

Sterowanie cyfrowe: próbkowanie, transformata Z i regulator w kodzie

Każdy regulator z tego kursu działa dziś w mikrokontrolerze, który co kilka milisekund czyta czujnik, liczy kilka linijek i wysyła nowe sterowanie. Ta lekcja pokazuje, co to zmienia w pętli. Próbkowanie działa jak opóźnienie o pół okresu. Za wolny czujnik może zobaczyć drgania, których nie ma. Pętla stabilna w ciągłym świecie może się rozkołysać w dyskretnym. Na koniec piszemy PID, który można wgrać do drona, w Pythonie i w C++.

CZAS

ok. 2,5 h teorii + ok. 3 h zadań

ZADANIA

15 zadań w treści + 5 do przemyślenia, z odpowiedziami

ŹRÓDŁA

G. C. Goodwin i in., *Control System Design*, rozdz. 12–13; J. Kowal, *Podstawy automatyki*, rozdz. 11; K. J. Åström, B. Wittenmark, *Computer-Controlled Systems*, rozdz. 1–2 i 8

W TEJ LEKCJI

- 0 Prolog: samolot sterowany komputerem z Apollo
- 1 Próbkowanie i ekstrapolator zerowego rzędu
- 2 Aliasing
- 3 Transformata Z i płaszczyzna z
- 4 Dyskretyzacja regulatora
- 5 PID w kodzie
- 6 Metoda i przykłady rozwiązań
- 7 Błędne intuicje
- 8 Zadania do przemyślenia
- 9 Sprawdź się
- 10 Karta wzorów i co dalej

Prolog: samolot sterowany komputerem z Apollo

25 maja 1972 r. w ośrodku badań w locie NASA w Edwards (dziś Armstrong) wystartował zmodyfikowany myśliwiec F-8 Crusader. Nie miał żadnego mechanicznego połączenia drążka ze sterami. Ruchy pilota odczytywał komputer, liczył sterowanie i wysyłał je do siłowników. Tym komputerem był egzemplarz komputera nawigacyjnego statku Apollo. Program F-8 Digital Fly-By-Wire pokazał, że samolotem może sterować maszyna, która widzi świat w dyskretnych chwilach i działa w pętli „czytaj – liczy – wyślij”.

Dziś to standard: od Airbusa po każdy dron za 200 zł. Konstruktorzy F-8 musieli jednak odpowiedzieć na pytania, które zadamy w tej lekcji. Jak często trzeba liczyć? Co się dzieje między chwilami próbkowania? Czy wzór na PID z kartki można po prostu przepisać do programu?

Pytanie na start. Regulator P prędkości drona jest stabilny przy każdym K_p (lekcja 1.3). Czy jego cyfrowa wersja, liczona co 0,1 s, też? Odpowiedź w części 3.

Próbkowanie i ekstrapolator zerowego rzędu

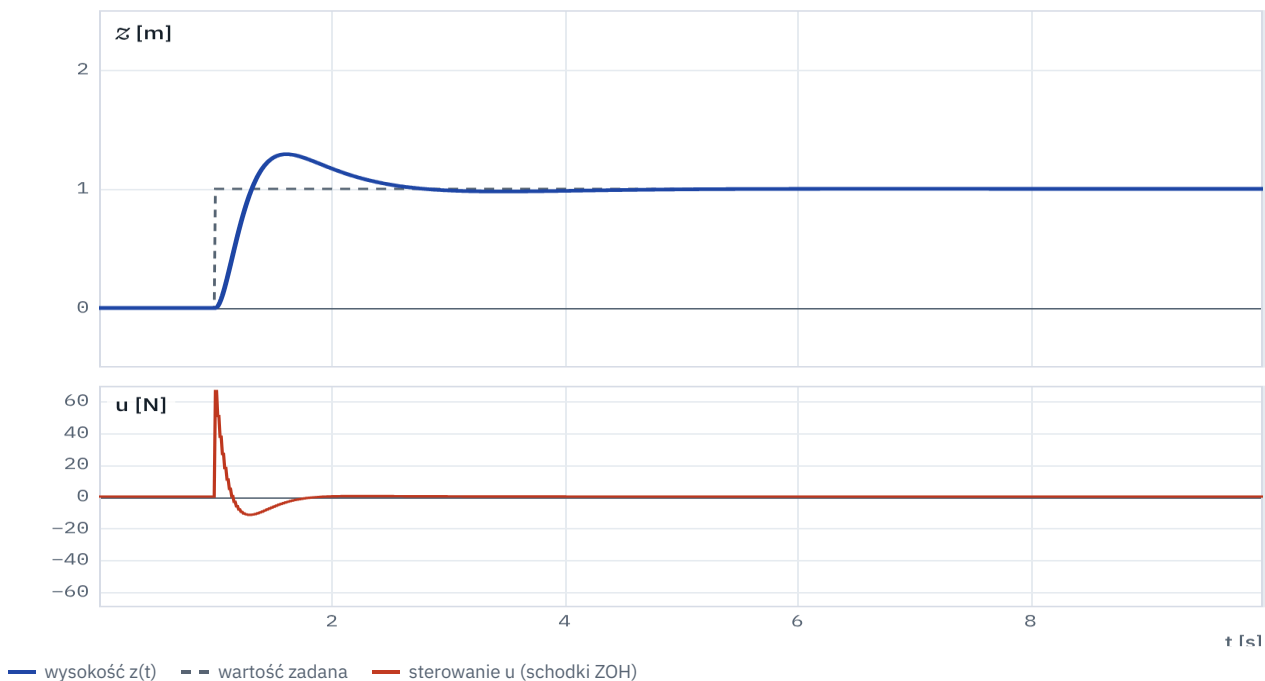
Regulator cyfrowy co T sekund (okres próbkowania, $f_s = 1/T$) czyta pomiar $y_k = y(kT)$, liczy u_k i wysyła je do przetwornika, który trzyma tę wartość stałą aż do następnego taktu. To trzymanie nazywa się **ekstrapolatorem zerowego rzędu** (ang. *zero-order hold*, ZOH). Sterowanie ma postać schodków.

