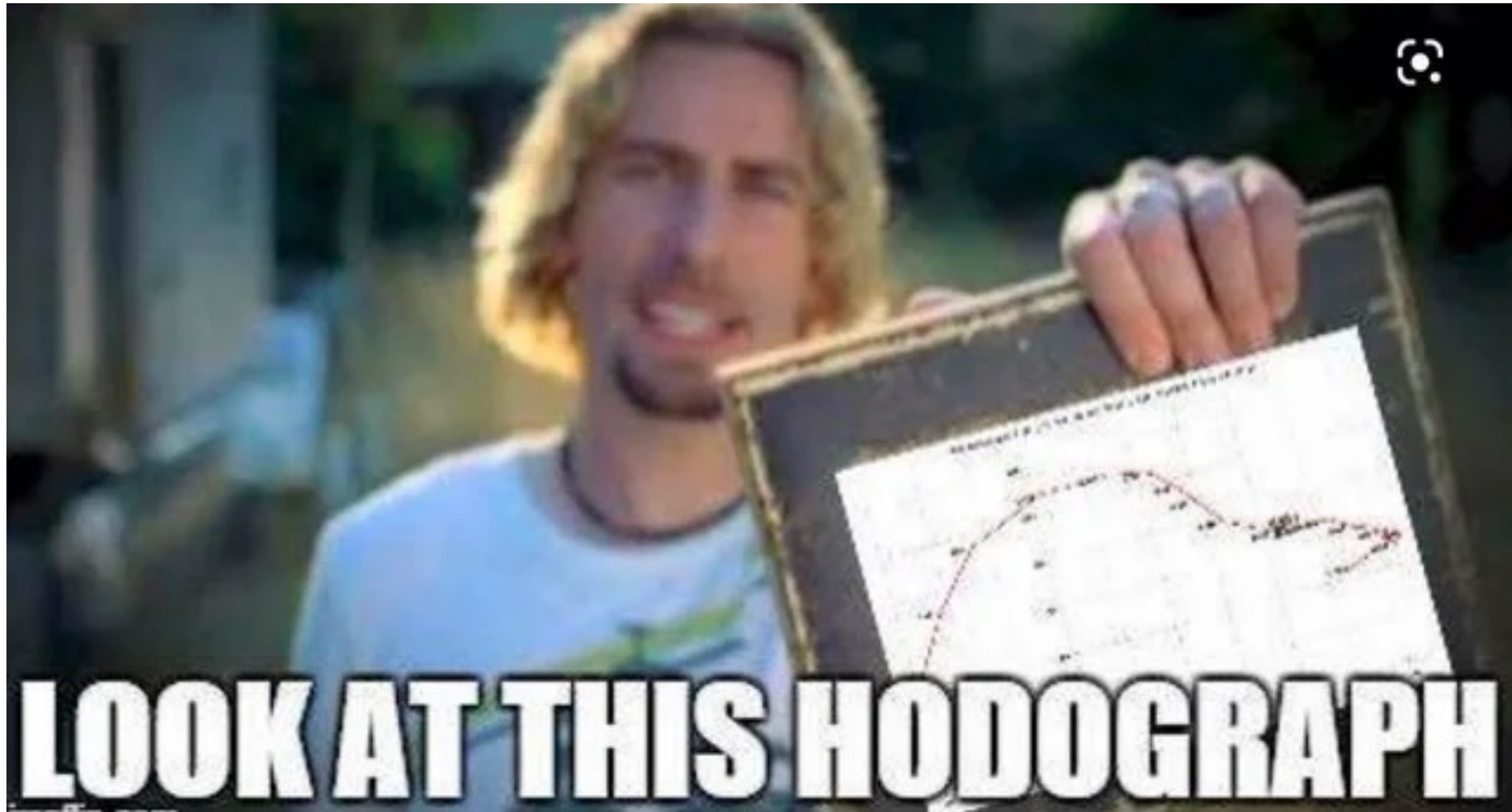


Sisteme Încorporate

Cursul 7

Sisteme de Control

Facultatea de Automatică și Calculatoare
Universitatea Politehnica București

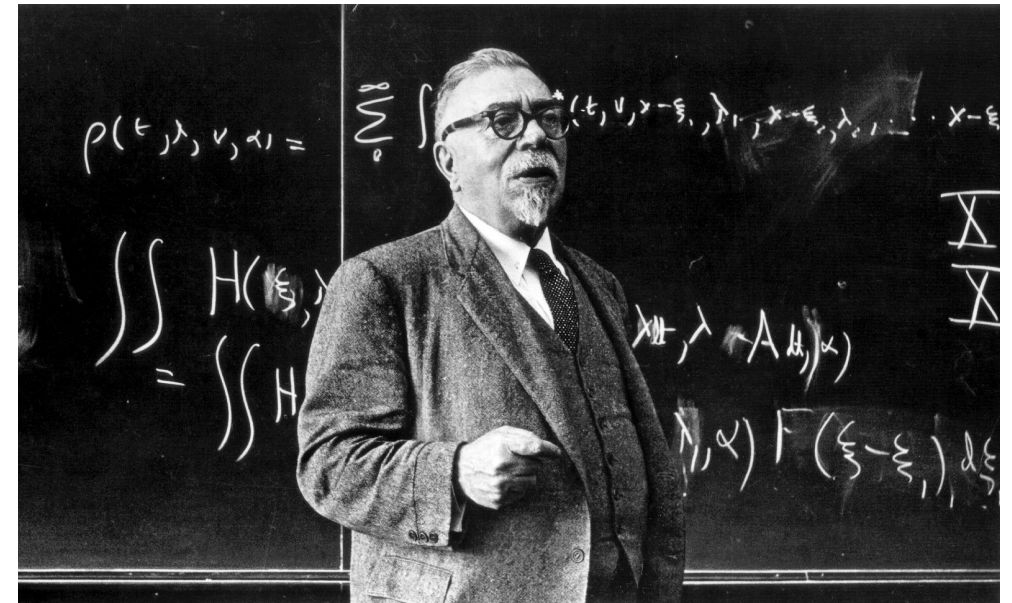


- Un sistem încorporat este un sistem dinamic
 - Sistemul acționează în funcție de stimulii primiți
 - Atunci când una sau mai multe ieșiri ale sistemului trebuie să se conformeze anumitor reguli, un controller manipulează intrarea sistemului pentru a aduce ieșirile la valorile dorite.
 - Sistemul poate fi afectat de perturbații exterioare
 - Parametrii de intrare trebuie calculați în funcție de erori și de parametrii de ieșire.

- Example:
 - Termostatarea unei incinte
 - Controlul vitezei de rotație a unui hard-disk
 - Controlul altitudinii de zbor
 - Cruise control/Traction control
 - Surse de alimentare
 - Controlul mișcării pentru roboții industriali

- Aplicarea unor valori la intrarea sistemului pentru care ieșirea acestuia se conformează unei valori de referință.
 - Cruise-control: $f_{\text{motor}}(t)=?$ → viteza=60 km/h
 - Server E-commerce: Alocarea resurselor? → $T_{\text{răspuns}} = 5 \text{ sec}$
 - Networking: Rata de transfer? → Întârziere = 1 sec

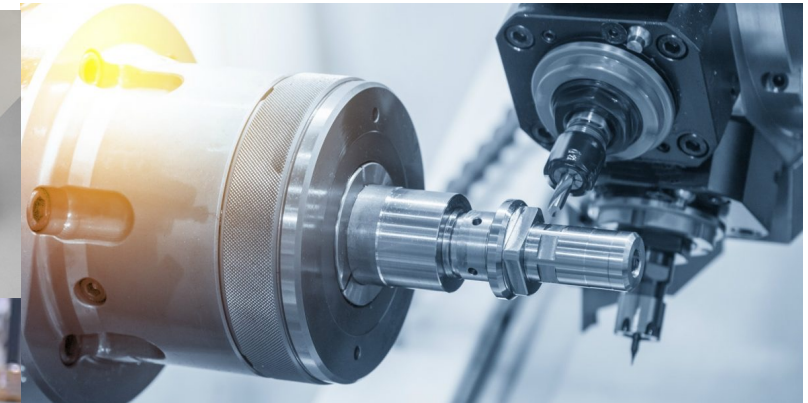
- Rădăcinile în altă știință: Cibernetica
- Disciplina care studiază comunicațiile și controlul între diferitele subsisteme constituind un organism viu sau o mașină construită de om. Feedback-ul este principala sa componentă.



- Norbert Wiener, 1948

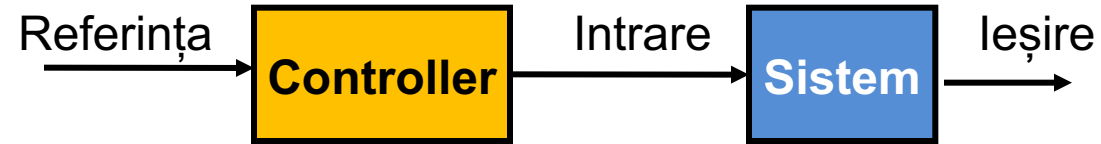
Ce este teoria controlului?

- Sistem = Un obiect sau ansamblu care se schimbă cu timpul
- Control = Modalitate de a influența schimbarea respectivă
- Exemple:
 - **Roboți**
 - Epidemii
 - Bursa de valori
 - Ordinea socială
 - Reglarea temperaturii
 - Circuite
 - Motoare
 - Rețele electrice
 - Sisteme autopilot

