LEGO® MINDSTORMS® EV3

REGULACIJE

UROŠ LOTRIČ
FRI

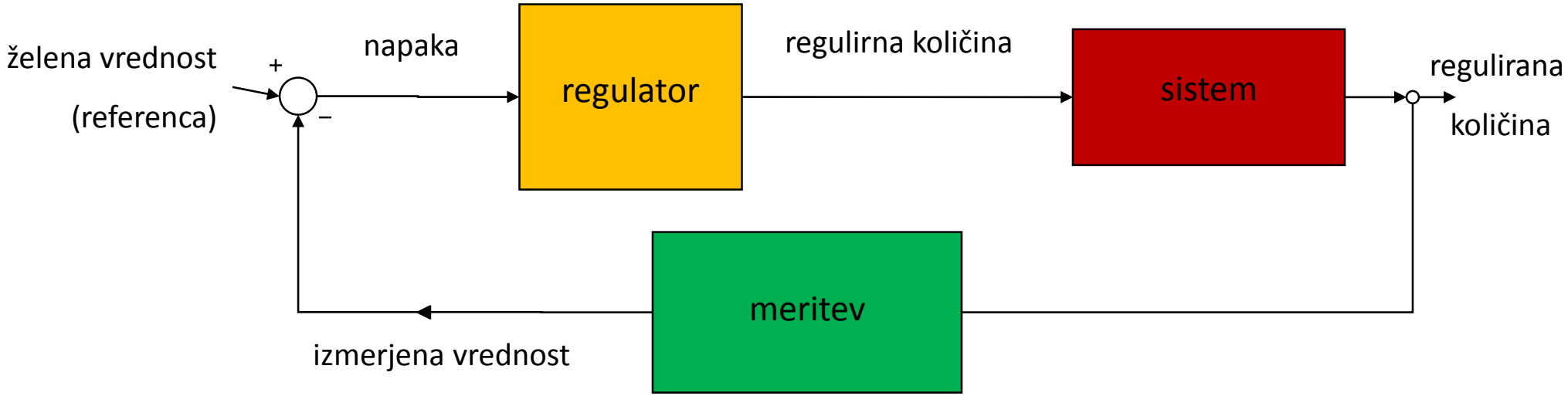
REGULACIJA

Zaprto-zančni regulator

Poskuša voditi proces tako, da poskuša merjeno vrednost kar najbolj približati želeni vrednosti

Odpravljanje vpliva motenj

Sledenje spremembam želene vrednosti



REGULACIJA

Standardni regulatorji
- Zahtevajo matematični model procesa
- Primeri: dvotočkovni, PID

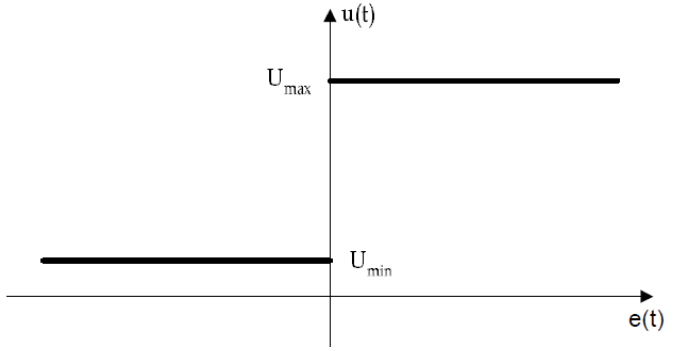
Moderni pristopi
- Model procesa ustvarijo iz meritev
- Primeri: mehka logika, nevronske mreže

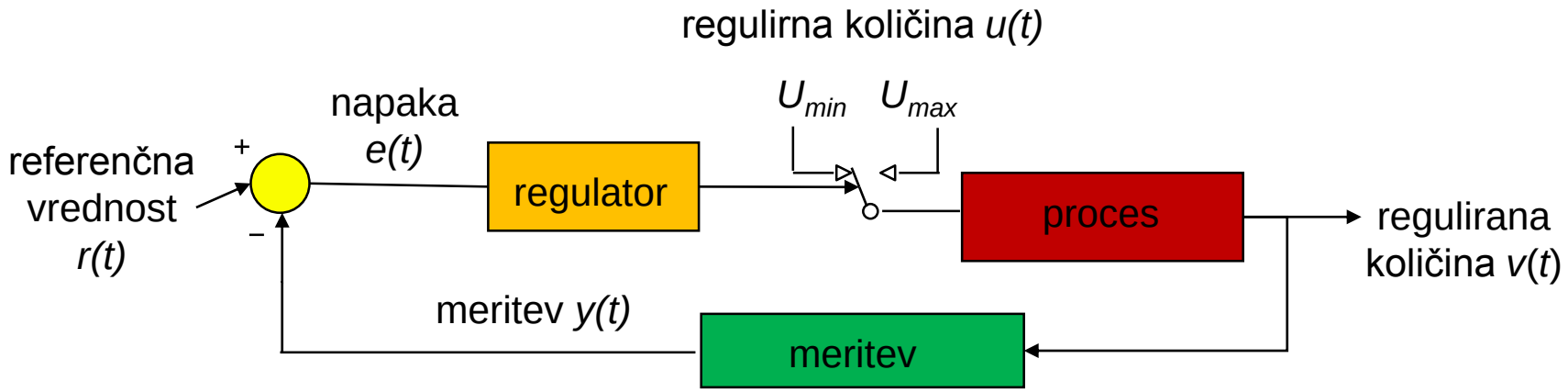
Praksa
- Mnogi realni procesi so nelinearni in jih je težko matematično opisati
- V praksi se je pokazalo, da se večino nelinearnih procesov obvladuje z regulatorji PID
- Pri bolj zapletenih procesih kombiniramo več regulatorjev PID

