 1,6 s).
- 3 Przy $m = 1,1 \text{ kg}$: $\zeta \approx 0,67$, symulacja 18,7%.
- 4 Z członem P liczonym tylko od pomiaru ($\Delta T = -K_p v + K_i \int e$) przeregulowanie spada do ok. 4,5%, praktycznie tyle, ile obiecuje człon oscylacyjny (4,6%). Sprawdź to w symulacji 1: zaznacz „człon P liczony tylko od pomiaru”.

Wynik: $K_p = 3,7 \text{ N/(m/s)}$, $K_i = 9 \text{ N/m}$. Nadmiarowe przeregulowanie pochodzi od członu P działającego na skok wartości zadanej.

Sprawdzenie: na zakłócenie (np. podmuch) obie wersje reagują identycznie, bo zakłócenie nie przechodzi przez wartość zadaną ✓. Przy skoku polecenia z kolei P liczone od pomiaru daje łagodniejsze sterowanie i mniejsze ryzyko nasycenia.

Przykład 6.3

PI prędkości, a wysokość i tak ucieka

Dron z kamerą ($m = 1,1 \text{ kg}$) ma wisieć: $v_{\text{zad}} = 0$, regulator PI prędkości $K_p = 3,5$, $K_i = 4$. Prędkość wraca do zera. Czy wraca też wysokość?

- 1 Wysokość to całka z prędkości: $z = \int v \, dt$. Przy $v_{\text{zad}} = 0$ uchyb to $-v$, więc człon I to $K_i \int (-v) \, dt = -K_i z$.
- 2 Równanie drona: $m\dot{v} = -K_p v - K_i z - bv + F$, czyli $m\ddot{z} + (b + K_p)\dot{z} + K_i z = F$. Regulator PI prędkości to regulator PD wysokości.
- 3 W stanie ustalonym $K_i z_\infty = F$: $z_\infty = -0,981/4 \approx -0,25 \text{ m}$.

Wynik: dron przestaje opadać, ale zostaje ok. 25 cm niżej. Prędkość jest regulowana bez uchybu, wysokość z uchybem.

Sprawdzenie: całkując numerycznie prędkość z symulacji, dostajemy $-0,245 \text{ m}$ ✓. Żeby usunąć uchyb wysokości, potrzeba całki uchybu *wysokości*, czyli pętli wysokości nad pętlą prędkości (kaskada, lekcja 3.2 Strojenie: model z testu, reguły nastaw i regulacja kaskadowa). Całka usuwa uchyb tej wielkości, którą całkuje.

Przykład 6.4

Jak często liczyć

Pętla PI z przykładu 6.2 ($\omega_0 = 3 \text{ rad/s}$) ma działać na mikrokontrolerze, który może liczyć regulator 100 razy na sekundę, ale woli rzadziej (oszczędza baterię). Jaki okres pętli wybrać?

- 1 Czas narastania (10–90%) tej pętli z członem P od uchybu to ok. 0,33 s (symulacja; czysty człon oscylacyjny dałby $2,1/\omega_0 \approx 0,7 \text{ s}$, ale szarpnięcie z członu P przyspiesza start).
- 2 Ok. dziesięć obiegów na ten czas: $\Delta t \approx 0,03 \text{ s}$ (ok. 30 Hz).
- 3 Dla zapasu i z myślą o szumie (filtr potrzebuje próbek) wybieramy 50–100 Hz.

Wynik: $\Delta t = 0,01\text{--}0,02 \text{ s}$; pętla prędkości drona i tak zwykle jest wolniejsza od wewnętrznych pętli orientacji, które działają z częstotliwością setek herców.

Sprawdzenie: w symulacji 2 (pętla z $\omega_0 = 2 \text{ rad/s}$, czas narastania 0,46 s) pogorszenie staje się wyraźne od ok. 0,2 s, czyli przy ok. dwóch obiegach na czas narastania. Wybrane 0,01–0,02 s daje ok. 15–30 obiegów, z dużym zapasem ✓.

Błędne intuicje, których trzeba się oduczyć

× „Uchyb regulatora P to вина złego strojenia.”

To cecha samej struktury: korekta $K_p e$ wymaga uchybu. Większe K_p go zmniejsza, usuwa dopiero całka.

× „Całka usuwa każdy uchyb.”