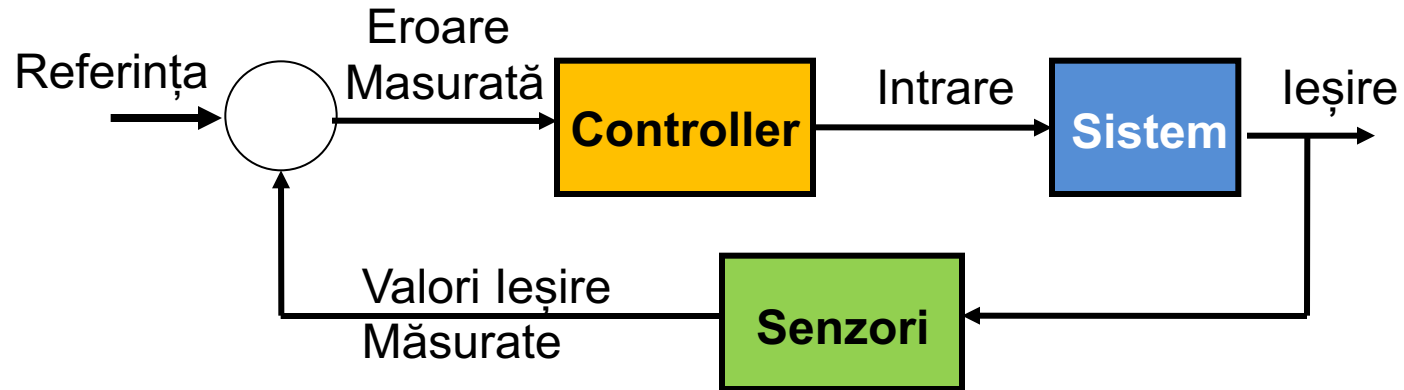


Tipuri de sisteme de control

- Sisteme în buclă deschisă

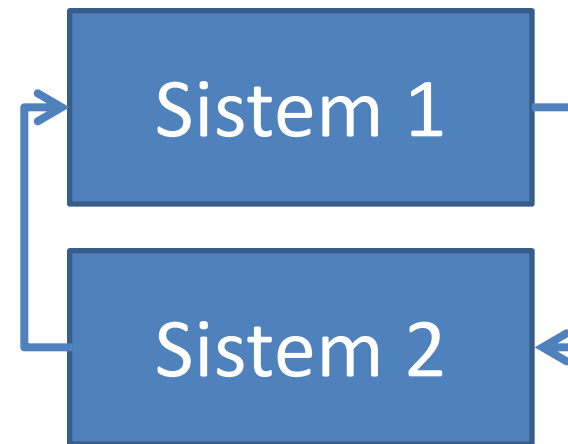


- Sisteme cu reacție



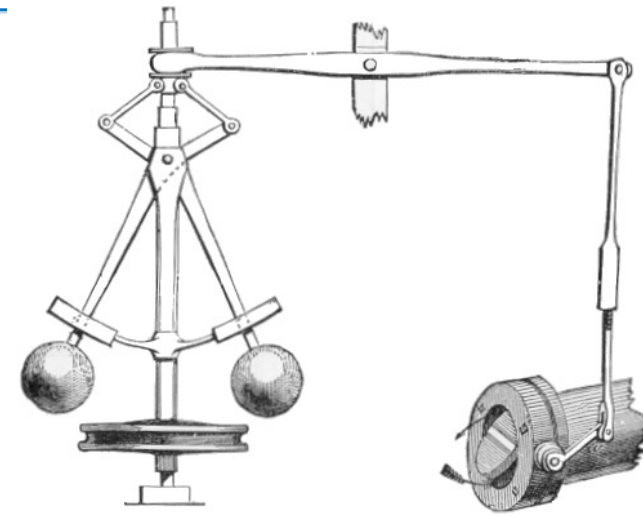
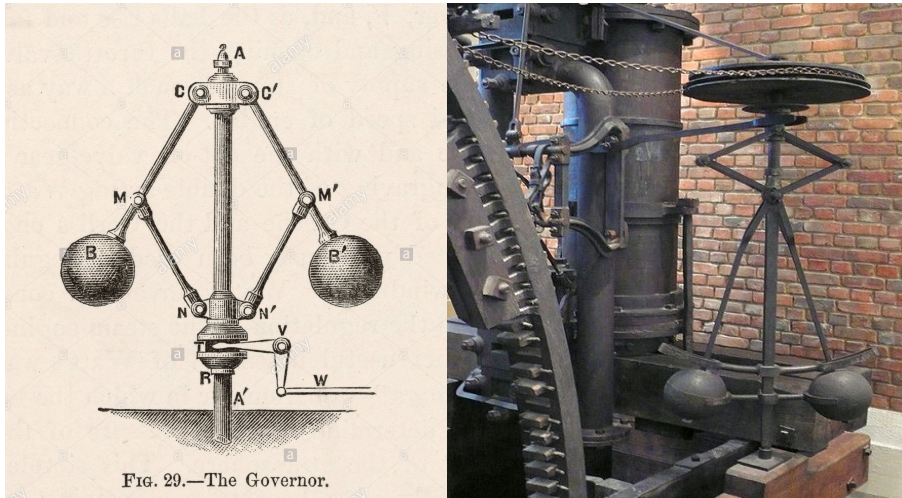
- Calculează valorile de intrare fără a măsura variabilele de sistem
 - Simplu de implementat
 - Trebuie cunoscute exact **TOATE VARIABILELE** din sistem ca totul să meargă cum trebuie
 - Cruise-control: $frecare(t)$, $unghi_plan(t)$
 - Server E-commerce: Încărcarea (rata de sosire a cererilor? Consumul de resurse?); sistem (timp de service? Defecțiuni?)
- Sistemele în buclă deschisă dau greș atunci când
 - Nu știm totul
 - Facem erori de modelare
 - Lucrurile se schimbă

- Ce este reacția?
 - Întoarcerea unei părți din ieșirea unui sistem la intrarea acestuia în scopul auto-corectării.
 - Presupune interconectarea mutuală a două sau mai multe sisteme
 - Relația cauză – efect e dificil de stabilit. Sisteme interdependente
 - Feedback-ul este prezent oriunde în sistemele naturale și artificiale



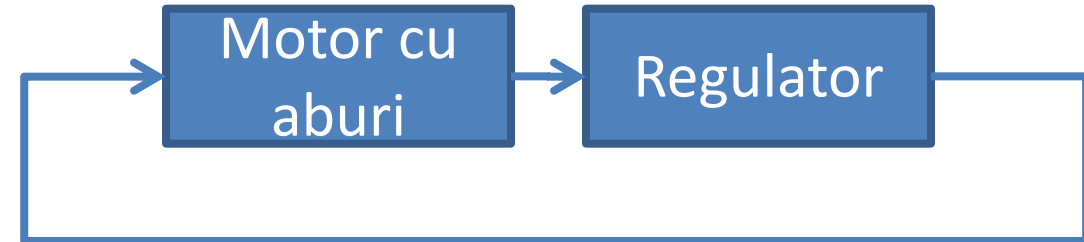
Exemplu #1 – Regulator de viteză

- “Flyball governor” (1788)
 - Regulează viteza unui motor cu aburi
 - Reduce efectele variației de sarcină (rejecția perturbațiilor)
 - Produce accelerarea revoluției industriale



Greutățile se îndepărtează
odată cu creșterea vitezei
de rotație

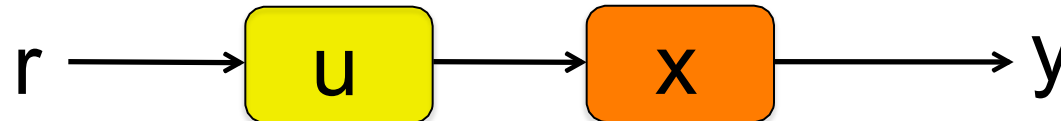
Supapa se închide,
micșorând turația motorului



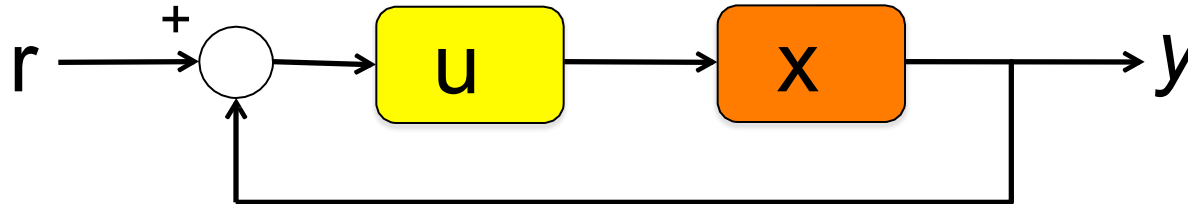
- **Stare** = Reprezentare a sistemului la un moment dat de timp
- **Dinamică** = Descriere a cum se schimbă starea sistemului
- **Referință** = Ce vrem ca sistemul să facă
- **Ieșire** = Măsură a (anumitor aspecte ale) sistemului



- **Stare** = Reprezentare a sistemului la un moment dat de timp
- **Dinamică** = Descriere a cum se schimbă starea sistemului
- **Referință** = Ce vrem ca sistemul să facă
- **Ieșire** = Măsură a (anumitor aspecte ale) sistemului
- **Intrare** = Semnal de control



- **Stare** = Reprezentare a sistemului la un moment dat de timp
- **Dinamică** = Descriere a cum se schimbă starea sistemului
- **Referință** = Ce vrem ca sistemul să facă
- **Ieșire** = Măsură a (anumitor aspecte ale) sistemului
- **Intrare** = Semnal de control
- **Feedback** = Mapare a ieșirii la intrare



- **Roboți**
- Epidemii
- Bursa de valori
- Termostate
- Circuite
- Motoare
- Rețele electrice
- Sisteme autopilot

Control = Senzori + Calcule + Efectoare

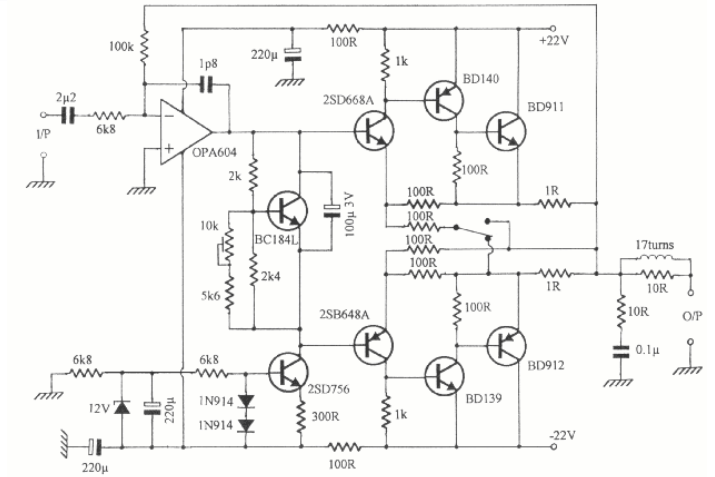


Țeluri:

1. Stabilitate: Sistemul își menține starea de-a lungul timpului (rulează cu viteză constantă)
2. Performanța: Sistemul reacționează rapid la schimbări (accelerează de la 0 la 100 km/h)
3. Robustețe: Sistemul tolerează perturbațiile (masă, frecare, unghiul pantei etc.)

Cele două principii de control

- **Robustețe la nesiguranță prin feedback**
 - Reacția asigură performanțe mărite chiar și la variațiile neprevizibile ale variabilelor de sistem
 - Amplificatoare care funcționează corect chiar dacă valorile componentelor variază
 - Ideea de bază: măsurarea cu acuratețe a diferenței dintre comportamentul obținut și cel dorit; corectare prin calcul și efectori.
- **Designul comportamentului dinamic prin feedback**
 - Reacția permite modificarea caracteristicilor dinamice ale unui sistem
 - Exemplu: Îmbunătățirea manevrabilității pentru avioanele instabile
 - Ideea de bază: Interdependența modifică comportamentul normal