DVOTOČKOVNI REGULATOR

Dvotočkovni regulator glede na vrednost napake e(t) = r(t) – y(t) preklaplja med dvema vrednostma regulirne količine u(t)

$$u(t) = \begin{cases} U_{max}; e(t) \geq 0 \\ U_{min}; e(t) < 0 \end{cases}$$

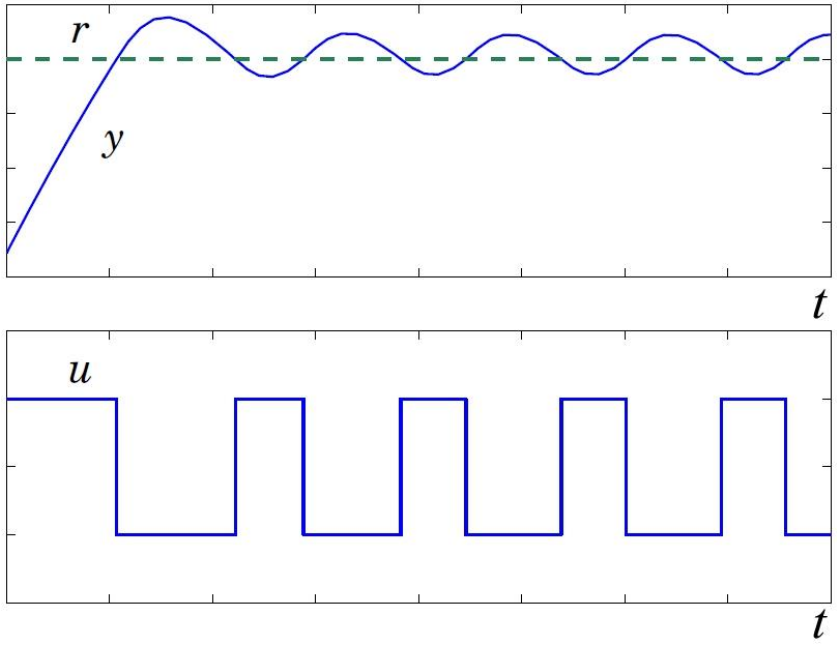




DVOTOČKOVNI REGULATOR

Problem

Regulirana količina neprestano oscilira okrog referenčne vrednosti

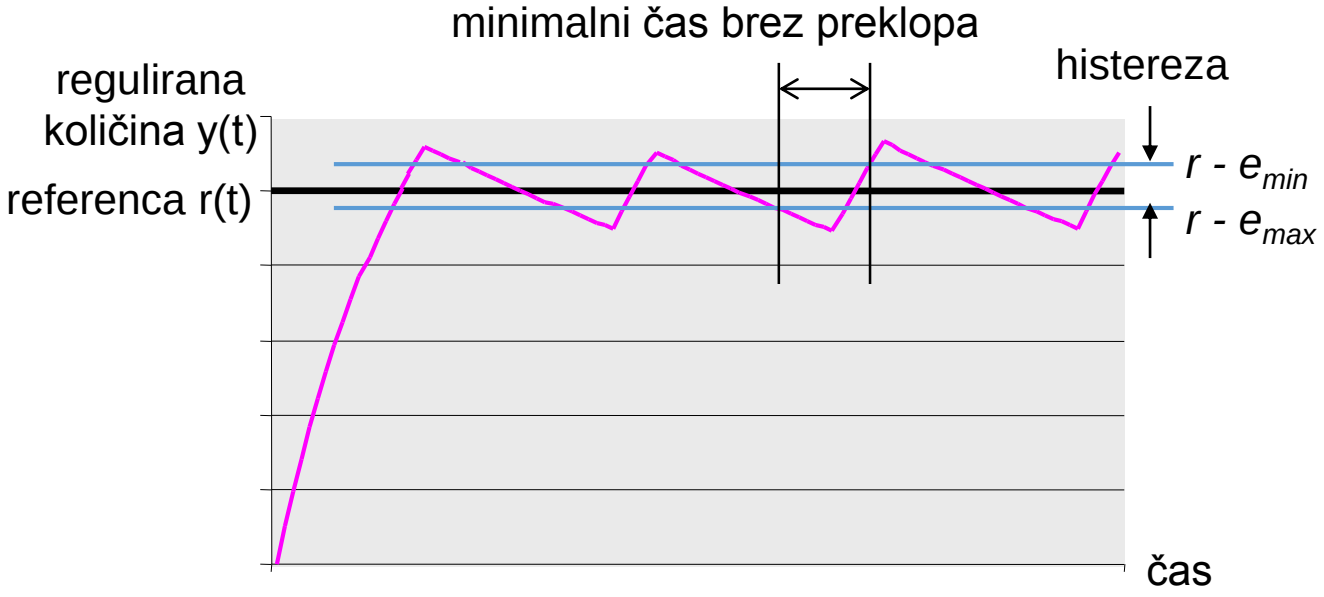


Histerezna zanka

Bolj ohlapne meje za preklop

Minimalni čas brez preklopa

$$u(t) = \begin{cases} U_{max}; e(t) > e_{max} \\ U_{min}; e(t) < e_{min} \end{cases}$$