Usuwa uchyb przy stałej wartości zadanej i stałych zakłóceniach, i tylko w tej wielkości, którą całkuje. Przy wartości zadanej rosnącej liniowo zostaje stały uchyb (zadanie 8.2), a PI prędkości zostawia uchyb wysokości (przykład 6.3).

× „Większe K_i to szybsze i lepsze usuwanie uchybu.”

Szybsze, ale przy stałym K_p tłumienie maleje jak $1/\sqrt{K_i}$ i pętla zaczyna się kołysać.

× „Regulator w programie działa tak samo jak na papierze.”

Tylko jeśli pętla obiega się dużo szybciej niż dynamika obiektu. Rzadkie próbkowanie działa jak opóźnienie.

× „Nasycenie to problem silników, nie regulatora.”

Nawijanie całki to błąd algorytmu: regulator gromadzi „dług”, którego element wykonawczy nie może spłacić. Naprawia się go w kodzie (anti-windup).

× „ $\zeta = 1$ gwarantuje brak przeregulowania.”

Tylko dla czystego członu oscylacyjnego. Człon P działający na skok wartości zadanej dodaje szarpnięcie i przelot (przykład 6.2).

Zadania do przemyslenia

Te zadania wymagają więcej myślenia niż liczenia.

ZADANIA DO PRZEMYŚLENIA

6 zadań · ok. 2 godziny

1

Całka i błąd czujnika. Czujnik prędkości ma stały błąd +0,1 m/s. Co zrobi regulator PI?



Czy całka „zauważy” ten błąd?

- 2 **Polecenie rosnące liniowo.** Dron ma przyspieszać: $v_{\text{zad}}(t) = at$, $a = 0,5 \text{ m/s}^2$ ($m = 1 \text{ kg}$, $K_p = 3,5$, $K_i = 4$). Pokaż, że uchyb ustali się na stałej wartości różnej od zera, i ją oblicz.
-
- 3 **PI na wysokości.** Ktoś proponuje pominąć pętlę prędkości i dać regulator PI bezpośrednio na wysokość: $\Delta T = K_p(z_{\text{zad}} - z) + K_i \int (z_{\text{zad}} - z) dt$. Równanie pętli ma postać $m\ddot{z} + b\dot{z} + K_p\dot{z} + K_iz = \dots$. Wiadomo (lekcja 2.2 Bieguny i zera pętli zamkniętej, linie pierwiastkowe), że taki układ jest stabilny tylko gdy $bK_p > mK_i$. Co z tego wynika dla drona o $m = 1 \text{ kg}$, $b = 0,5 \text{ N}\cdot\text{s/m}$?
-
- 4 **Start z ziemi.** Dron stoi na ziemi z włączonym regulatorem PI wysokości, a wartość zadana to 1 m. Silniki są jeszcze zablokowane przez 5 s (procedura bezpieczeństwa). Co się stanie po ich odblokowaniu? Jak to naprawić?
-
- 5 **Watt pod obciążeniem.** Regulator odśrodkowy Watta jest proporcjonalny. Maszyna bez obciążenia pracuje z prędkością 100 obr./min; pełne obciążenie przy tym samym otwarciu zaworu obniżyłoby ją do 70 obr./min. Wzmocnienie pętli wynosi $L = 9$. Jaka będzie prędkość pod pełnym obciążeniem? Co by dał gubernator Maxwella?
-
- 6 **Za wolną pętlą.** Regulator P prędkości drona ($m = 1 \text{ kg}$, $b = 0,5 \text{ N}\cdot\text{s/m}$, $K_p = 3,5 \text{ N}/(\text{m/s})$) liczony jest co Δt , a obiekt między obiegami to człon inercyjny. Można pokazać, że po jednym obiegu odchyłka mnoży się przez $a - (1 - a)K_p/b$, gdzie $a = e^{-\Delta t b/m}$. Dla jakiego Δt pętla traci stabilność?

▢▢▢ podstawowe ▢▢▢ średnie ▢▢▢ feynmanowskie

9 Sprawdź się

Odpowiedz w myślach, zanim rozwiniesz odpowiedź.

- 1 **Dlaczego regulator P zostawia uchyb przy stałym obciążeniu?**

Bo korekta $K_p e$ istnieje tylko przy niezerowym uchybie, a stałe obciążenie wymaga stałej korekty.

- 2 **Dlaczego z całką uchyb ustalony jest zerowy?**

W stanie ustalonym całka jest stała, a jej pochodna to $K_i e$, więc $e = 0$.