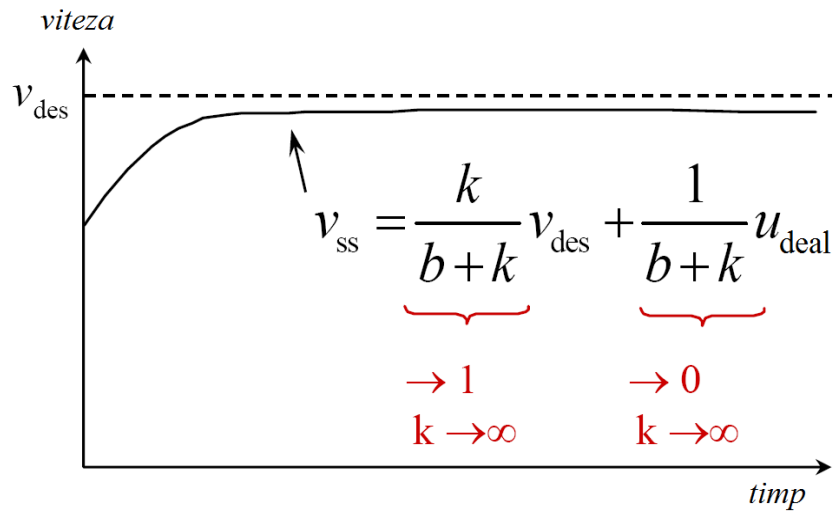


Exemplul #2 – Cruise Control



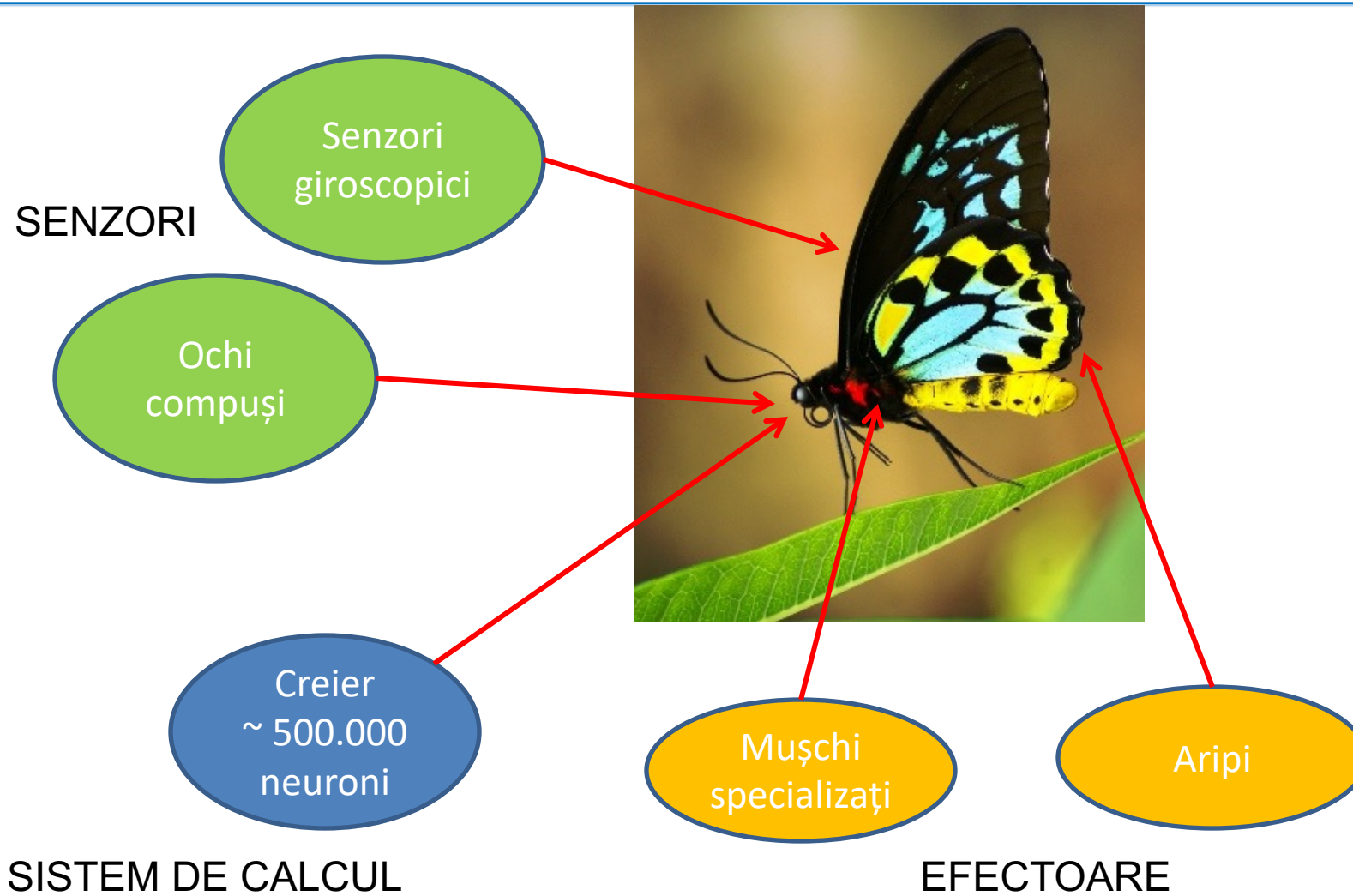
$$m\dot{v} = -bv + u_{motor} + u_{drum}$$

$$u_{motor} = k(v_{des} - v)$$



- Stabilitate/performanță
 - Viteza steady state (V_{ss}) se apropie de viteza dorită pentru $k \rightarrow \infty$
 - Răspuns lin, fără depășire sau oscilații
- Rejectia perturbațiilor
 - Efectele perturbațiilor (dealurile) sunt eliminate când $k \rightarrow \infty$
- Robustețe
 - Rezultatele nu depind de valorile factorilor m, b, k pentru k suficient de mare.

Exemplul #3 – Zborul unei insecte



- Sisteme de zbor
 - Fly-by-wire
 - UAV
- Robotică
 - Determinarea exactă a poziției pentru operații precise
 - Medii greu accesibile: spațiu, mare, operații non-invazive
- Procese chimice
 - Reglarea temperaturii, vitezei de reacție, dozarea reactanților
- Comunicații și rețelistică
 - Amplificatoare și repetoare de semnal
 - Power management pentru comunicațiile wireless
- Automobile
 - Controlul motorului, tracțiunii, climatizării, stabilității etc.

Și multe altele....

- Teoria controlului = Cum să găsim semnalul de intrare u ?
- Obiective:
 - Stabilitate
 - Tracking
 - Robustețe
 - Respingerea perturbațiilor
 - Optimalitate

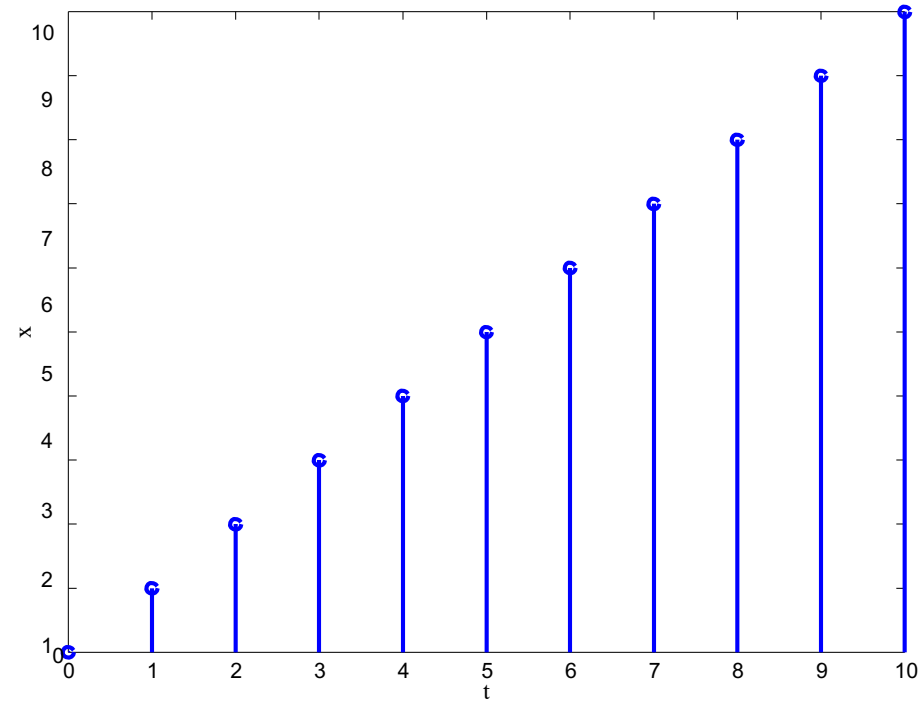
- Strategiile eficiente de control se bazează pe modele predictive
- Timp discret:

$$x_{k+1} = f(x_k, u_k) \quad \longleftarrow \text{Ecuație a diferențelor}$$

Exemplu: Ceas

$$x_{k+1} = x_k + 1$$

Ceas de timp discret



Dinamică = schimbare cu timpul

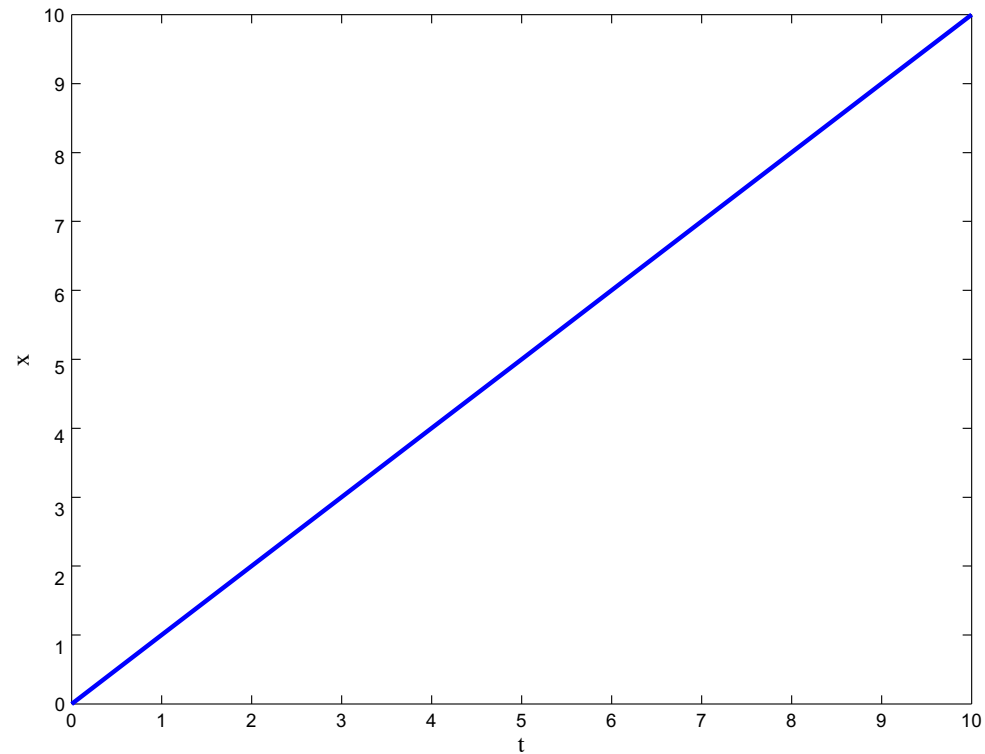
- Legile fizicii (și Natura) sunt scrise pentru timp continuu: În loc de starea următoare, avem nevoie de variația în raport cu timpul a mărimii
- Timp continuu:

$$\frac{dx}{dt} = f(x, u) \quad \sim \quad x' = f(x, u) \quad \longleftarrow \quad \text{Ecuație diferențială}$$

- Ceas:

$$\dot{x} = 1$$

Ceas de timp continuu



- Dar, într-o implementare totul este discret/eșantionat!

$$\dot{x} = f(x, u)$$

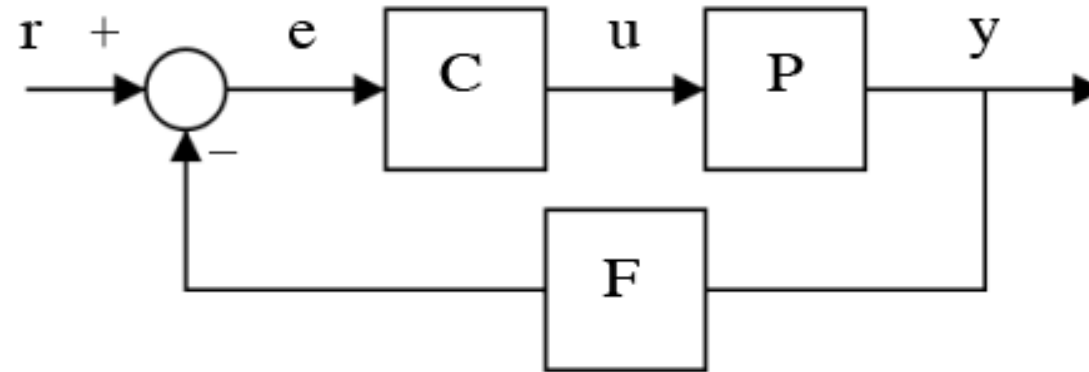
- Timpul de eșantionare: δt

$$x_k = x(k\delta t) \quad \Rightarrow \quad x_{k+1} = x((k+1)\delta t) = ???$$

$$x(k\delta t + \delta t) \approx x(k\delta t) + \delta t x'(k\delta t)$$

$$x_{k+1} = x_k + \delta t f(x_k, u_k)$$

Funcția de transfer



$$Y(s) = P(s)U(s) \quad U(s) = C(s)E(s) \quad E(s) = R(s) - F(s)Y(s).$$

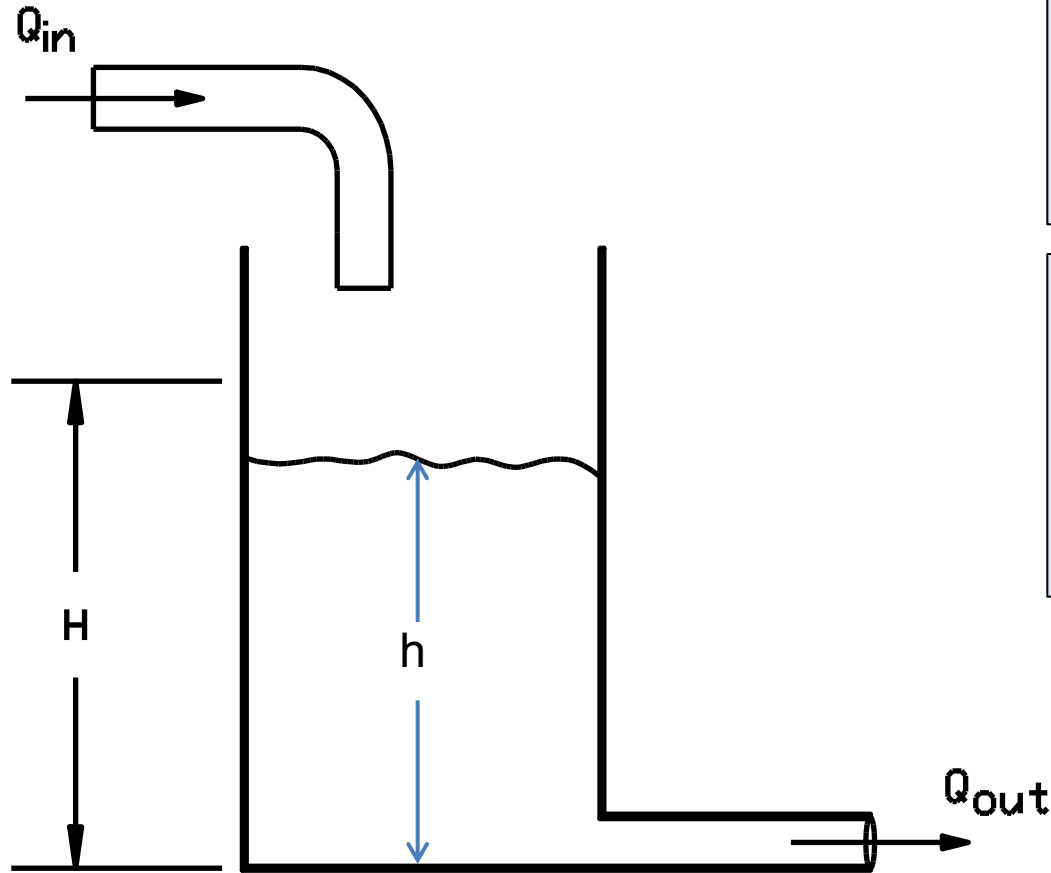

$$Y(s) = \left(\frac{P(s)C(s)}{1 + F(s)P(s)C(s)} \right) R(s) = H(s)R(s).$$

$$H(s) = \frac{P(s)C(s)}{1 + F(s)P(s)C(s)}$$

Funcția de transfer a sistemului

Problema controlului

Concepte fundamentale



Menținem o variabilă a unui process la o valoare dorită și rejectăm efectele perturbațiilor prin manipularea alte variabile din sistem.