Schodki są średnio spóźnione o pół okresu względem gładkiej krzywej, którą przybliżają. Przy częstotliwościach dużo niższych od f_s ZOH działa więc jak **opóźnienie o $T/2$** , a z lekcji 2.4 Stabilność pętli: punkt -1 , zapasy i opóźnienie wiemy, ile kosztuje opóźnienie:

$$\Delta\varphi \approx -\omega_c \frac{T}{2} \quad (\text{w radianach}).$$

Do tego dochodzi czas obliczeń i pomiaru (często znów rzędu T). Reguła praktyczna: **20–40 próbek na okres częstotliwości przejścia**, czyli $\omega_c T \approx 0,15\text{--}0,3$, co kosztuje 4–9° zapasu fazy. (Åström–Wittenmark)

SYMULACJA 1 PID wysokości drona z lekcji 3.1 (z filtrem D) liczony co T i trzymany przez ZOH. Wydłużaj T od 5 ms do 450 ms. Przy jakim T przeregulowanie zaczyna wyraźnie rosnąć, a przy jakim pętla traci stabilność?



opóźnienie ZOH $\approx T/2$ 10 ms zapas fazy $\approx 55,9^\circ - \omega_c T/2$ 53,2° przeregulowanie 29,1 % próbek na okres ω_c 66

Szybkie próbkowanie: regulator cyfrowy zachowuje się prawie jak ciągły. Reguła praktyczna: 20–40 próbek na okres częstotliwości przejścia.

Stan na rysunku: PID 9,6 / 8 / 4,3 z filtrem D ($N = 8$), dyskretyzacja metodą trapezów, $T = 20$ ms; dron $1/(s(s + 0,5))$, skok 1 m, bez nasycenia.

ZADANIA · CZĘŚĆ 1

3 zadania · ok. 20 min

- 1 PID wysokości z filtrem D ma $\omega_c = 4,76$ rad/s i zapas fazy $55,9^\circ$. Ile zapasu zostanie przy $T = 20$ ms, 100 ms, 200 ms?
- 2 Pętla prędkości kątowej drona ma $\omega_c \approx 50$ rad/s. Jak często trzeba ją liczyć według reguły 20–40 próbek na okres?
- 3 Dlaczego ZOH przypomina opóźnienie o $T/2$, a nie o T ?

podstawowe średnie feynmanowskie

2 Aliasing

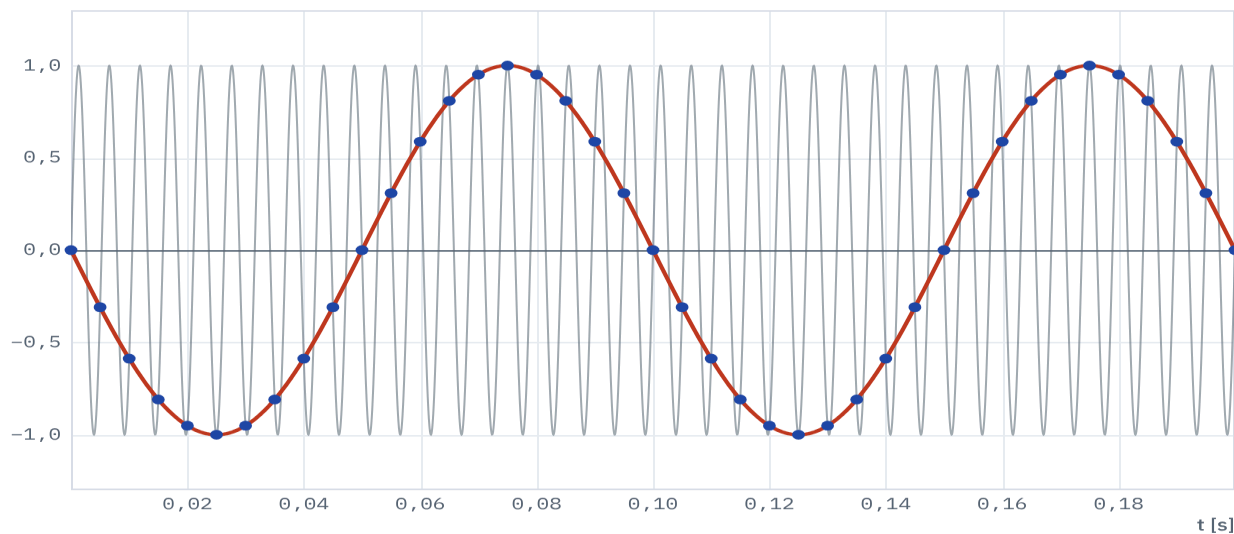
Próbkowanie gubi informacje: sinusoidy o częstotliwościach f i $f \pm n f_s$ dają dokładnie te same próbki. Sygnał szybszy niż częstotliwość Nyquista $f_s/2$ udaje wolniejszy. To **aliasing** (po polsku też „nakładanie się widm”), znany z filmów, na których koła samochodu kręcą się do tyłu.

$$f_{\text{widziana}} = |f - n f_s|, \quad n = \text{najbliższa liczba całkowita do } f/f_s.$$

Twierdzenie o próbkowaniu (Nyquist, Shannon) mówi, że sygnał bez składowych powyżej $f_s/2$ da się z próbek odtworzyć dokładnie. Wniosek dla regulatora: przed próbkowaniem trzeba usunąć wszystko powyżej $f_s/2$ **filtrem antyaliasingowym** (analogowym albo w czujniku). Po próbkowaniu jest za późno: alias wygląda dokładnie jak prawdziwy wolny sygnał.