3 Na jakiej wartości ustala się człon I?

Na sile (sterowaniu), której brakuje w modelu: całka „uczy się” zakłócenia i błędu modelu.

4 Jak K_p i K_i przekładają się na ω_0 i ζ pętli z obiektem inercyjnym?

$$\omega_0 = \sqrt{K_i/m}, \zeta = (b + K_p)/(2\sqrt{K_i m}).$$

5 Co to jest nawijanie całki i jak mu zapobiec?

Całka rośnie, gdy element wykonawczy jest nasycony; potem trzyma sterowanie za długo i powoduje przelot. Zapobiega mu całkowanie warunkowe: nie całkuj w nasyceniu.

6 Czemu rzadkie obiegi pętli szkodzą?

Sterowanie stoi między obiegami, co działa jak opóźnienie ok. $\Delta t/2$.

7 Czy PI prędkości drona usuwa uchyb wysokości?

Nie: to w istocie PD wysokości; uchyb wysokości F/K_i zostaje.

8 Kim byli „moderatory” i „gubernatory” Maxwella?

Regulatory z korektą proporcjonalną do odchyłki (zostawiają błąd) i do całki odchyłki (doprowadzają do zadanej).

REGULATOR P

obiekt: $m\dot{v} + bv = \Delta T + F$

P: $\Delta T = K_p e$; $\tau_z = \frac{m}{b + K_p}$

$$e_\infty = \frac{bv_{\text{zad}} - F}{b + K_p}$$

REGULATOR PI

$$\Delta T = K_p e + K_i \int_0^t e \, ds; \quad e_\infty = 0$$

$$I_\infty = bv_{\text{zad}} - F \text{ (brakująca siła)}$$

STROJENIE I IMPLEMENTACJA

$$\omega_0 = \sqrt{K_i/m}, \quad \zeta = \frac{b + K_p}{2\sqrt{K_i m}}$$

$$K_i = m\omega_0^2, \quad K_p = 2\zeta\omega_0 m - b$$

w kodzie: $I \leftarrow I + K_i e \Delta t$

anti-windup: nie całkuj w nasyceniu

rampa: $e_\infty = ba/K_i$; PI prędkości: $z_\infty = F/K_i$

Do przeczytania

- **Obowiązkowo:** K. J. Åström, R. M. Murray, *Feedback Systems*, rozdz. 1 (sprzężenie, regulator PI i rola całki na przykładach).
- **Dla chętnych:** G. C. Goodwin i in., *Control System Design*, rozdz. 6 (klasyczny regulator PID); J. C. Maxwell, *On Governors*, Proceedings of the Royal Society 16 (1868), krótki i wciąż czytelny.

To koniec części I. Umiemy już zamknąć pętlę, rozpoznać obiekt po odpowiedzi skokowej i dobrać regulator PI tak, by pętla zachowywała się jak człon oscylacyjny o wybranych ζ i ω_0 . Dla obiektu z jedną inercją to wystarcza. Dla obiektów złożonych (dron z silnikami, czujnikiem i opóźnieniem) liczenie równań w czasie staje się uciążliwe. Część II wprowadza narzędzie, które zamienia całki i pochodne na mnożenie: transmitancję i schematy blokowe.



Odpowiedzi

Zadania · część 1

1. $\tau_z = 1/10,5 \approx 0,095$ s; $e_\infty = 0,5 \cdot 2/10,5 \approx 0,095$ m/s.
2. $(b + K_p)v_\infty = K_p v_{\text{zad}} + F$; $e_\infty = v_{\text{zad}} - v_\infty = \frac{(b+K_p)v_{\text{zad}} - K_p v_{\text{zad}} - F}{b+K_p} = \frac{bv_{\text{zad}} - F}{b+K_p}$.
3. $v_\infty = F/(b + K_p) = -0,981/4 \approx -0,25$ m/s, czyli ok. 15 m na minutę. Większe K_p zmniejsza prędkość opadania, ale dron wciąż opada. W obiekcie całującym każdy stały uchyb prędkości to rosnący błąd wysokości.