Exemple:

Încălzirea sau răcirea unei incinte
Cruise control la automobile
Menținerea altitudinii de zbor a unei drone
Menținerea constantă a nivelului unui rezervor

Q_{out} depinde de h

Dacă $Q_{out} = Q_{in}$, h constant

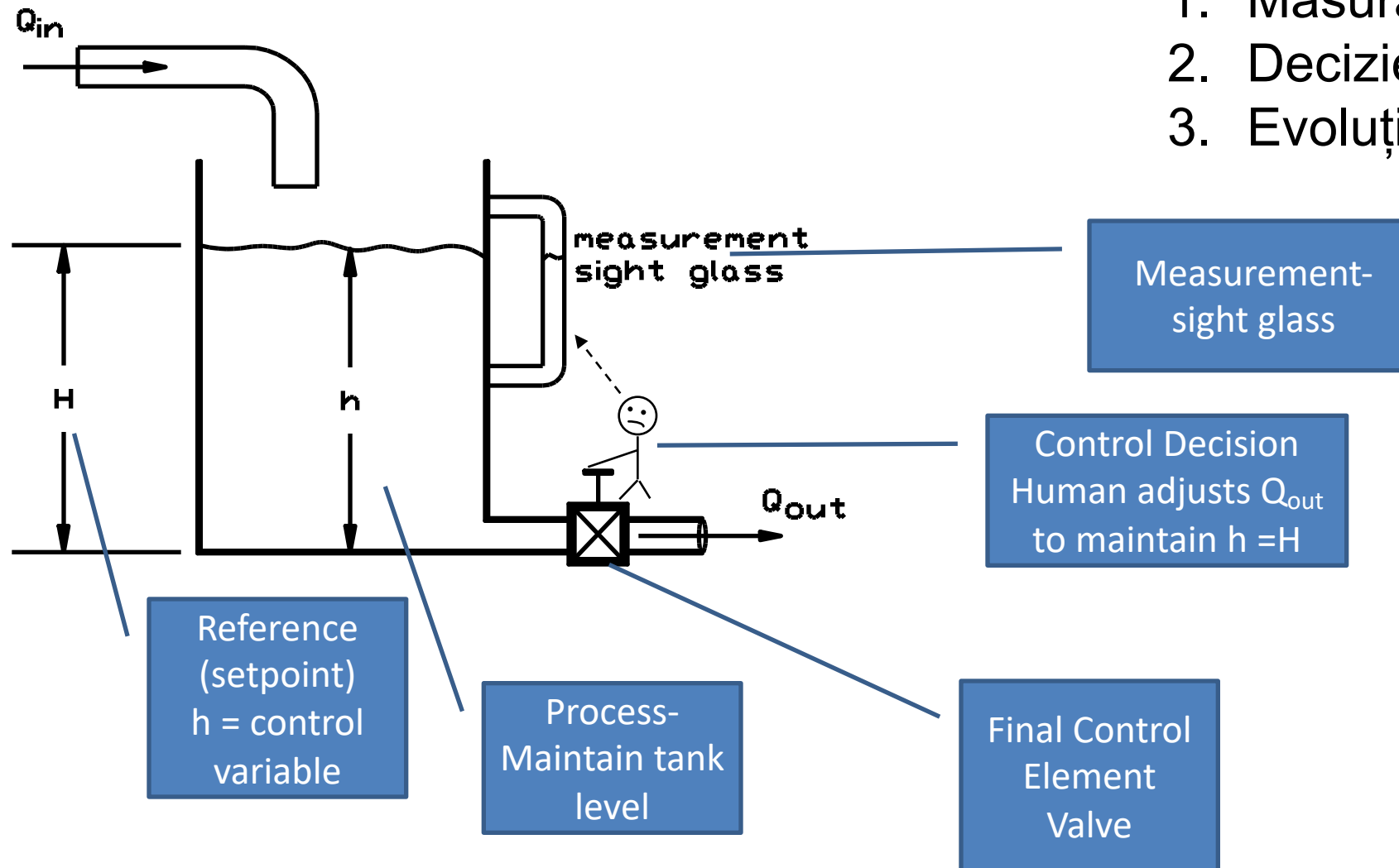
$Q_{out} > Q_{in}$, rezervorul se golește