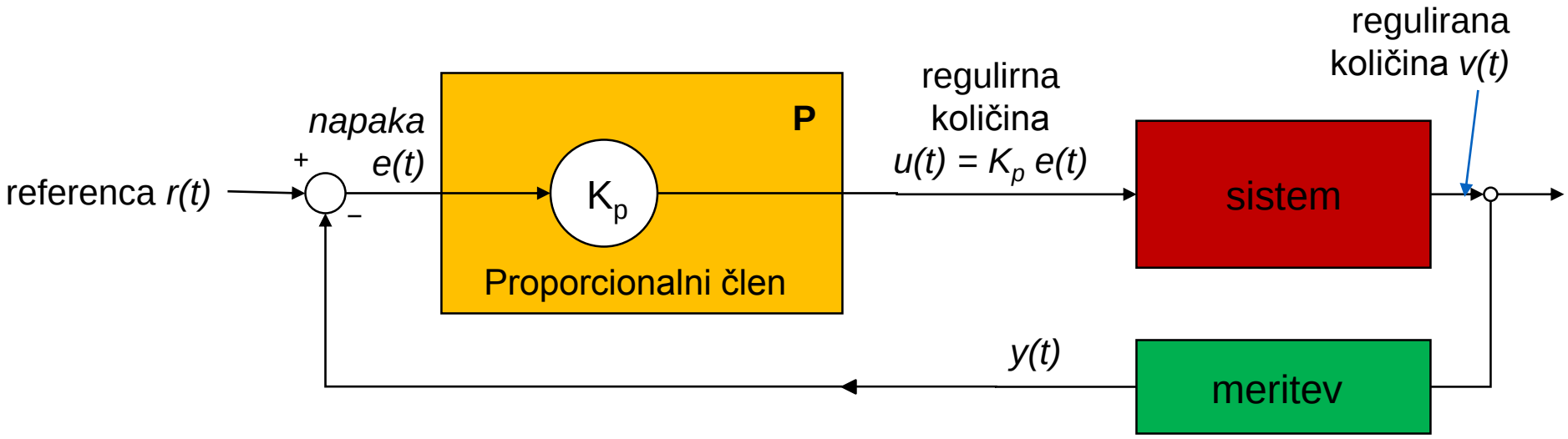


REGULATOR P

Najpreprostejši zvezni regulator

Posplošitev diskretnega dvotočkovnega regulatorja

Regulirna količina je proporcionalna trenutni napaki

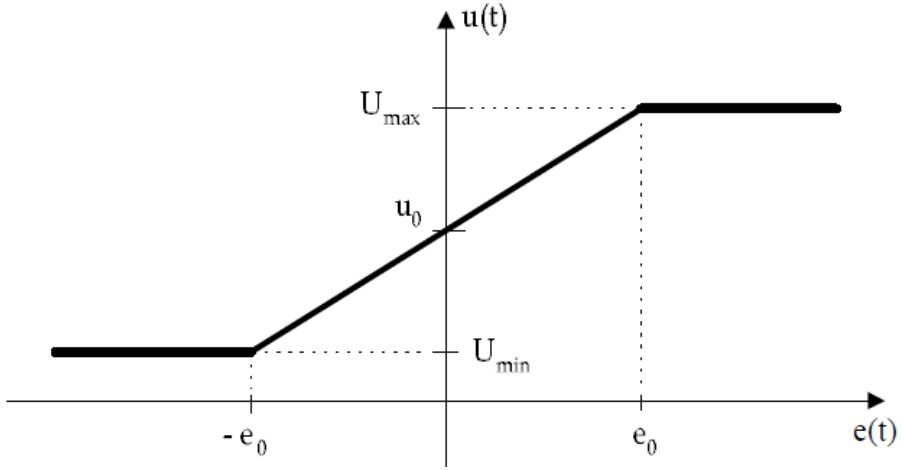


REGULATOR P

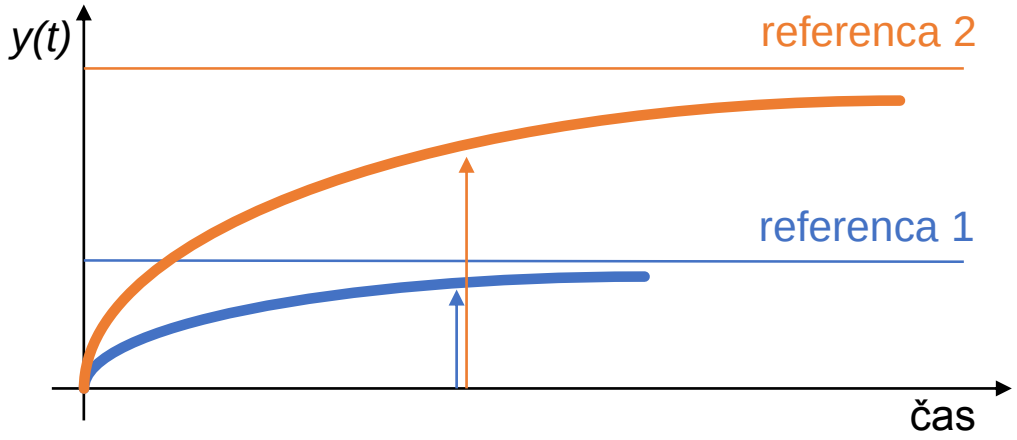
Zaradi fizikalnih omejitev je regulirna količina navzgor in navzdol omejena