SYMULACJA 2

Drgania ramy drona od śmigła (190 Hz) widziane przez czujnik próbkowany 200 Hz. Próbkki układają się w wolną sinusoidę 10 Hz, którą regulator wzięby za prawdziwy ruch. Zwiększ f_s powyżej 380 Hz.



— prawdziwy sygnał — próbki — to, co „widzi” regulator

częstotliwość Nyquista $f_s/2$ 100 Hz

częstotliwość widziana 10 Hz

próbkę na okres sygnału 1,05

Sygnał szybszy niż $f_s/2$: próbki układają się w sinusoidę o zupełnie innej, niskiej częstotliwości. Regulator potraktuje ją jak prawdziwy ruch. Żaden filtr cyfrowy po próbkowaniu tego nie odróżni.

Stan na rysunku: $f = 190$ Hz, $f_s = 200$ Hz, okno 0,2 s.

ZADANIA · CZĘŚĆ 2

3 zadania · ok. 15 min

- 1 Jaką częstotliwość „zobaczy” czujnik próbkowany 200 Hz, gdy prawdziwe drganie ma 210 Hz? A 50 Hz próbkowane przy 60 Hz?
- 2 Kamera filmuje 24 klatki na sekundę koło z 6 szprychami. Przy jakiej prędkości obrotowej koło wydaje się stać?
- 3 Dlaczego filtr cyfrowy po próbkowaniu nie usunie aliasu?

□ □ □ podstawowe ▨ ▨ ▨ średnie ▩ ▩ ▩ feynmanowskie

3 Transformata Z i płaszczyzna z

Dla układów ciągłych narzędziem była transformata Laplace'a. Dla ciągów liczb $x_0, x_1, x_2, \dots$ jej odpowiednikiem jest transformata Z:

$$X(z) = \sum_{k=0}^{\infty} x_k z^{-k}.$$

Najważniejszy przykład to ciąg geometryczny $x_k = a^k$. Z sumy szeregu geometrycznego (lekcja 1.6 Szeregi liczbowe i potęgowe): $X(z) = \sum (az^{-1})^k = \frac{1}{1-az^{-1}} = \frac{z}{z-a}$ dla $|z| > |a|$. Druga reguła: opóźnienie o jedną próbkę to mnożenie przez z^{-1} . Dzięki niej równania różnicowe zamieniają się w algebrę, jak równania różniczkowe przy Laplasie.

Ciągły biegun p daje e^{pt} , a w chwilach próbkowania $e^{pkT} = (e^{pT})^k$, czyli ciąg geometryczny o ilorazie $z = e^{pT}$. Stąd odwzorowanie

$$z = e^{sT} :$$

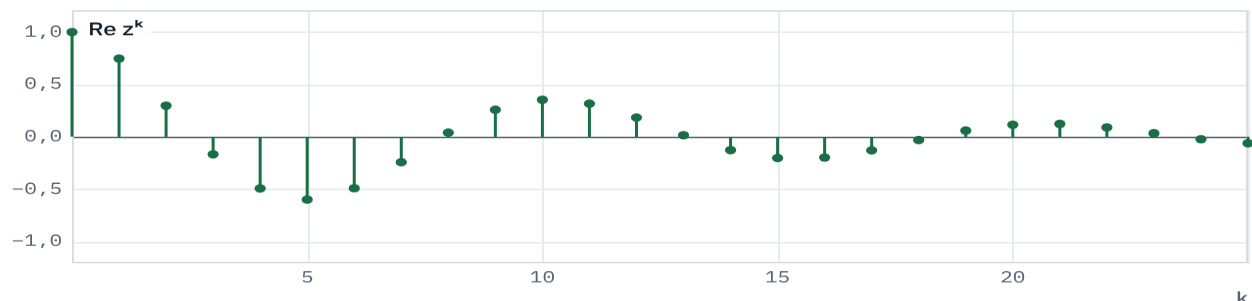
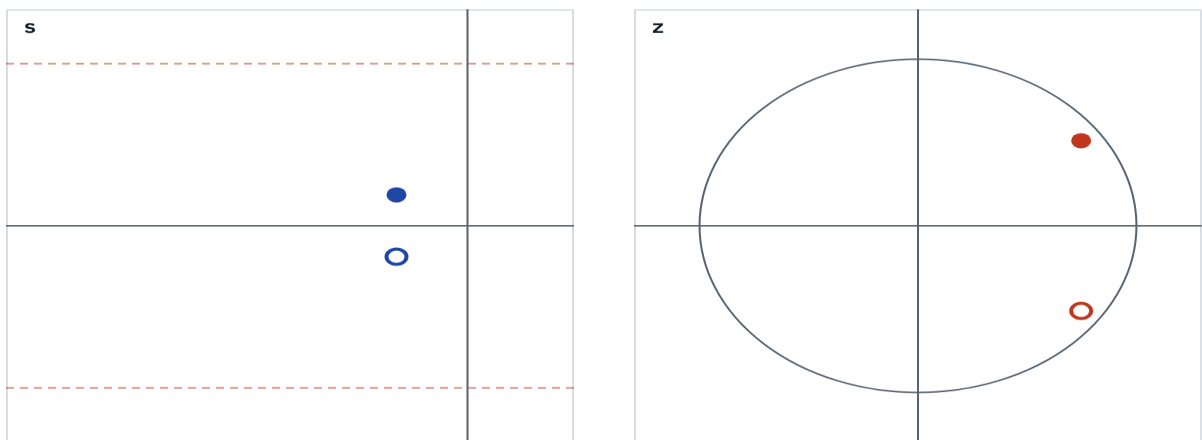
- lewa półpłaszczyzna s (stabilność) → wewnątrz okręgu jednostkowego: układ dyskretny jest stabilny, gdy wszystkie bieguny mają $|z| < 1$;
- oś urojona → okrąg jednostkowy; $z = 1$ to $s = 0$ (całkowanie);
- częstotliwości ω i $\omega + 2\pi/T$ dają ten sam z : to aliasing w języku biegunów.