$Q_{out} < Q_{in}$, rezervorul refulează

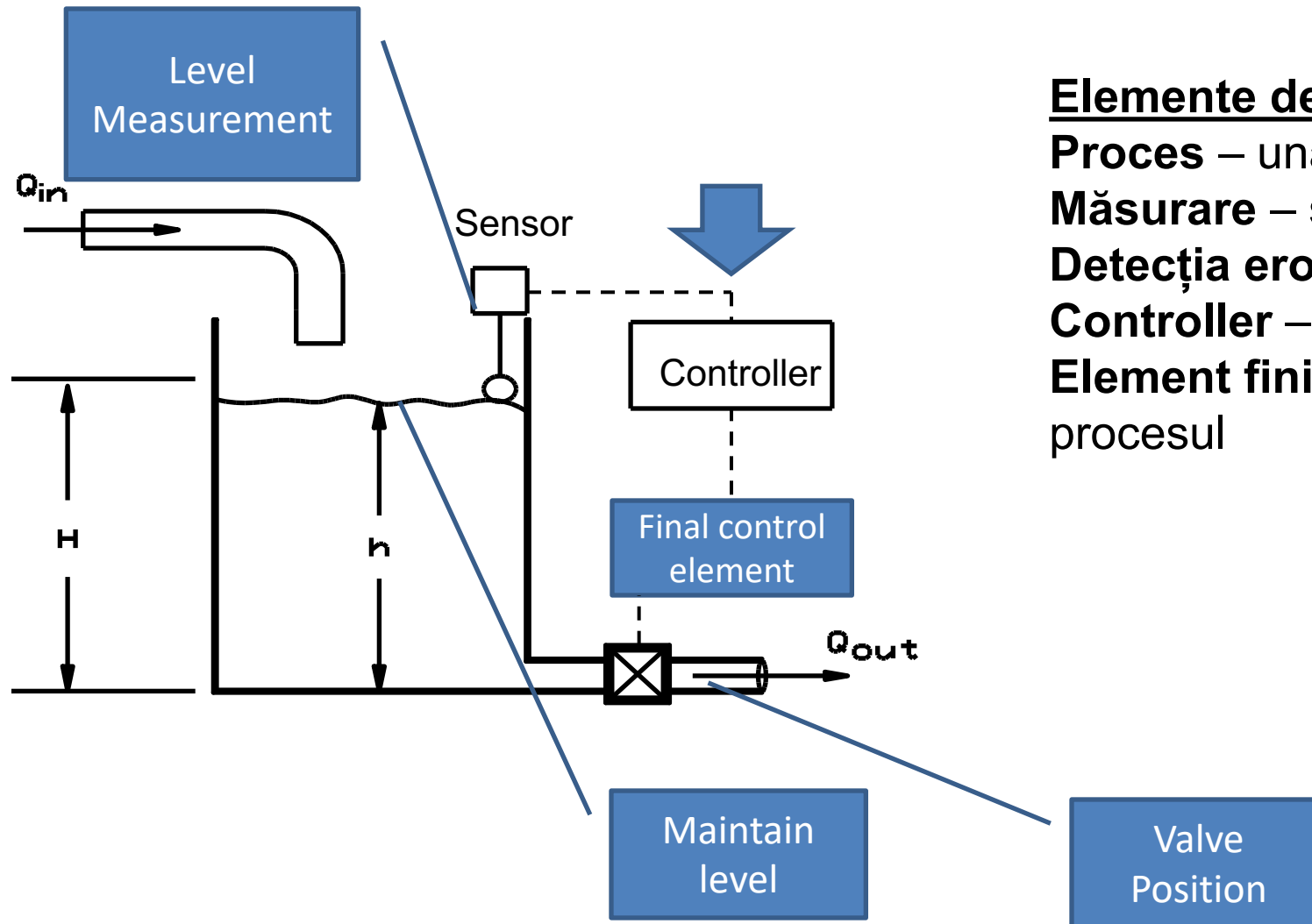
Teoria controlului: idei de bază

Sisteme cu reacție

1. Măsurare
2. Decizie/lege de control
3. Evoluția sistemului



Folosim senzori și electronica pentru a monitoriza și controla sistemul



Elemente de control automat

Proces – una sau mai multe variabile

Măsurare – senzori

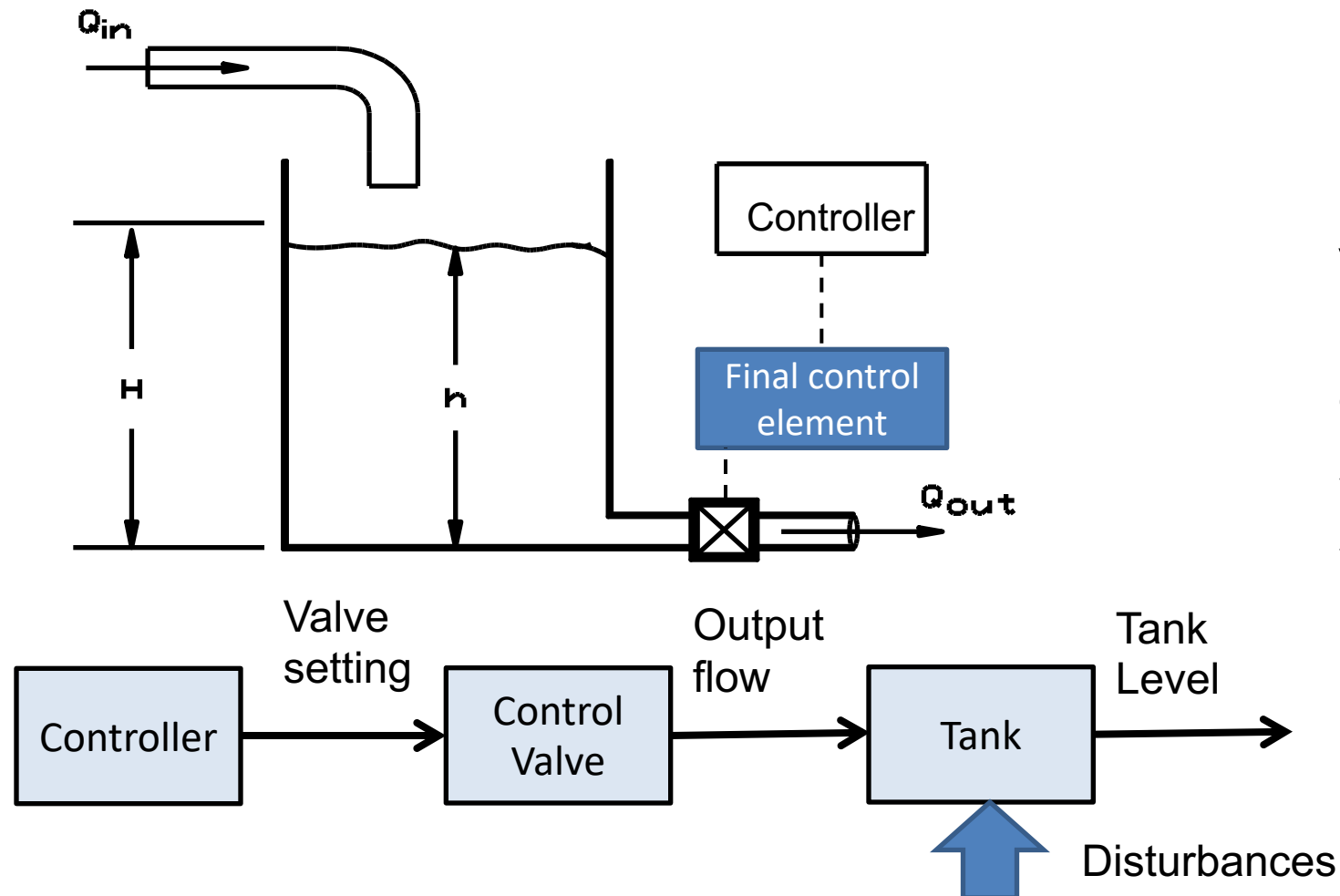
Detectia erorii – compară H cu h

Controller – aplică corecții

Element finit de control – modifică procesul

Sistem în buclă deschisă

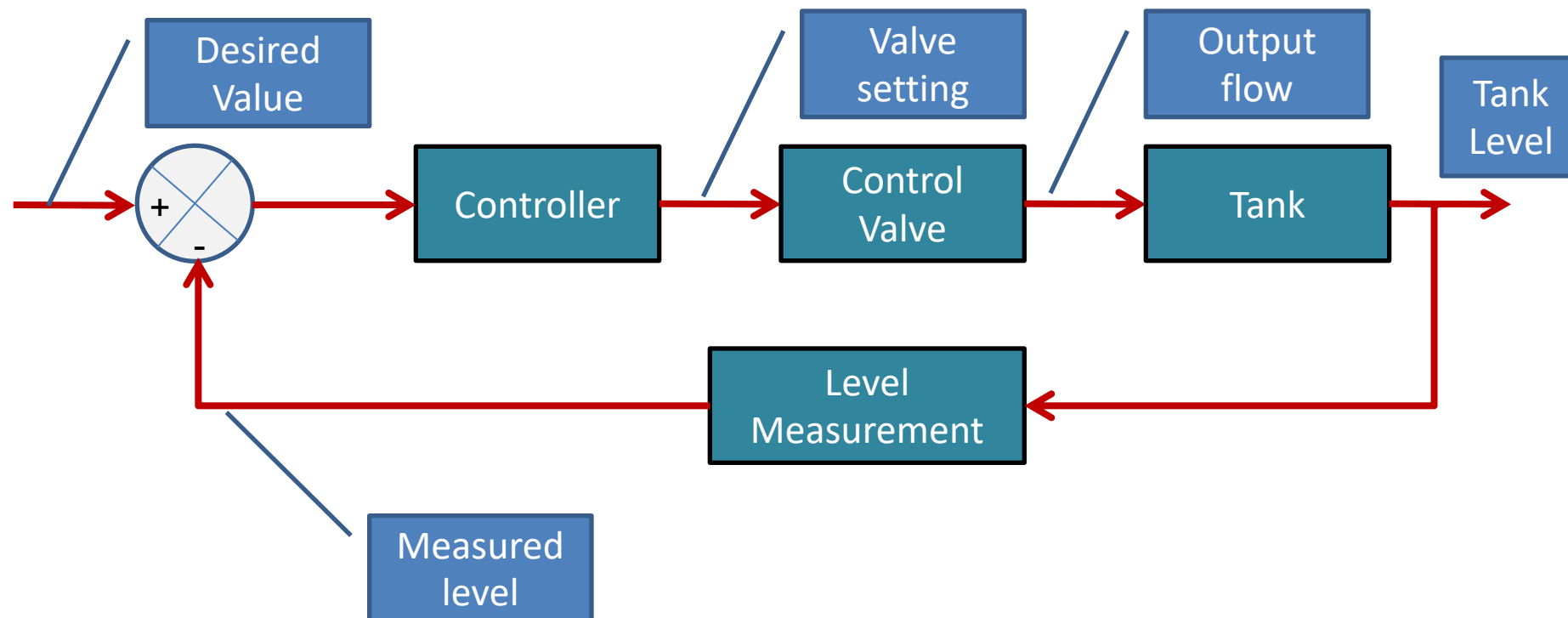
Sistemul în buclă deschisă modifică ieșirea în baza unor valori predeterminate de control. Nu se măsoară cantitatea controlată.



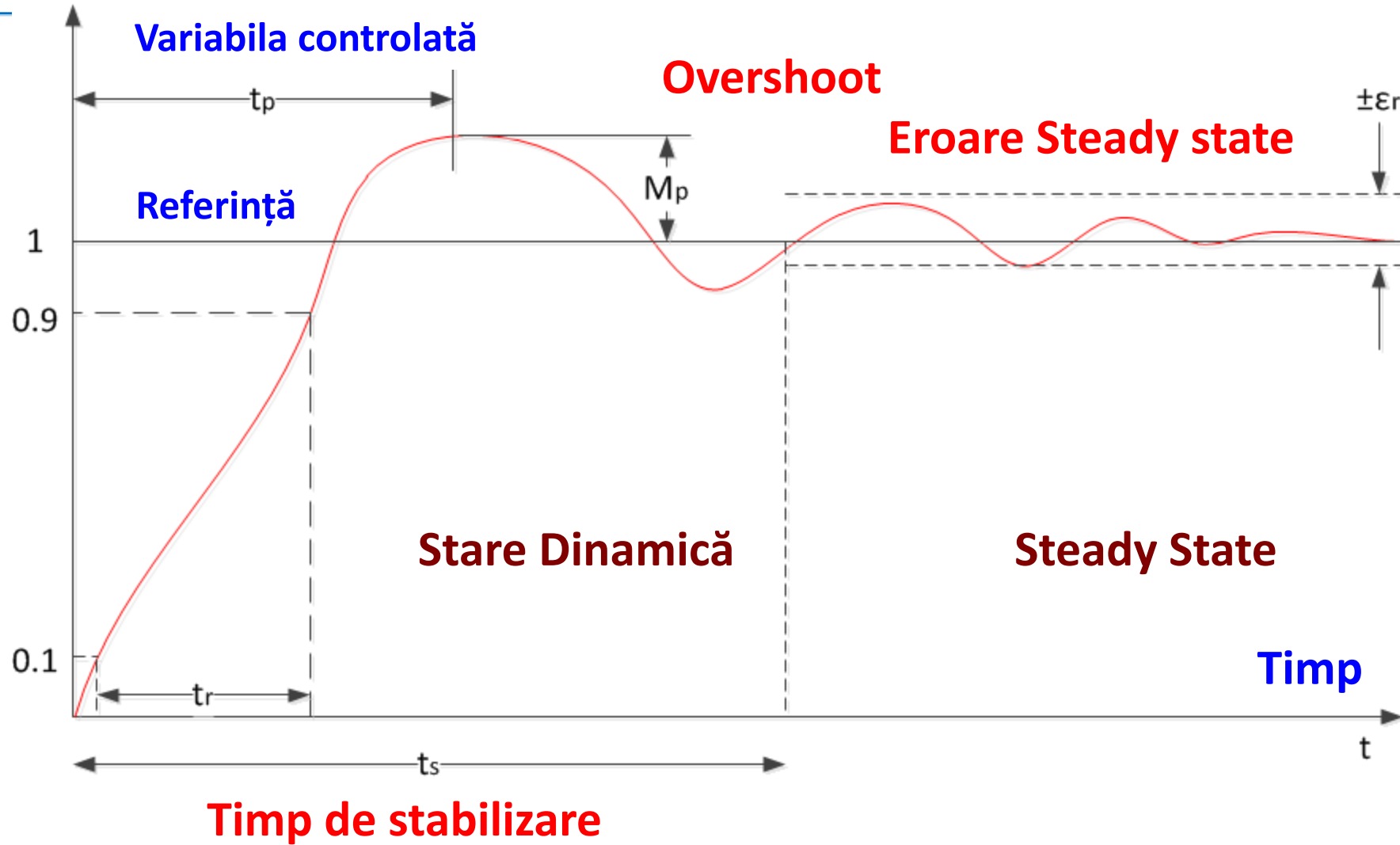
Nivelul rezervorului poate să varieze la perturbații externe sau la schimbări de sistem

Sistem cu reacție

Reacția presupune o modificare a ieșirii în funcție de valorile măsurate ale variabilei de control. Valoarea măsurată este comparată cu valoarea dorită și ajustată la apariția perturbațiilor. Sistemul cu reacție aduce o parte din ieșire înapoi la intrare.



Indicatori de Performanță



- Stabilitate
- ieșirea “urmărește” intrarea pentru orice tip de semnal de intrare
- Ex: în controlul proporțional, stabilitatea este estimată prin stabilirea dacă polii funcției de transfer au modulul mai mic ca 1

- Acuratețe
- Acuratețea este dată de mărimea erorii de steady-state.
- Ex: în controlul proporțional, acuratețea este calculată în raport cu câștigul funcției de transfer pentru o valoare de referință a intrării. Eroarea este zero dacă și numai dacă acest câștig este 1.

- Depășirea superioară (Overshoot)
- Este o proprietate a răspunsului la o excitație de tip treaptă a intrării.
- Depășirea superioară este caracteristică pentru un comportament oscilant al sistemului. O valoare mare pentru overshoot va fi urmată de obicei de o depășire inferioară (undershoot).

Tipuri de control

- Control Proporțional (P)
- Control Proporțional Integral (PI)
- Control Proporțional Diferențial (PD)
- Control PID

- Face o mașină să meargă la o viteză dorită, de referință (r)

- A doua lege a lui Newton:

$$F = ma$$

- Stare: viteza (x)

- Intrare:
accelerează/frână (u)

$$F = cu \quad (c = \text{coeficient transmisie electro-mecanică})$$

- Dinamică:

$$x' = a \Rightarrow mx' = cu \Rightarrow \underline{\underline{x' = \frac{c}{m} u}}$$

- Presupunem că măsurăm viteza ($y=x$)
- Sistemul de control trebuie să fie o funcție de $r-y$ ($=e$)
- Ce proprietăți trebuie să aibă semnalul de control?
 - Un e mic va genera un u mic
 - u nu ar trebui să fie “zgâlțâit”
 - u nu trebuie să depindă de valorile c și m (nu putem oricum să le cunoaștem în toate cazurile)

- Modelul mașinii:

$$\dot{x} = \frac{c}{m} u$$

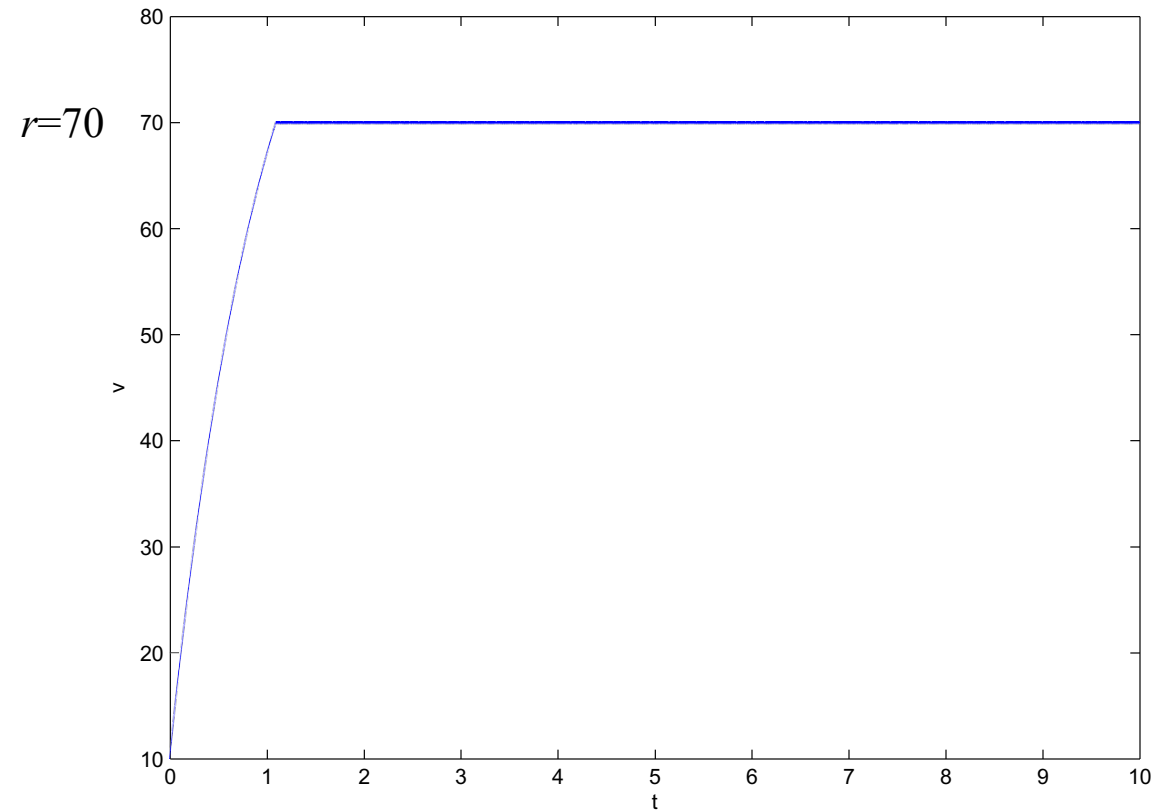
- Vrem să

$$x \rightarrow r \text{ când } t \rightarrow \infty \quad (e = r - x \rightarrow 0)$$