Enačba

$$u(t) = \begin{cases} U_{max}; e(t) > e_0 \\ U_0 + K_p\, e(t); e(t) < e_{mi} \\ U_{min}; e(t) < -e_0 \end{cases}$$



Odziv



majhna napaka → majhen popravek

velika napaka → velik popravek

REGULATOR P

Z regulatorjem P lahko odstranimo oscilacije, ki jih povzroči dvotočkovni regulator

Problem: napaka v mirovanju

V delovnem območju regulatorja $[-e_0, +e_0]$ velja $u(t) = U_0 + K_p\, e(t) \quad \rightarrow \quad e(t) = \frac{u(t) - U_0}{K_p}$
Napaka je lahko nič, če je

$K_p = \infty$: dvotočkovni regulator, ki pripelje nazaj oscilacije

$u(t) = U_0$: kadar lahko za vsako referenčno vrednost r(t) nastavimo svojo U_0

Z večanjem konstante K_p dosežemo

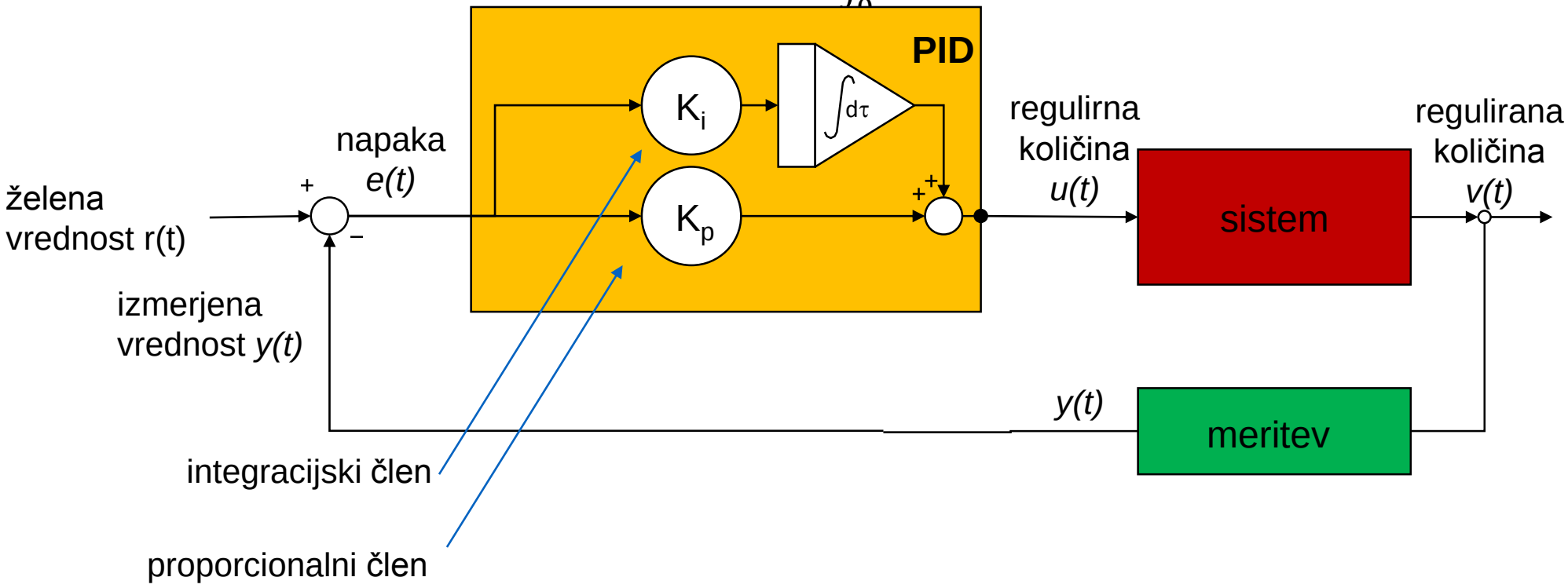
Manjšo napako v mirovanju in s tem boljše sledenje referenčni vrednosti
Manjšo amplitudo napake
V proces vnesemo hitrejšo dinamiko in s tem večjo občutljivost na merski šum