Zadania · część 2

1. $I_\infty = (1,1 - 1) \cdot 9,81 + 0,5 \cdot 1 \approx 1,48$ N; człon P: 0, bo $e_\infty = 0$. Ciąg: $9,81 + 1,48 = 11,29$ N.
2. W czasie podmuchu uchyb jest dodatni, więc całka rośnie i dokłada ciągu, aż zrównoważy dodatkowe 3 N. Po podmuchu ten nadmiar ciągu wypycha drona za szybko w górę: uchyb jest ujemny, a całka maleje z powrotem do 1,48 N. Pamięć działa w obie strony. Dlatego po zakłóceniu z całką zwykle widać mały przelot w drugą stronę.
3. Usunie: dowód z ramki nie potrzebuje członu P. Ale sama całka reaguje dopiero, gdy uchyb trwa: na początku sterowanie jest zerowe i rośnie powoli. Przy $K_p = 0$ tłumienie pętli to tylko opór b , więc pojawia się wyraźne kołysanie ($\zeta = b/(2\sqrt{K_i m})$, część 3).
4. Żeby oddać 20 km/h, potrzeba $20/3,33 = 6,0$ punktów procentowych. Tyle wyniesie człon I (przykład 6.1).

Zadania · część 3

1. $K_i = 9$ N/m, $K_p = 2 \cdot 0,7 \cdot 3 \cdot 1 - 0,5 = 3,7$ N/(m/s).
2. ω_0 : z 2 na 4,47 rad/s; ζ : z 1 na $4/(2\sqrt{20}) \approx 0,45$. Wzór daje ok. 21%; symulacja pokazuje ok. 30%, bo dochodzi szarpnięcie od $K_p \dot{v}_{\text{zad}}$.
3. Podstawiamy $K_p = 0$: $\zeta = 0,5/(2 \cdot 2) = 0,125$, silne kołysanie. Człon P działa jak sztuczny opór: dodaje się do b i to on daje pętli PI tłumienie.
4. Np. $\zeta = 0,8$, $t_s \approx 4/(\zeta\omega_0) = 1$ s, czyli $\omega_0 = 5$ rad/s: $K_i = 25$ N/m, $K_p = 7,5$ N/(m/s). Uwaga: z członem P liczącym od uchybu skok wartości zadanej da ok. 15% przelotu (szarpnięcie z części 3). Skok o 2 m/s daje od razu $K_p \cdot 2 = 15$ N korekty, a zapas ciągu to 9,81 N, więc silniki się nasycą (część 5). Oba problemy usuwa człon P liczony od pomiaru: przelot spada do ok. 1,5%, a największy ciąg do ok. 14,4 N, poniżej granicy.

Zadania · część 4

1. Program pokazuje $v \rightarrow 0,63$ m/s. Ze wzoru: $e_\infty = (bv_{\text{zad}} - F)/(b + K_p) = (0,5 + 0,981)/4 \approx 0,37$ m/s, czyli $v_\infty \approx 0,63$ m/s ✓.
2. Trzeba zapamiętać sterowanie i wysyłać do obiektu to z poprzedniego obiegu (np. $T_{\text{out}}, T_{\text{next}} = T_{\text{next}}, T$). To dodatkowe opóźnienie o jeden okres Δt , które dokłada się do opóźnienia ok. $\Delta t/2$ od schodków. Przy małym Δt zmiana jest niewidoczna, przy dużym pogarsza zapas stabilności, jak

dłuższa rura w prysznicu. Sama zamiana kolejności dwóch instrukcji w pętli niczego takiego nie robi: zmienia tylko pierwszy obieg.

3. Ok. $\Delta t = 0,3$ s (przeregulowanie ok. 31%), czyli tylko ok. półtora obiegu na czas narastania. Przy $\Delta t = 0,1$ s (prawie pięć obiegów) przeregulowanie jest prawie takie jak w pętli ciągłej (7,7% zamiast 7,3%), a przy 0,2 s (ok. dwa obiegi) już wyraźnie większe (8,8%).
4. Prędkość prawie się nie zmienia (dron uśrednia szybkie szarpnięcia swoją bezwładnością), ale ciąg drga o ok. $K_p \cdot 0,05 \approx 0,18$ N przy każdym obiegu. Szarpie człon P. Człon I sumuje szum z wagą $K_i \Delta t = 0,04$, więc go uśrednia. Problem szumu w członie P (a jeszcze bardziej w D) wróci w lekcjach 2.5 Zakłócenia, szum i wrażliwość: S, T i kształtowanie pętli i 3.1 Regulator PID: działanie P, I, D i uchyb ustalony.