- Încercarea 1

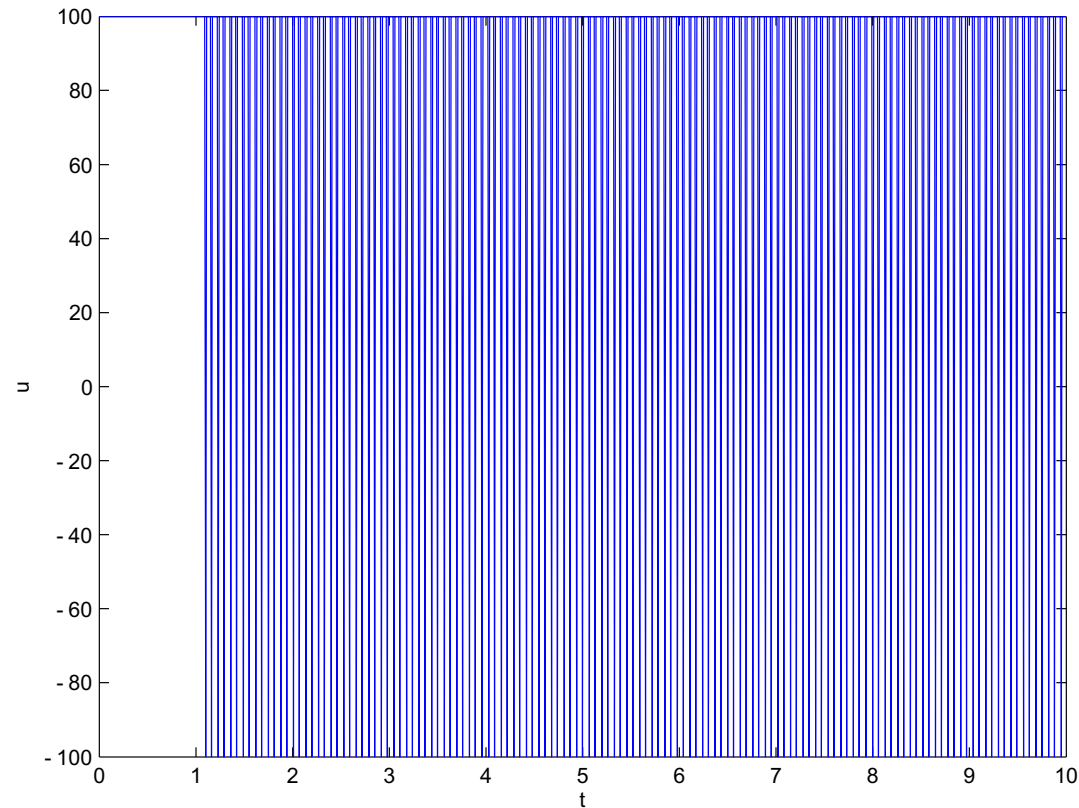
$$u = \begin{cases} u_{\max} & \text{if } e > 0 \\ -u_{\max} & \text{if } e < 0 \\ 0 & \text{if } e = 0 \end{cases}$$

Control Bipozițional (Bang-Bang) - Teoretic



Bang-Bang Control - Realitatea

BAD!



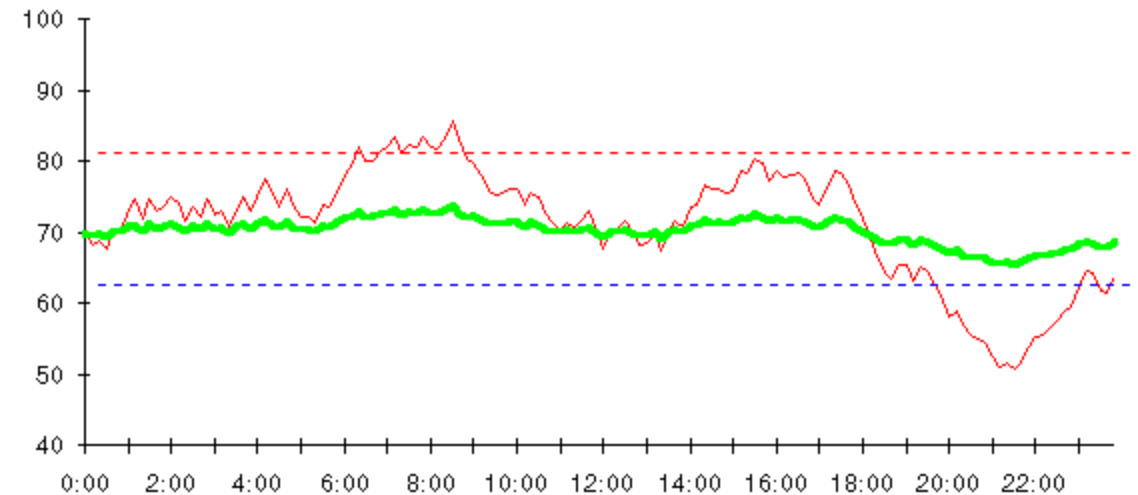
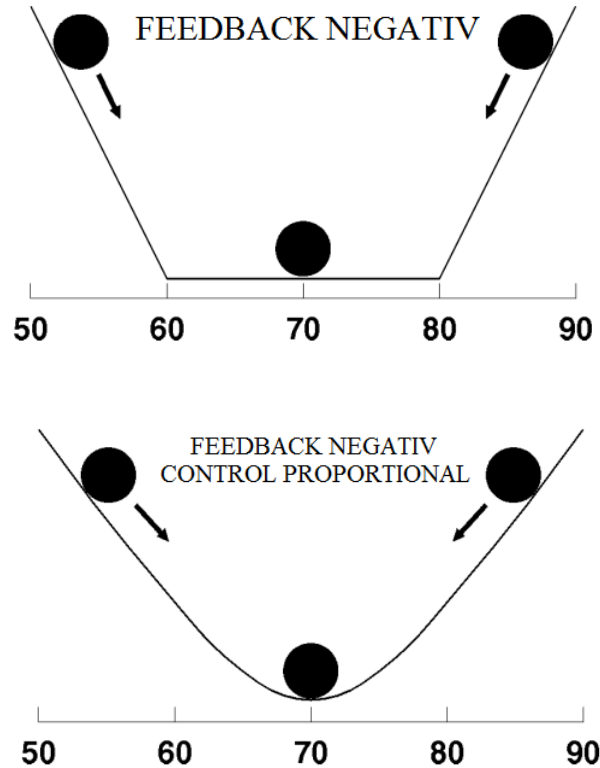
Bumpy ride
Burns out actuators
....

Bang-Bang Control – Proces lent

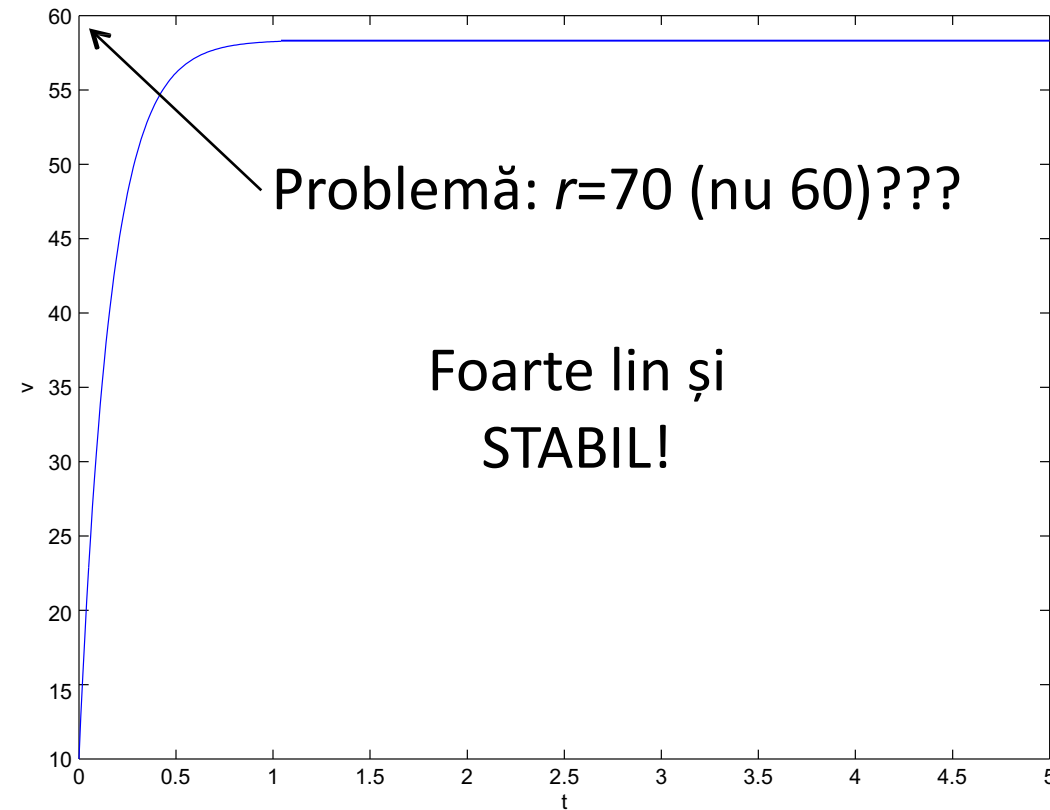


Putem avea un control mai precis?

- Controlul temperaturii unei incinte
 - Termostat cu 2 praguri: 14 si 20 grade C (**bang-bang**)
 - Regulator proporțional



- Problemă: Regulatorul reacționează prea puternic la erori mici
- *Încercarea 2*
 $u = ke$
 - Erorile mici dau un semnal de control mic
 - Control foarte "lin"
 - Numit și regulator proporțional (P-regulator)



- Deficiență: Modelul “real” este îmbunătățit pentru a include rezistența la înaintare:

$$\dot{x} = \frac{c}{m}u - \gamma x$$

- Acest model este folosit pentru a simula controller-ul
- La steady-state (x nu se mai schimbă)

$$\dot{x} = 0 = \frac{c}{m}u - \gamma x = \frac{c}{m}k(r - x) - \gamma x \Rightarrow (ck + m\gamma)x = ckr$$

$$x = \frac{ck}{ck + m\gamma}r < r$$

- Un regulator ar trebui să ofere:

- Stabilitate (BIBO)
- Tracking
- Robustețe

$$\dot{x} = \frac{c}{m} u - \gamma x$$

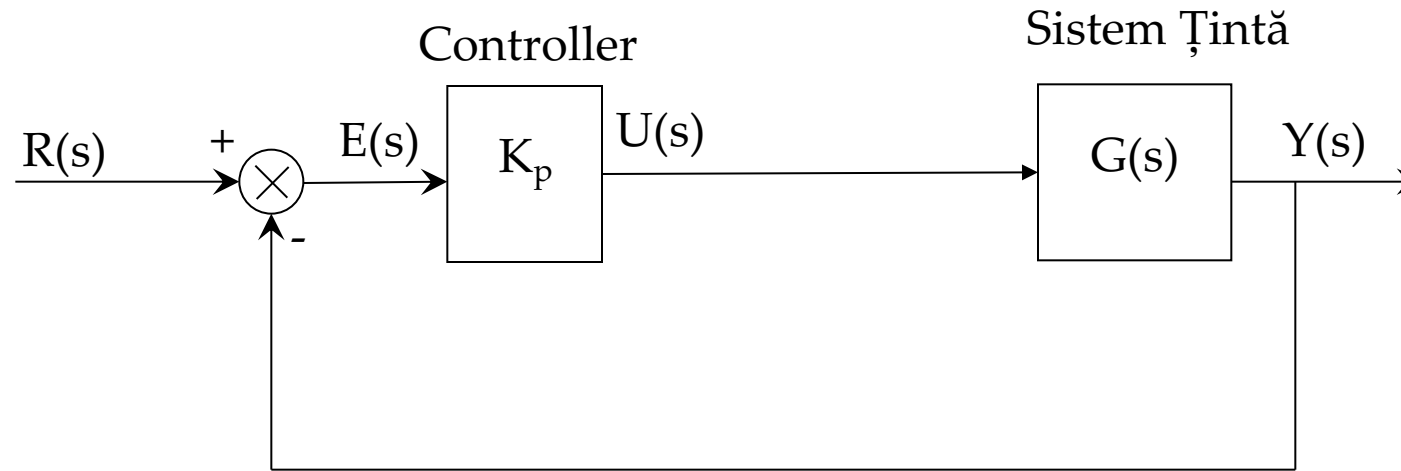
Problemă: Dintr-odată
trebuie să cunoaștem
toți parametrii fizici pe
care nu îi cunoaștem –
Nu este robust!