Odpowiedź na pytanie z prologu. Dron z regulatorem P liczącym co T : w czasie jednego taktu sterowanie jest stałe, więc prędkość dokładnie spełnia $v_{k+1} = a v_k + \frac{1-a}{b} u_k$ z $a = e^{-bT/m}$. Z $u_k = K_p(r - v_k)$ biegun pętli zamkniętej to

$$z = a - K_p \frac{1-a}{b}, \quad |z| < 1 \iff K_p < b \frac{1+a}{1-a}.$$

Dla $T = 0,1$ s: $a = 0,951$ i $K_p < 20$. Ciągły regulator P był stabilny przy każdym wzmocnieniu, cyfrowy nie: przy $K_p > 20$ każdy takt przestrzeliwuje coraz mocniej i pętla oscyluje z okresem dwóch próbek.

SYMULACJA 3 Biegun $s = \sigma \pm j\omega$ i jego obraz $z = e^{sT}$. Na dole ciąg próbek $\operatorname{Re} z^k$. Przesuń σ na prawo od zera: punkt w z wychodzi poza okrąg. Zwiększ ω ponad π/T (przerzywane linie w płaszczyźnie s): obraz zawraca i udaje wolniejsze drganie.



— biegun $s = \sigma + j\omega$ — obraz $z = e^{sT}$ — ciąg $\operatorname{Re} z^k$

$|z| = e^{(\sigma T)} 0,905$

$\arg z = \omega T 34,4^\circ$

$\omega_{\text{Nyquist}} = \pi/T 31,4 \text{ rad/s}$

Lewa półpłaszczyzna s trafia do wewnątrz okręgu jednostkowego. Odległość od środka to tempo zaniku na próbkę, kąt to obrót na próbkę.

Stan na rysunku: $s = -1,00 \pm j6,0$, $T = 100 \text{ ms}$, $z = 0,747 \pm j0,511$.

ZADANIA · CZĘŚĆ 3

3 zadania · ok. 25 min

1 Ciągłe bieguny $-0,5$ oraz $-2 \pm 3j$. Gdzie są w płaszczyźnie z przy $T = 0,1$ s?

2 Policz $K_{p,\max}$ dla cyfrowego regulatora P prędkości drona przy $T = 0,02$ s i $T = 0,5$ s.

3 Biegun pętli zamkniętej $z = -0,5$. Jak wygląda odpowiedź? Czy coś takiego może mieć układ ciągły?

podstawowe średnie feynmanowskie

4 Dyskretyzacja regulatora

Najprostsza droga do regulatora cyfrowego: zaprojektować ciągły (wszystko z części II i III) i zamienić s na przybliżenie w z . Każde przybliżenie to inny sposób liczenia całki (lekcja 6.2 Metody numeryczne: Python i C++): [Goodwin 13.4](#)

metoda	podstawienie	całka I_k	uwagi
Euler w przód	$s \approx \frac{z-1}{T}$	$I_{k+1} = I_k + Te_k$	może zdestabilizować stabilny regulator
Euler wstecz	$s \approx \frac{z-1}{Tz}$	$I_k = I_{k-1} + Te_k$	zawsze stabilny; dobry dla filtru D
Tustin (trapezy)	$s \approx \frac{2}{T} \frac{z-1}{z+1}$	$I_k = I_{k-1} + \frac{T}{2}(e_k + e_{k-1})$	oś urojona $\rightarrow$ okrąg; najwierniejszy

Przy $\omega_c T \lesssim 0,3$ wszystkie trzy dają prawie to samo. Ważniejsze od wyboru metody jest to, żeby w ogóle szybko próbować.

Druga decyzja to postać algorytmu (Kowal 11):

- **pozycyjny**: liczymy całe u_k (jak dotąd);
- **przyrostowy** (prędkościowy): liczymy tylko zmianę $\Delta u_k = u_k - u_{k-1}$, np. dla PI: $\Delta u_k = K_p(e_k - e_{k-1}) + K_i T e_k$. Element wykonawczy sam ją sumuje (silnik krokowy, zawór z siłownikiem). Zaleta: przy nasyceniu nic się nie „nakręca”, bo regulator nie przechowuje całki. Zmiana nastaw w trakcie pracy nie daje skoku sterowania.

ZADANIA · CZĘŚĆ 4

3 zadania · ok. 25 min

1 Wyprowadź algorytm przyrostowy PI z pozycyjnego $u_k = K_p e_k + K_i T \sum_{j \leq k} e_j$.

2 Pokaż, że Tustin odwzorowuje $s = j\omega$ na okrąg jednostkowy, ale z „przesuniętą” częstotliwością: $\omega_{ciągła} = \frac{2}{T} \tan \frac{\omega_{dyskr} T}{2}$.