Zadania · część 5

1. Bez ograniczenia regulator żąda na starcie ok. 27,5 N, czyli więcej, niż silniki mogą dać. Przebieg jest ładny (przeregulowanie ok. 7%), ale nierealny. Z ograniczeniem przeregulowanie rośnie do ok. 12%: to prawdziwy dron.
2. Bo w czasie nasycenia całka stoi w miejscu. Rośnie dopiero, gdy ciąg zejdzie poniżej maksimum. Nie gromadzi więc „długu”, który potem trzeba spłacać przelotem.
3. Nie ma czego nawijać: sterowanie zależy tylko od bieżącego uchybu, więc gdy wyjdzie z nasycenia, od razu wraca do działania liniowego. Nasycenie spowalnia go (dron przyspiesza najszybciej, jak może, a nie tak, jak przewiduje τ_z), ale nie wywołuje dodatkowego przelotu.

Zadania do przemyślenia

1. 1. Regulator doprowadzi do zera uchyb *zmierzony*: $v_{\text{zad}} - (v + 0,1) = 0$, więc prawdziwa prędkość będzie o 0,1 m/s za mała.
2. Całka nie ma jak zauważyć błędu czujnika: widzi tylko pomiar. To ta sama lekcja co w 1.1. Pętla, nawet z całką, jest tak dokładna jak jej czujnik.
2. 1. Po ustaleniu się przebiegów prędkość rośnie jak v_{zad} , a potrzebna siła $ma + bv$ rośnie liniowo w tempie ba . Rosnącą siłę musi dawać człon I, a ten rośnie w tempie $K_i e$.
2. $K_i e_\infty = ba$, więc $e_\infty = ba/K_i = 0,5 \cdot 0,5/4 = 0,0625$ m/s. Symulacja daje to samo. Do śledzenia rampy bez uchybu potrzeba dwóch całkowań w pętli (lekcja 3.1 Regulator PID: działanie P, I, D i uchyb ustalony, typ układu).
3. 1. Warunek: $K_i < K_p b/m = K_p/2$. Przy $K_p = 4$ N/m całka musi być mniejsza niż 2 N/(m·s).
2. Tłumienie pochodzi tu tylko z oporu b , którego jest mało, a całka dodaje opóźnienia. Sam regulator P na wysokości kołysał się już w lekcji 1.1, a całka to pogarsza. Potrzebny jest pomiar prędkości: człon D albo wewnętrzna pętla prędkości (kaskada).
4. 1. Przez 5 s uchyb wynosi 1 m, a całka rośnie: $5 \text{ s} \cdot 1 \text{ m} \cdot K_i$. Po odblokowaniu regulator każe dać dużo za dużo ciągu i dron wystrzeli wysoko ponad 1 m. To klasyczny windup, tylko z innego powodu niż nasycenie.
2. Naprawa: zerować całkę albo jej nie liczyć, dopóki regulator nie ma kontroli nad obiektem. Autopiloty robią to przy przełączaniu trybów („bezuderzeniowe przełączanie”, lekcja 3.3 Nieliniowa rzeczywistość: nasycenie, windup, feedforward i gain scheduling).
5. 1. Zakłócenie -30 obr./min dzielone przez $1 + L = 10$: spadek o 3 obr./min, czyli 97 obr./min.
2. Z całką prędkość wróciłaby do 100 obr./min. Kosztem byłaby skłonność do kołysania (tzw. hunting), którą Maxwell analizował właśnie w swojej pracy.

6. 1. Niestabilność, gdy $|a - 7(1 - a)| > 1$. Dla ujemnego mnożnika: $7(1 - a) - a > 1$, czyli $a < 0,75$.
2. $e^{-\Delta t/2} < 0,75 \Rightarrow \Delta t > 2 \ln(4/3) \approx 0,58$ s. Odchyłka zmienia wtedy znak przy każdym obiegu i rośnie, jak u nerwowego regulatora pod prysznicem. Z całą granicą przesuwamy się do jeszcze krótszych okresów (symulacja 2).
-

Automatyka · Lekcja 1.3: Regulator P i PI w pętli · Źródła: K. J. Åström, R. M. Murray, *Feedback Systems*, rozdz. 1–2; G. C. Goodwin, S. F. Graebe, M. E. Salgado, *Control System Design* (Prentice Hall 2001), rozdz. 6; J. C. Maxwell, *On Governors* (1868). Programy z części 4 sprawdzone (Python 3, g++ 13). Wszystkie liczby policzone na nowo (tools/verify/c1_3.py).