- Încercarea 3

$$u = ke + \gamma \frac{m}{c} x$$

$$\dot{x} = 0 = \frac{c}{m} k(r - x) + \gamma x - \gamma x \quad \text{Tracking!} \Rightarrow x = r$$

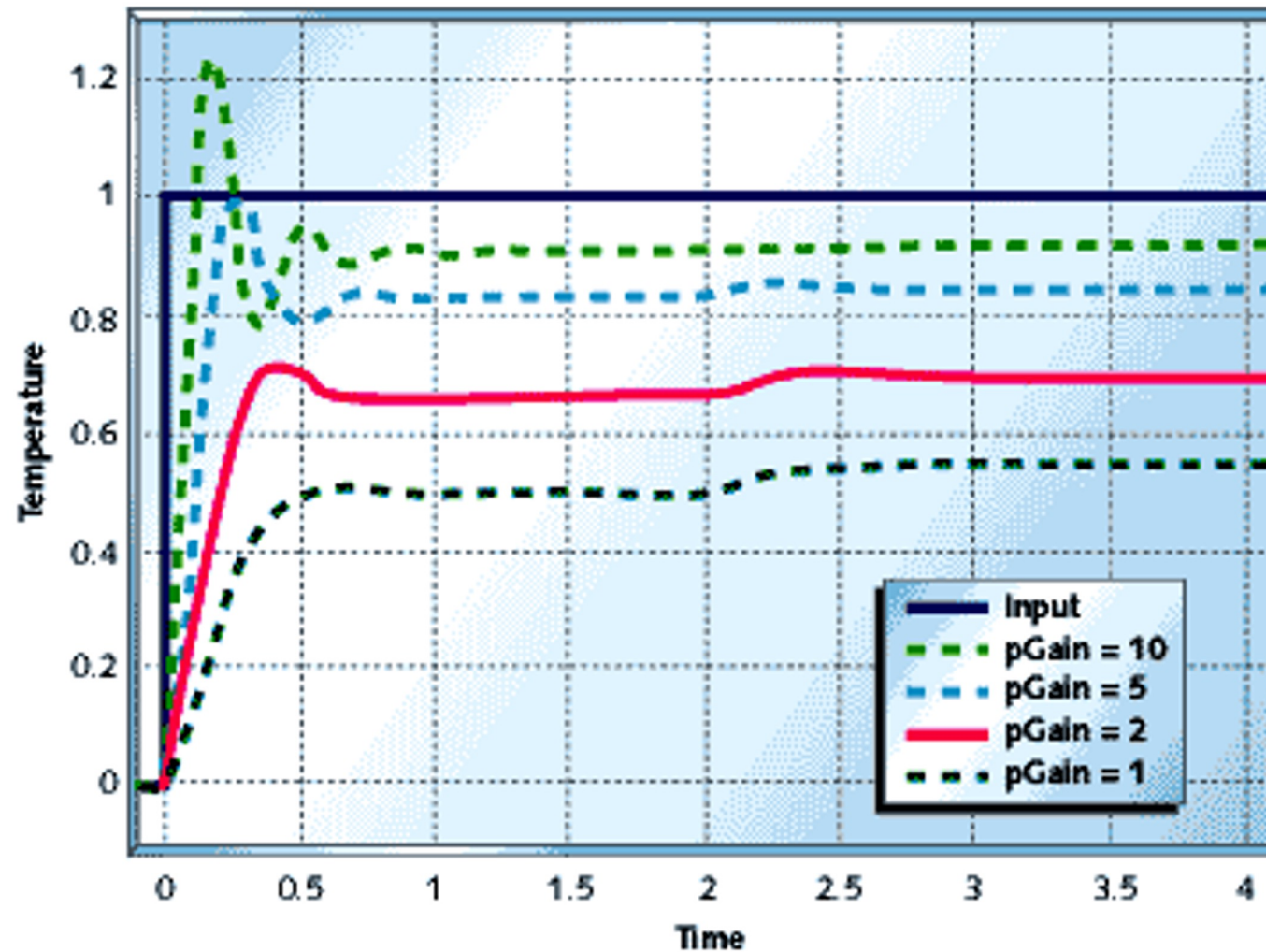
- Una dintre cele mai des întâlnite bucle de control
- Ieșirea controllerului este direct proporțională cu eroarea
- Produce rezultate bune pentru majoritatea aplicațiilor unde parametrii de funcționare nu sunt critici (overshoot, stabilitate, eroare steady-state)



$$E(s) = R(s) - Y(s) \quad U(s) = K_p E(s) \quad Y(s) = G(s)U(s)$$

$$H(s) = \frac{K_p \cdot G(s)}{1 + K_p \cdot G(s)}$$

Răspunsul la semnal de tip treaptă



www.embedded.com

- Ecuația caracteristică a sistemului:

$$1 + K_p \cdot G(s) = 0$$

- Când $K_p \rightarrow 0$, soluțiile următoarei ecuații sunt polii lui $G(s)$:

$$\frac{1}{G(s)} + K_p = 0$$

- Când $K_p \rightarrow \infty$, soluțiile următoarei ecuații sunt zerourile lui $G(s)$:

$$\frac{1}{K_p} + G(s) = 0$$

- Sistemul cu control proporțional este stabil dacă pentru o anumită valoare a lui K_p , toți polii sunt în interiorul cercului unitate
(modul < 1).
- Orice sistem care are cel puțin un zero la infinit va deveni întotdeauna instabil pentru o valoare suficient de mare a lui K_p .

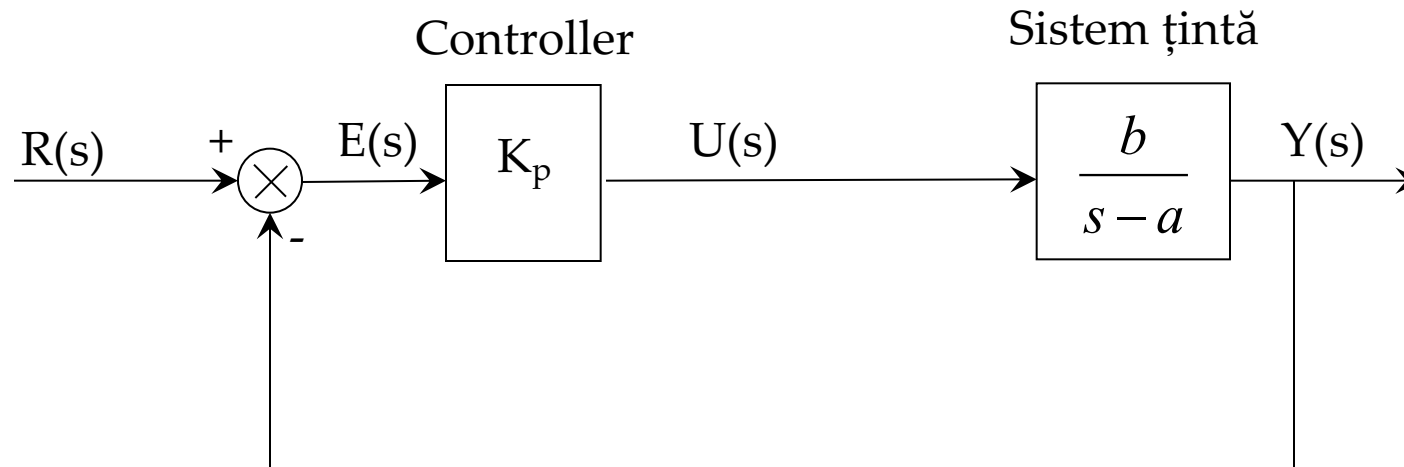
$$e_{ss} = \lim_{k \rightarrow \infty} (r(k) - y(k)) = r_{ss} - y_{ss}$$

$$K_p = \frac{H(s)}{G(s)(1 - H(s))}$$

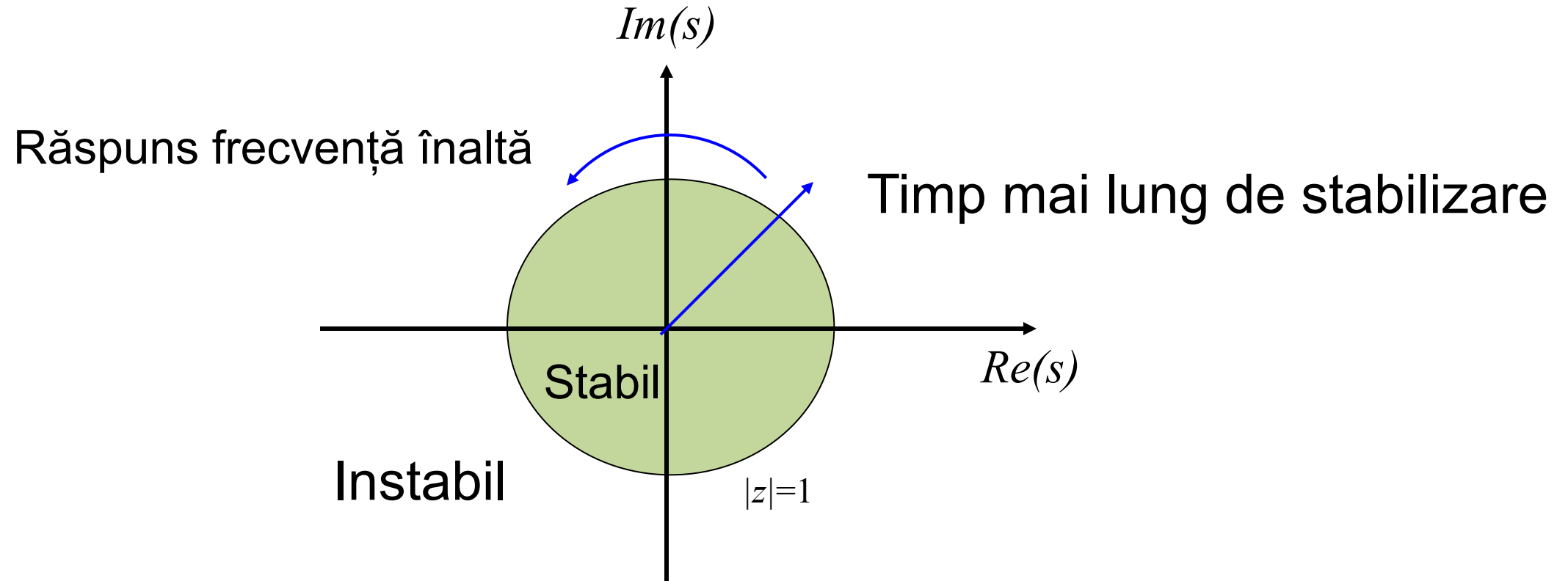
$$e_{ss} = \lim_{k \rightarrow \infty} e(k) = \lim_{s \rightarrow 0} \frac{1}{1 + K_p G(s)}$$

- Eroarea de stare stabilă este cu atât mai mică cu cât K_p este mai mare.