REGULATOR PI

Rešitev

Regulatorju dodamo integracijski člen

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau$$

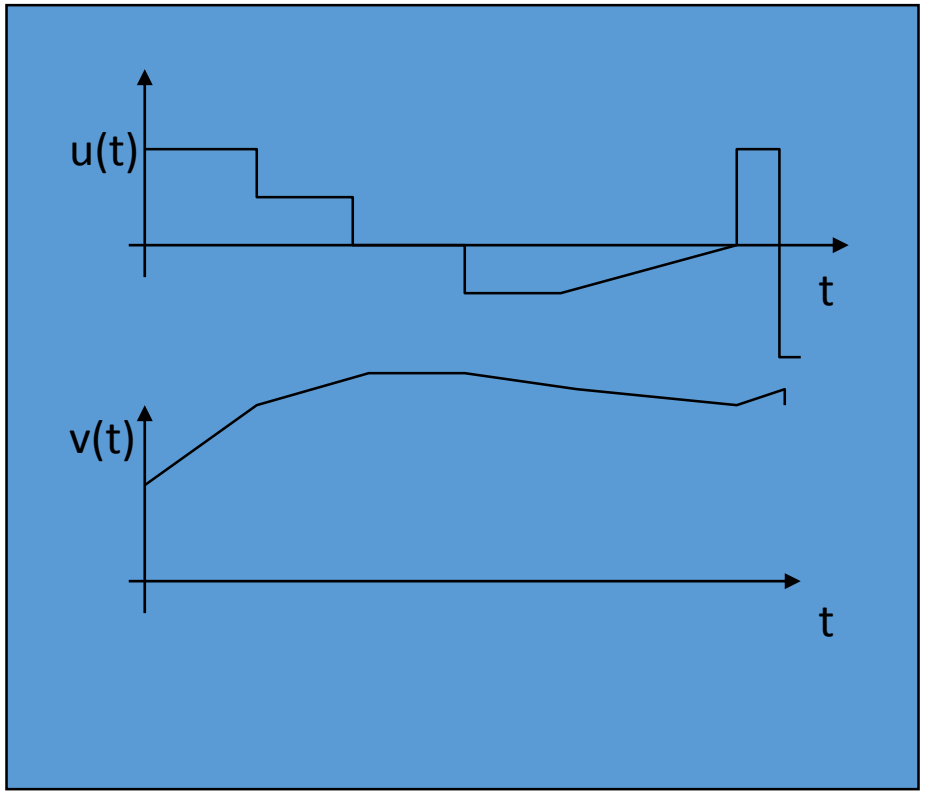


REGULATOR PI

Integracijski člen na delu

Enačba:

$$v(t) = K_i \int_0^t u(\tau) d\tau$$



REGULATOR PI

Odziv integracijskega člena ni vezan na trenutno napako,
temveč na vsoto vseh prejšnjih napak

Primerjava

Regulator P: $u(t) = K_p e(t) + U_0$

Regulator PI: $u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau$

Zamenjava U_0 z integralom nam omogoča izničenje napake v mirovanju

Regulator PI bo odstranil oscilacije, značilne za dvotočkovni regulator, in napako v mirovanju, značilno za regulator P

Slabost: uvedba integracijskega člena privede do počasnejšega odziva in slabo vpliva na stabilnost celotnega sistema

REGULATOR PI

Regulator PI se večkrat zapiše nekoliko drugače:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau$$

$$u(t) = K_p \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau \right]$$

Konstanta T_i se imenuje tudi čas ničenja (ang. reset time)

Z integracijskim členom si regulator zapomni, kaj se je dogajalo v preteklosti

Regulator I skoraj nikoli ne nastopa samostojno, vedno samo kot dodatek k regulatorju P

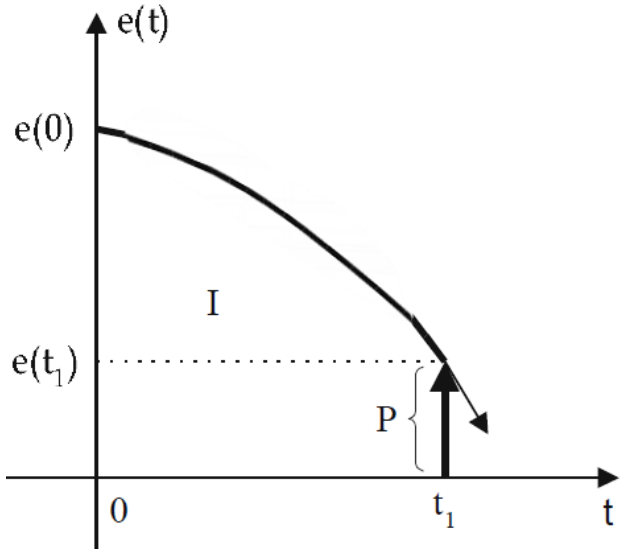
Da ne pride do težav zaradi zasičenja integracijskega člena, ga je dobro omejiti

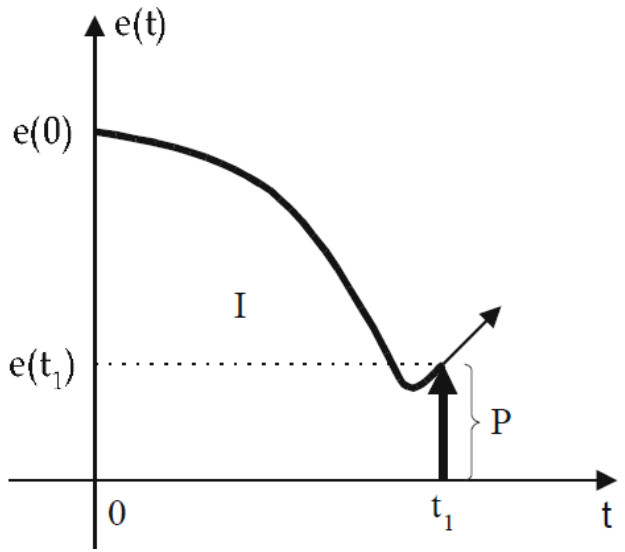
REGULATOR PI

Primerjava dveh potekov

proporcionalni in integralni člen sta v obeh primerih enaka

razlika je v smeri gibanja napake v času t_1





Regulator PI ne zazna razlike med levim in desnim potekom in se bo v obeh primerih odzval enako