3 Dlaczego algorytm przyrostowy nie ma windupu, ale ma inny kłopot: nie da się w nim zrobić czystego regulatora P?

podstawowe średnie feynmanowskie

5 PID w kodzie

Wszystko z lekcji 3.1–3.3 w jednej klasie: P od uchybu, D od pomiaru z filtrem (Euler wstecz), całka metodą trapezów, zatrzymanie całki w nasyceniu. Funkcję `step` wywołuje się co T z przerwania zegara.

```
class PID:
    def __init__(self, kp, ki, kd, T, n=8.0, umin=-1e9, umax=1e9):
        self.kp, self.ki, self.kd, self.T = kp, ki, kd, T
        self.tf = kd / (kp * n) if kp > 0 else 0.0 # stała czasowa filtru D
        self.umin, self.umax = umin, umax
        self.i = 0.0; self.d = 0.0; self.e_prev = 0.0; self.y_prev = None

    def step(self, r, y):
        e = r - y
        if self.y_prev is None:
            self.y_prev = y
        a = self.tf / (self.tf + self.T) # D od pomiaru, z filtrem
        self.d = a * self.d - self.kd * (y - self.y_prev) / (self.tf + self.T)
        u = self.kp * e + self.i + self.d
```

```

u_sat = min(max(u, self.umin), self.umax)
# anti-windup: nie całkuj, gdy uchyb pcha dalej w nasycenie
if u == u_sat or (u > self.umax and e < 0) or (u < self.umin and e > 0):
    self.i += self.ki * self.T * (e + self.e_prev) / 2
self.e_prev, self.y_prev = e, y
return u_sat

```

Ta sama logika w C++ (do mikrokontrolera wystarczy zamienić `double` na `float`):

```

struct PID {
    double kp, ki, kd, T, tf, umin, umax;
    double i = 0, d = 0, e_prev = 0, y_prev = 0;
    bool first = true;
    PID(double kp, double ki, double kd, double T, double n, double umin, double umax)
        : kp(kp), ki(ki), kd(kd), T(T), tf(kd / (kp * n)), umin(umin), umax(umax) {}
    double step(double r, double y) {
        double e = r - y;
        if (first) { y_prev = y; first = false; }
        double a = tf / (tf + T);
        d = a * d - kd * (y - y_prev) / (tf + T);
        double u = kp * e + i + d;
        double u_sat = std::clamp(u, umin, umax);
        if (u == u_sat || (u > umax && e < 0) || (u < umin && e > 0)) i += ki * T * (e + e_prev) / 2;
        e_prev = e; y_prev = y;
        return u_sat;
    }
};

```

Test na modelu drona ($T = 10$ ms, ciąg $\pm 9,81$ N): skok 1 m daje 28% przeregulowania, skok 5 m 14%, jak w symulacjach z lekcji 3.3. Pełne pliki z testem są w `tools/verify/c3_4_kod/`.

⚠ PUŁAPKI IMPLEMENTACJI

Stałe T : jeśli pętla nie jest wywoływana równo (system operacyjny, przerwanie o różnym priorytecie), trzeba mierzyć rzeczywiste T albo przenieść regulator do przerwania zegara. Pierwsze wywołanie: bez `y_prev = y` człon D dostałby skok z 0 do pierwszego pomiaru. Liczby całkowite: na mikrokontrolerach bez FPU całkę trzyma się w większej precyzji niż resztę, bo małe przyrosty $K_i T e$ mogą się zaokrągląć do zera.

🔗 ZADANIA · CZĘŚĆ 5

3 zadania · ok. 20 min

- 1 Sprawdź, że filtr D w kodzie to Euler wstecz zastosowany do $\frac{K_d s}{1+T_f s}$ dla sygnału $-y$.
- 2 Przy $K_i = 8$, $T = 1$ ms i uchybie 1 mm ile wynosi przyrost całki w jednym takcie? Dlaczego to może być problem przy typie `float` (ok. 7 cyfr znaczących), gdy całka ma wartość ok. 10 N?
- 3 Dlaczego warunek anti-windup sprawdza, która granica jest aktywna (`u > umax` albo `u < umin`), a nie po prostu porównuje znaki `e` i `u`?

▢▢ podstawowe ▢▢▢ średnie ▢▢▢ feynmanowskie

☰ METODA: OD REGULATORA CIĄGŁEGO DO KODU

1. **Okres:** T tak, żeby $\omega_c T \approx 0,15-0,3$; uwzględnij czas obliczeń.
2. **Antyaliasing:** filtr analogowy (lub w czujniku) z pasmem poniżej $f_s/2$; jego faza też obciąża zapas.
3. **Dyskretyzacja:** Tustin dla całki, Euler wstecz dla filtru D; sprawdź zapas fazy wersji dyskretnej (python-control: `c2d`).
4. **Kod:** D od pomiaru, anti-windup, inicjalizacja przy pierwszym wywołaniu, stałe T .
5. **Sprawdzenie:** symulacja z ZOH i opóźnieniem obliczeń, test na sprzęcie z małymi nastawami.

Przykład 6.1

Jak wolno można liczyć PID drona?

PID 9,6 / 8 / 4,3 z filtrem D ($N = 8$), ciągły: $\omega_c = 4,76$ rad/s, zapas fazy $55,9^\circ$, przeregulowanie ok. 26%. Sprawdź wersje dyskretne (Tustin + ZOH).