Smer gibanja napake (napoved) nam pove tangenta na krivuljo oziroma odvod

REGULATOR D

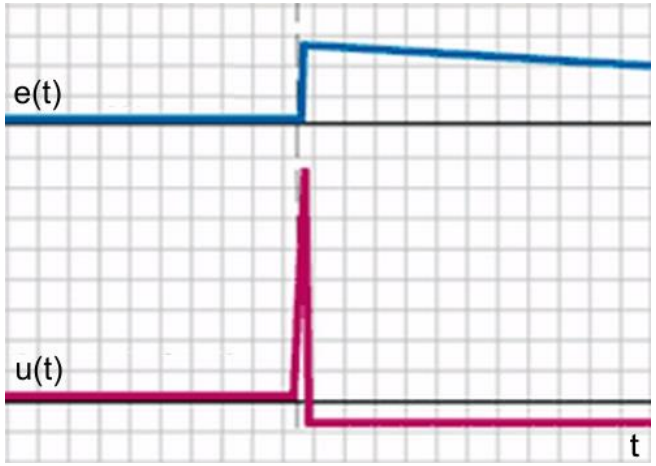
Napaka in njen odvod

- Večja kot je sprememba napake, večji je odziv
- Velikost odziva ustreza naklonu tangente na časovni funkciji napake

Z naklonom premice si lahko pomagamo pri napovedovanju naslednje vrednosti

Diferencialni člen

$$u(t) = K_d \frac{de(t)}{dt}$$



REGULATOR D

Ob nenadni spremembi lahko z diferencialnim členom povzročimo močno korekcijo regulirane količine in tako v kar največji meri odstranimo napako

Na ta način mnogim procesom močno povečamo odzivnost

Popravek je v pravi smeri, ni točen, dosežemo ga mnogo hitreje kot z regulatorjem PI

Eden največjih problemov diferencialne regulacije je šum

- Regulacije P šum ne moti, regulacija I ga s povprečenjem nevtralizira
- Regulacija D ojači nenadne spremembe in s tem tudi visoko frekvenčni šum

Regulator D nikoli ne nastopa samostojno, saj ne more odstraniti napake

Po prehodnem obdobju vpliv diferencialnega člena zamre, regulacijo lahko dobro zaključimo v kombinaciji z regulatorjem PI

REGULATOR PID

Regulator PID je posplošitev dvotočkovnega regulatorja

Regulatorji PID uporabljajo tri režime dela:

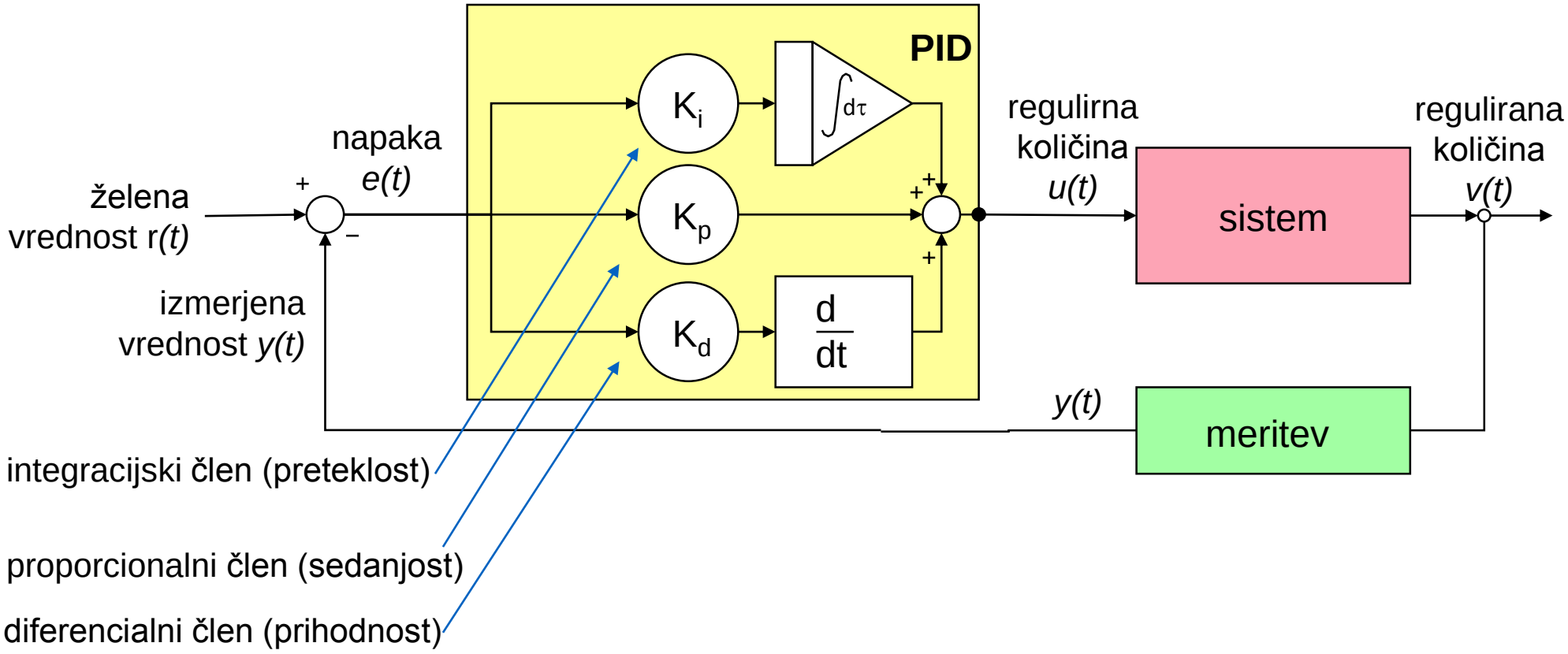
- P – proporcionalni (odziv na trenutno stanje)
- I – integracijski (odziv na dogajanje v preteklosti)
- D – diferencialni (odziv na dogajanje v prihodnosti)