1. $T = 5$ ms: zapas $55,2^\circ$, przeregulowanie 26%.
2. $T = 50$ ms: $49,1^\circ$, 31%. $T = 100$ ms: $42,4^\circ$, 39%.
3. $T = 200$ ms: $28,7^\circ$, 65%. $T = 300$ ms: $13,9^\circ$, 113%.
4. Powyżej ok. 0,38 s: niestabilna.

Wynik: do ok. 50 ms różnica jest niewielka; przybliżenie $55,9^\circ - \omega_c T/2$ zgadza się z dokładnym wynikiem z dokładnością do ok. 1° .

Sprawdzenie: symulacja 1 liczy człon D Eulerem wstecz i daje przeregulowanie o 1–3 punkty procentowe większe, ale ten sam trend. Granica stabilności przy $\omega_c T/2 \approx 55^\circ$, czyli $T \approx 0,4$ s ✓.

Przykład 6.2

Wibracje, które udają przechylanie

Akcelerometr drona próbkowany 200 Hz, śmigła dają drgania ramy 190 Hz. Co zobaczy regulator i jak to naprawić?

1. Alias: $|190 - 200| = 10$ Hz, czyli ok. 63 rad/s, w paśmie pętli prędkości kątowej.
2. Regulator „widzi” przechylanie 10 Hz i z nim walczy: silniki zaczynają drgać naprawdę.
3. Naprawa: filtr dolnoprzepustowy w czujniku (wiele układów MEMS ma go wbudowany) poniżej 100 Hz albo próbkowanie np. 1 kHz z filtrem cyfrowym po nim, plus amortyzacja mechaniczna czujnika.

Wynik: 190 Hz przy 200 Hz to 10 Hz fałszywego ruchu; przy 1 kHz drganie zostaje 190 Hz i da się je odfiltrować cyfrowo.

Sprawdzenie: symulacja 2 ✓.

Przykład 6.3

Regulator P, który oscyluje co próbkę

Prędkość drona, regulator P, $T = 0,1$ s, $K_p = 25$. Opisz zachowanie.

- 1 $a = e^{-0,05} = 0,951$, biegun $z = 0,951 - 25 \cdot \frac{0,049}{0,5} = 0,951 - 2,44 = -1,49$.
- 2 $|z| > 1$, ujemny: oscylacja z okresem dwóch próbek (0,2 s), amplituda rośnie 1,49 razy na takt.

Wynik: niestabilna, choć wersja ciągła byłaby stabilna przy każdym K_p . Granica: $K_p = 20$.

Sprawdzenie: dla $K_p = 20$: $z = 0,951 - 1,95 = -1,0 \checkmark$, granica stabilności.

7 Błędne intuicje, których trzeba się oduczyć

× „Szybciej próbkować zawsze lepiej.”

Do pewnego momentu. Bardzo małe T to mniejsze przyrosty (problemy z precyzją), większy wpływ szumu na różnice w członie D i większe obciążenie procesora.

× „Wystarczy próbkować 2 razy szybciej niż najwyższa częstotliwość.”

Twierdzenie o próbkowaniu dotyczy odtwarzania sygnału, nie sterowania. Regulator potrzebuje 20–40 próbek na okres ω_c , bo każda próbka to opóźnienie.

× „Filtr cyfrowy usunie zakłócenia o wysokiej częstotliwości.”

Tylko te poniżej $f_s/2$. Wszystko powyżej już udaje coś innego i trzeba to usunąć przed próbkowaniem.

× „Układ stabilny w wersji ciągłej będzie stabilny po dyskretyzacji.”

Regulator P prędkości drona jest kontrprzykładem. Zawsze sprawdzaj wersję dyskretną.

× „Transformata Z to zupełnie nowa teoria.”

To szereg geometryczny i $z = e^{sT}$. Wszystko z części II ma odpowiednik: okrąg jednostkowy zamiast osi urojonej.

8 Zadania do przemyślenia

Te zadania wymagają więcej myślenia niż liczenia.

ZADANIA DO PRZEMYŚLENIA

5 zadań · ok. 1,5 godziny

- 1 **Deadbeat.** W pętli cyfrowej można umieścić biegun w $z = 0$. Co to znaczy w czasie? Ile wynosiłby ciągły odpowiednik? Dlaczego w praktyce tego się unika?
- 2 **Czas obliczeń.** Regulator czyta pomiar na początku taktu, ale wynik wysyła dopiero pod koniec (obliczenia zajmują prawie całe T). Ile wynosi wtedy łączne opóźnienie? Jak można to poprawić?
- 3 **Euler w przód rozkołysze.** Regulator ciągły zawiera człon $\frac{1}{0,01s+1}$ (filtr o stałej 10 ms). Dyskretyzujemy go Eulerem w przód z $T = 25$ ms. Co się stanie?

- 4 **Kwantyzacja przetwornika.** 12-bitowy przetwornik A/C mierzy wysokość w zakresie 0–100 m. Jaka jest rozdzielczość?
Co zobaczy człon D liczony co 10 ms?
- 5 **Jedna pętla, wiele częstotliwości.** Kontroler lotu liczy pętlę prędkości kątowej z częstotliwością 2 kHz, orientacji 250 Hz, pozycji 50 Hz, a GPS daje pomiar 5 Hz. Czy pętla pozycji z $\omega_c \approx 1$ rad/s może korzystać z GPS 5 Hz? Co trzeba zrobić?

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

9 Sprawdź się

Odpowiedz w myślach, zanim rozwiniesz odpowiedź.

- Ile fazy kosztuje ZOH?**
Ok. $\omega T/2$, jak opóźnienie o pół okresu.
- Ile próbek na okres ω_c ?**
20–40, czyli $\omega_c T \approx 0,15$ –0,3.
- Co to aliasing i jak go uniknąć?**
Sygnał powyżej $f_s/2$ udaje sygnał o częstotliwości $|f - nf_s|$. Filtr antyaliasingowy przed próbkowaniem.
- Jaki jest związek s i z ?**
 $z = e^{sT}$; lewa półpłaszczyzna $\rightarrow$ wnętrze okręgu jednostkowego.
- Kiedy układ dyskretny jest stabilny?**
Gdy wszystkie bieguny mają $|z| < 1$.
- Ile wynosi $Z\{a^k\}$?**
 $\frac{z}{z-a}$, z sumy szeregu geometrycznego.
- Podstawienie Tustina?**
 $s \approx \frac{2}{T} \frac{z-1}{z+1}$, całkowanie metodą trapezów.
- Czym różni się algorytm przyrostowy od pozycyjnego?**
Liczy zmianę sterowania Δu_k ; element wykonawczy ją sumuje; brak windupu.

PRÓBKOWANIE

ZOH: $\frac{1-e^{-sT}}{sT}$, faza $-\omega T/2$

$\omega_c T \approx 0,15-0,3$; strata fazy $\omega_c T/2$

Nyquist: $f_s/2$; alias $|f - n f_s|$

$z = e^{sT}$, stabilność: $|z| < 1$

$\mathcal{Z}\{a^k\} = \frac{z}{z-a}$, $\mathcal{Z}\{x_{k-1}\} = z^{-1}X$

DYSKRETYZACJA I KOD

Euler w przód $\frac{z-1}{T}$; wstecz $\frac{z-1}{Tz}$; Tustin $\frac{2}{T} \frac{z-1}{z+1}$

całka (trapezy): $I_k = I_{k-1} + \frac{K_d T}{2}(e_k + e_{k-1})$

filtr D: $d_k = \frac{T_f}{T_f+T}d_{k-1} - \frac{K_d}{T_f+T}(y_k - y_{k-1})$

przyrostowy PI: $\Delta u_k = K_p(e_k - e_{k-1}) + K_i T e_k$

dron, P cyfrowy: $K_p < b \frac{1+a}{1-a}$, $a = e^{-bT/m}$

Do przeczytania

- **Obowiązkowo:** J. Kowal, *Podstawy automatyki*, rozdz. 11 (sterowanie cyfrowe, algorytm pozycyjny i przyrostowy, po polsku); G. C. Goodwin i in., *Control System Design*, rozdz. 12 (próbkowanie) i 13.4.
- **Dla chętnych:** K. J. Åström, B. Wittenmark, *Computer-Controlled Systems*, klasyczny podręcznik sterowania cyfrowego; NASA, materiały o programie F-8 Digital Fly-By-Wire (strona NASA Armstrong).

To koniec części III. Umiemy zaprojektować, nastroić i zaprogramować regulator PID dla jednego wejścia i jednego wyjścia. Dron w przestrzeni ma jednak kilka wejść (silniki), kilka wyjść (położenie, orientacja) i czujniki, które nie mierzą wszystkiego, czego potrzeba. Część IV przechodzi do opisu w przestrzeni stanów: sprzężenie od stanu, obserwatory, sterowanie optymalne i filtr Kalmana.



Odpowiedzi

Zadania • część 1

- $\omega_c T/2$ w stopniach: $2,7^\circ$, $13,6^\circ$, $27,3^\circ$. Zostaje ok. 53° , 42° , 29° (python-control z dokładną dyskretyzacją: $53,1^\circ$, $42,4^\circ$, $28,7^\circ$).
- Okres $2\pi/50 \approx 0,126$ s, czyli $T \approx 3\text{--}6$ ms ($f_s \approx 160\text{--}320$ Hz). W praktyce kontrolery lotu liczą tę pętlę jeszcze szybciej (1–8 kHz), bo dochodzą filtry i opóźnienia czujników.
- Schodek zaczyna się dokładnie w chwili próbki (opóźnienie 0) i kończy po T (opóźnienie T). Średnio sygnał jest przesunięty o $T/2$.
Ściśle: transmitancja ZOH to $\frac{1-e^{-sT}}{sT}$, a jej faza wynosi dokładnie $-\omega T/2$.

Zadania • część 2

- $|210 - 200| = 10$ Hz; $|50 - 60| = 10$ Hz.
- Gdy między klatkami koło obraca się o wielokrotność 60° : 4 obroty/s (i wielokrotności). Szprychy mijają wtedy pozycję z częstotliwością 24 Hz, równą częstotliwości klatek.
- Po próbkowaniu alias 10 Hz i prawdziwy ruch 10 Hz mają identyczne próbki. Żaden algorytm nie odróżni ich na podstawie tych samych liczb. Rozróżnić je można tylko przed próbkowaniem, gdy sygnał jeszcze ma swoją prawdziwą częstotliwość.

Zadania • część 3

- $e^{-0,05} = 0,951$; $e^{-0,2}e^{\pm 0,3j} = 0,819 e^{\pm 17,2^\circ j} \approx 0,782 \pm 0,242j$.
- $T = 0,02$; $a = 0,990$, $K_p < 100$. $T = 0,5$; $a = 0,779$, $K_p < 4$.
- $(-0,5)^k$: znak zmienia się co próbkę, amplituda maleje o połowę. Drganie z okresem dwóch próbek (częstotliwość Nyquista). Układ ciągły nie ma odpowiednika: e^{sT} nigdy nie jest ujemną liczbą rzeczywistą, chyba że $\omega T = \pi$. To typowy objaw za dużego wzmocnienia w pętli cyfrowej.