Načina P in I se lahko uporabljata samostojno, način D skoraj ne

Kombinacije načinov PI, PD in PID so v praksi zelo pogoste

REGULATOR PID

Enačba: $u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$



REGULATOR PID

Regulator PID se lahko zapiše nekoliko drugače:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$$

$$u(t) = K_p \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau)d\tau + T_d \frac{de(t)}{dt} \right]$$

Regulator PID ima vso potrebno dinamiko

primerno obnašanje znotraj kontrolnega območja $|e(t)| < e_0$ za odstranjevanje oscilacij (P)

ničenje napake za gibanje proti referenčni točki (I)

hitro reakcijo na spremembe napake (D)

REGULATOR PID

Integralni in diferencialni člen v sistem uvedeta nestabilnost

Kompleksnost

- Kljub temu, da razumemo, kaj je naloga vsakega člena posebej, je zelo težko nastaviti vse tri parametre tako, da bodo delovali usklajeno
- Nastaviti jih moramo za vsako aplikacijo posebej – večkrat je težko (nemogoče) izmeriti vse potrebne parametre za dobro nastavitev

Sreča v nesreči

- 90 % aplikacij v industriji deluje dobro kljub temu, da parametri niso optimalno nastavljeni

REGULATOR PID

Zvezno in diskretno

Danes je večina regulatorjev PID izvedena programsko

Diskretizacija enačb

integral → vsota: $\int e(t)dt \rightarrow \sum e(t)T_s$

odvod → diference: $\frac{de(t)}{dt} \rightarrow \frac{\Delta e}{T_s}$

Na integralni in diferencialni člen ima zelo močan vpliv čas vzorčenja

Čas vzorčenja naj bo med 1/10 in 1/1000 pričakovanega odzivnega časa

REGULATOR PID

Sledenje črti
z diskretnim
regulatorjem PID

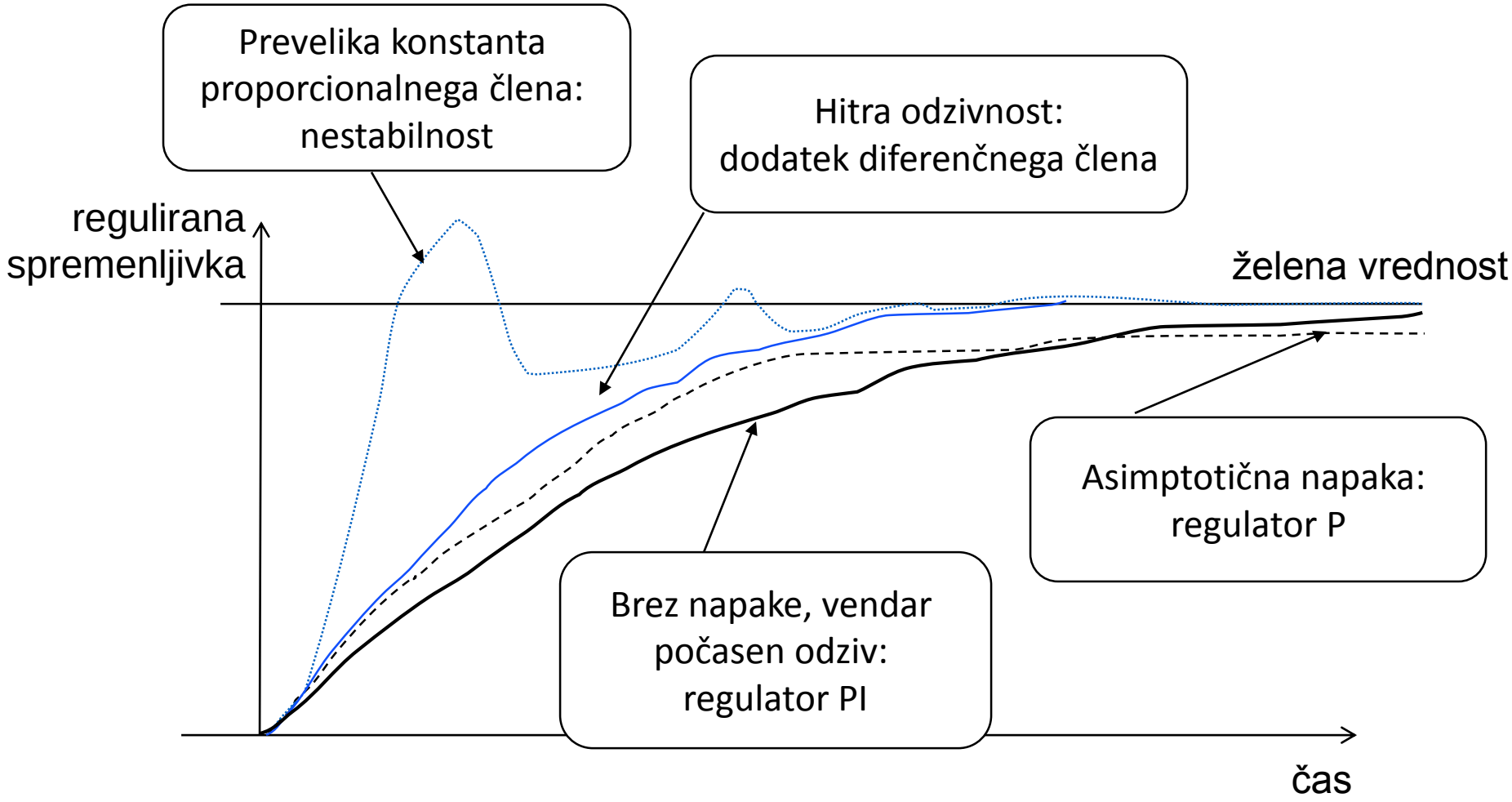
crta_pid.c

crta_pid.ev3

```
task main()
{
  int hitrostPopravek;
  int temno, svetlo, prag;
  int napaka, napakaStara, napakaSprememba;
  float integral, cas;
  float clenP, clenI, clenD;

  // hitrost voznje
  int hitrost = 50;

  // konstante regulatorja PID
  float Kp = 0.5;
  float Ti = 5;
  float Td = 0;

  // dolocitev praga
  temno = SensorValue[S4];
  moveMotor(motorB, 90, degrees, 20);
  wait(1000, milliseconds);
  svetlo = SensorValue[S4];
  moveMotor(motorB, -90, degrees, 20);
  prag = (temno + svetlo) / 2;

  // zacetne vrednosti
  integral = 0.0;
  napakaStara = SensorValue[S4] - prag;
  clearTimer(T1);

  // glavna zanka
  repeat(forever)
  {
    // cas cikla
    cas = time1[T1] / 1000.0;
    clearTimer(T1);

    // napaka
    napaka = SensorValue[S4] - prag;
    napakaSprememba = napaka - napakaStara;
    napakaStara = napaka;

    // proporcionalni clen
    clenP = Kp * napaka;

    // integralni clen
    integral = integral + napaka * cas;
    clenI = Kp / Ti * integral;

    // diferencialni clen
    if (cas > 0)
      clenD = Kp * Td * napakaSprememba/ cas;
    else
      clenD = 0.0;

    // izracun regulirne kolicine
    hitrostPopravek = (int)(clenP + clenI + clenD);

    // sprememba reguliranih kolicin (hitrost motorjev)
    setMotor(motorB, hitrost - hitrostPopravek);
    setMotor(motorC, hitrost + hitrostPopravek);
  }
}
```

REGULATOR PID

Kako nastaviti regulator PID?



REGULATOR PID

Pristop s poskušanjem

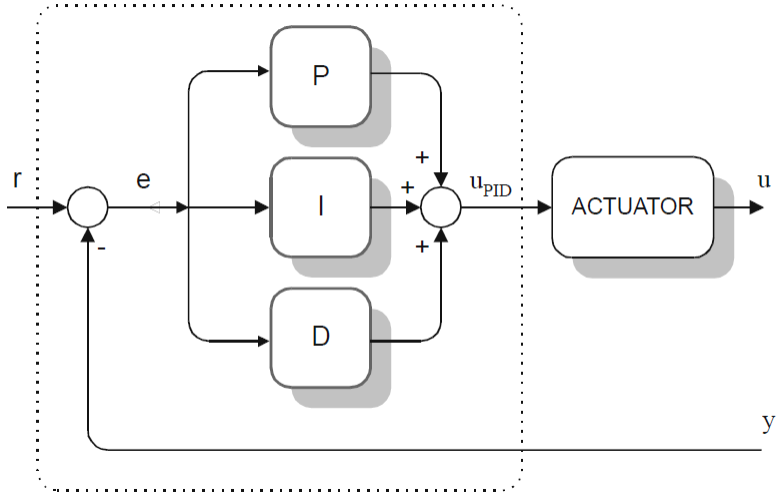
Vse člene postavimo na nič, nato pa jih dodajamo postopno

- Proporcionalni člen
  - Konstanto K_p postavimo na tako vrednost, da sistem oscilira
  - Nato konstanto zmanjšujemo za faktor 8 do 10 dokler oscilacije ne izzvenijo
  - Z manjšimi spremembami (faktor 2) poiščemo všečen odziv
- Integralni člen
  - Začnemo z majhnimi vrednostmi T_i, na primer 1000, nato pa s faktorji 10 vrednost
  - Zmanjšujemo dokler odziv ni všečen
- Diferencialni člen
  - Poiščemo točko, kjer sistem začne oscilirati, nato T_d postavimo za faktor 10 nižje
  - Nadaljnje popravke delamo za faktor 2 navzgor ali navzdol

REGULATOR PID

Arhitekture

Paralelna



Zaporedna