Zadania • część 4

- $u_k - u_{k-1} = K_p(e_k - e_{k-1}) + K_i T e_k$: suma skraca się do ostatniego składnika.
- Dla $z = e^{j\theta}$: $\frac{z-1}{z+1} = \frac{e^{j\theta/2} - e^{-j\theta/2}}{e^{j\theta/2} + e^{-j\theta/2}} = j \tan \frac{\theta}{2}$. Czyli $s = j \frac{2}{T} \tan \frac{\omega T}{2}$, czysto urojone. Przy małym ωT $\tan x \approx x$ i zgodność jest dobra; dla $\omega = 10$, $T = 0,1$: 10,9 zamiast 10.
- Element wykonawczy sumuje przyrosty: gdy jest w nasyceniu, odrzuca nadmiar i nic się nie odkłada. Za to „P” w postaci przyrostowej $\Delta u = K_p \Delta e$ zsumowane daje $u = K_p e + u_0$, zależne od stanu początkowego, a dryf zaokrągleń kumuluje się bez członu, który by go korygował. W praktyce algorytm przyrostowy zawsze ma człon I.

Zadania • część 5

- $T_f \dot{d} + d = -K_d \dot{y}$. Euler wstecz: $T_f \frac{d_k - d_{k-1}}{T} + d_k = -K_d \frac{y_k - y_{k-1}}{T}$. Mnożymy przez T i dzielimy przez $T_f + T$: $d_k = \frac{T_f}{T_f + T} d_{k-1} - \frac{K_d}{T_f + T} (y_k - y_{k-1})$. ✓.
- $8 \cdot 0,001 \cdot 0,001 = 8 \cdot 10^{-6}$ N. Przy wartości ok. 10 N sąsiednie liczby `float` różnią się o ok. $9,5 \cdot 10^{-7}$, więc przyrost $8 \cdot 10^{-6}$ to tylko ok. 8 takich kroków i jest zaokrąglany z błędem ok. 5% w każdym takcie (zawsze w tę samą stronę). Uchyby poniżej ok. 0,06 mm dają przyrost mniejszy niż pół kroku i giną całkowicie.
- Porównanie znaków działa tylko wtedy, gdy zakres $[u_{\min}, u_{\max}]$ zawiera zero (jak symetryczna korekta ciągu wokół zawisu). Dla zakresu np. 1000–2000 μ s (sygnał PWM) sterowanie poniżej dolnej granicy jest dodatnie, a uchyb ujemny: porównanie znaków pozwoliłoby całce dalej maleć, czyli nakręcać się w nasycenie. Sprawdzenie aktywnej granicy działa dla dowolnego zakresu. Porównanie `u == u_sat` jest bezpieczne, bo `u_sat` jest albo dokładnie `u`, albo jedną z granic.

Zadania do przemyślenia

1. $z^k = 0$ dla $k \geq 1$: odpowiedź ustala się dokładnie w skończonej liczbie kroków (ang. *deadbeat*). Ciągły odpowiednik: $e^{sT} = 0$, czyli $s = -\infty$, nieskończenie szybki biegun.
- Wymaga ogromnych sterowań (dla drona przy $T = 0,1$ s: $K_p = ab/(1-a) \approx 9,75$, a przy krótszym T jeszcze więcej) i dokładnego modelu. Między próbkami obiekt może się zachowywać źle (ukryte oscylacje).
1. T (obliczenia) + $T/2$ (ZOH) = $1,5T$. Strata fazy trzy razy większa niż z samego ZOH.
- Wysłać wynik od razu po obliczeniu (a nie na początku następnego taktu), przenieść obliczenia niezależne od nowego pomiaru przed odczyt, uwzględnić opóźnienie w modelu (predyktor Smitha, lekcja 2.4).
1. Euler w przód: $x_{k+1} = x_k + \frac{T}{0,01}(u_k - x_k) = (1 - 2,5)x_k + 2,5u_k$, biegun $z = -1,5$.
2. Niestabilny filtr, choć ciągły był stabilny. Euler w przód wymaga $T < 2\tau$ dla każdego członu. Euler wstecz i Tustin nie mają tego problemu.

4. 1. $100/4096 \approx 2,4$ cm.
2. Zmiana o jeden poziom w ciągu 10 ms to 2,4 m/s „prędkości”. Przy powolnym wznoszeniu 0,1 m/s człon D widzi przez większość taktów zero, a co ok. 0,24 s szpilkę 2,4 m/s. Potrzebny silny filtr D, czujnik prędkości albo estymator (filtr Kalmana, lekcja 4.4 Filtr Kalmana).
5. 1. 5 Hz to ok. 31 próbek na okres 1 rad/s (2π s): reguła 20–40 jest spełniona, ale do tego GPS ma opóźnienie rzędu 0,1–0,2 s.
2. W praktyce pozycję między pomiarami GPS przewiduje się z akcelerometru (fuzja czujników, filtr Kalmana), a pomiar GPS tylko ją koryguje.
- To temat lekcji 4.4 Filtr Kalmana i 4.5 Rozszerzony filtr Kalmana i fuzja czujników w jednej osi.

Automatyka - Lekcja 3.4: Sterowanie cyfrowe - Źródła: G. C. Goodwin, S. F. Graebe, M. E. Salgado, *Control System Design* (Prentice Hall 2001), rozdz. 12–13; K. J. Åström, B. Wittenmark, *Computer-Controlled Systems* (Prentice Hall); J. Kowal, *Podstawy automatyki* (AGH), rozdz. 11; NASA, F-8 Digital Fly-By-Wire. Wszystkie liczby policzone na nowo (tools/verify/c3_4.py, c3_4_kod/).