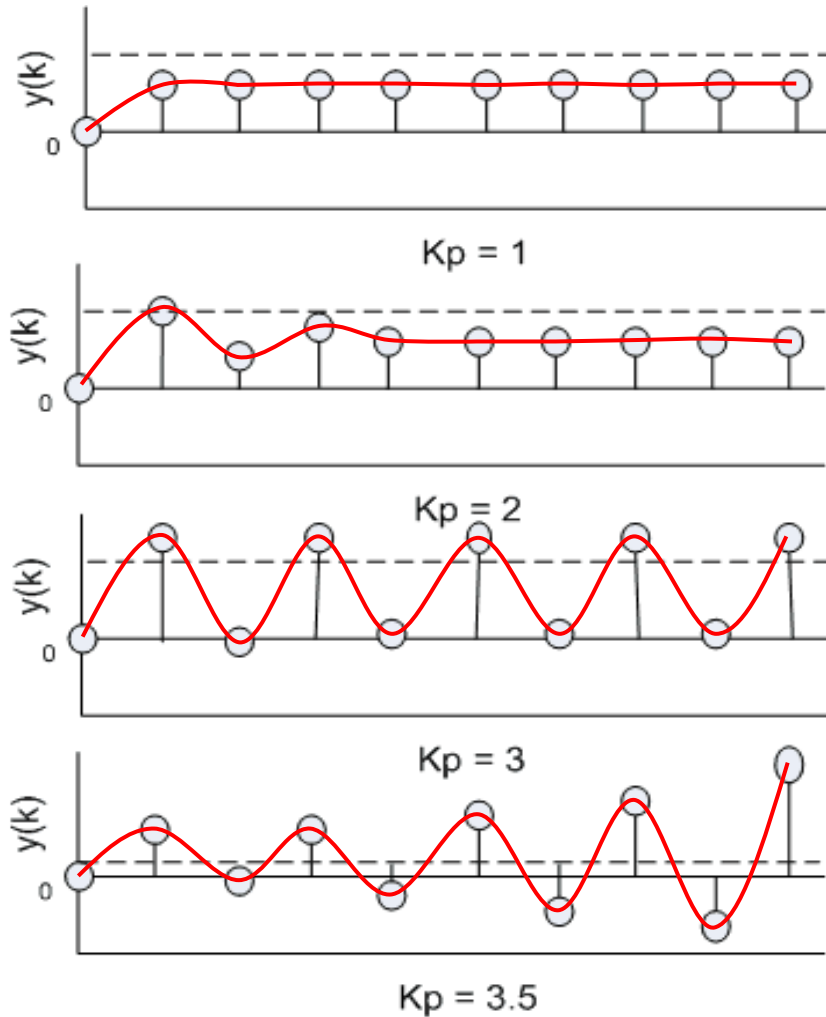
- Ce valoare a lui K_p aleg pentru ca sistemul meu sa fie cât mai performant?
- Exemplu: controllerul de temperatură



Efectul polilor funcției de transfer



Efecte pentru diferite valori ale K_p

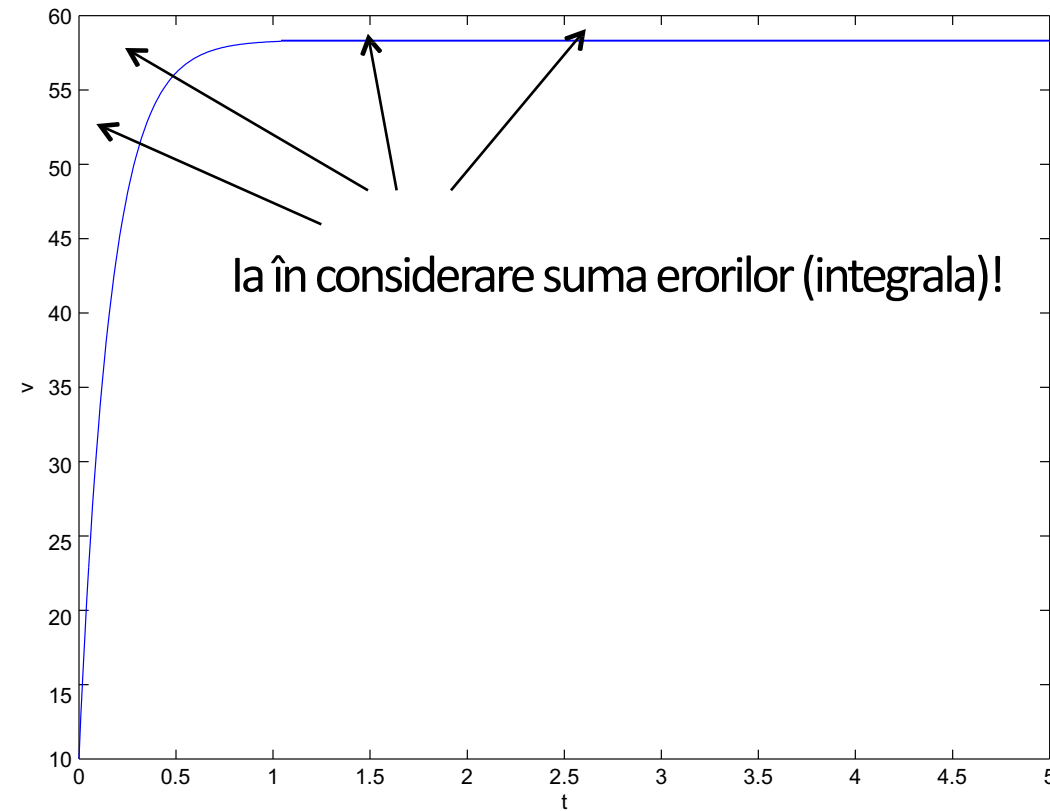


$$G(s) = \frac{0.43}{s - 0.47}$$

- Evoluția sistemului în timp în funcție de K_p
- $K_p = 1$, polul funcției de transfer este aproape 0, $|e_{ss}|$ este mare.
- $K_p = 2$, pol în -0.51, overshoot și oscilație.
- $K_p = 3$, pol la -1, oscilație.
- $K_p = 3.5$, pol la -1.25, sistemul este instabil.

- + Este foarte ușor de implementat
- + Stabilitate bună dacă alegem K_p potrivit
- Timp destul de îndelungat pentru stabilizare
- Poate intra foarte ușor în oscilație
- Pentru valori din ce în ce mai mari ale K_p sistemul devine instabil
- În general, una sau mai multe cerințe (eroare de stare stabilă, depășire, oscilație) sunt imposibil de satisfăcut pentru orice valoare a K_p

Regulator P



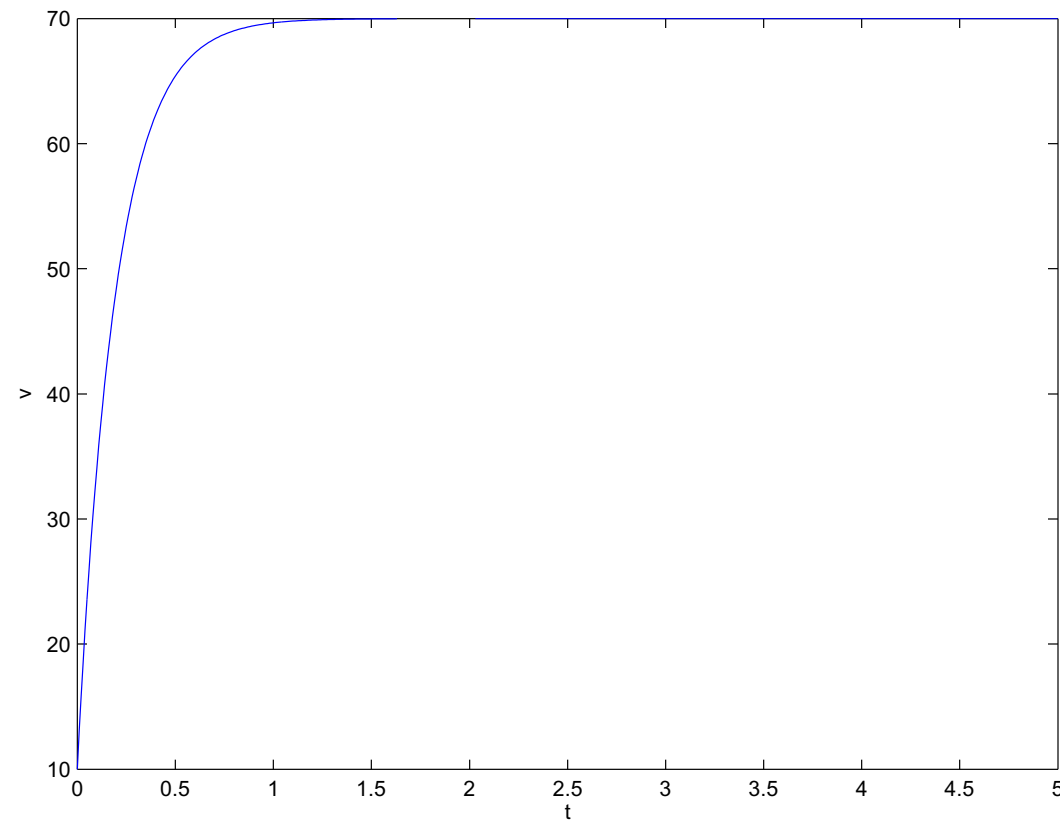
- Încercarea 4

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau$$

- Sau, de ce nu un Regulator PID?

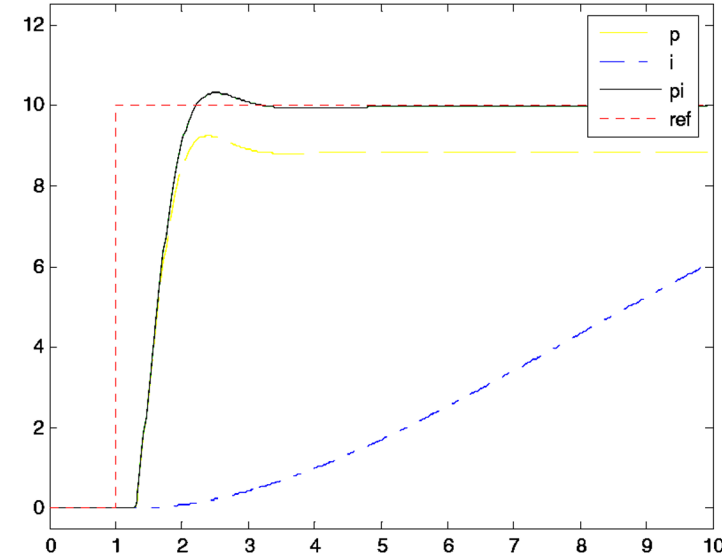
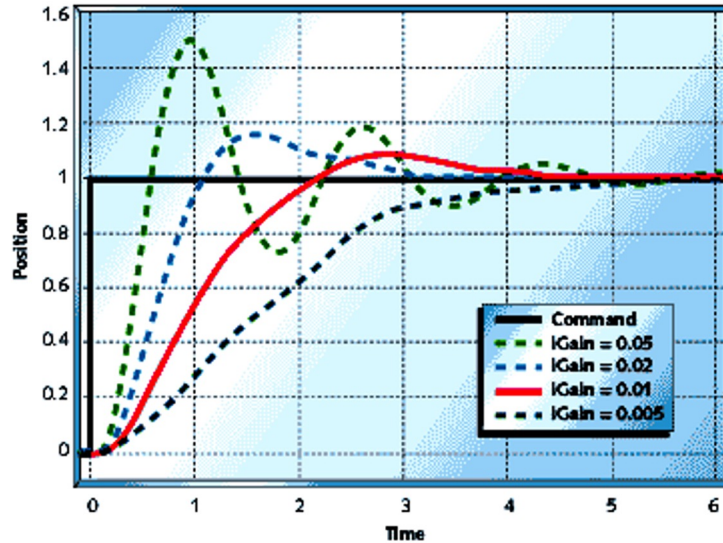
$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt}$$

Regulator PI



- Folosit pentru a adăuga precizie pe termen lung sistemului
- Aproape întotdeauna este folosit în conjuncție cu controlul proporțional (PI)
- Regulatorul însumează erorile trecute până când ieșirea ajunge la valoarea de referință

Control integral pur

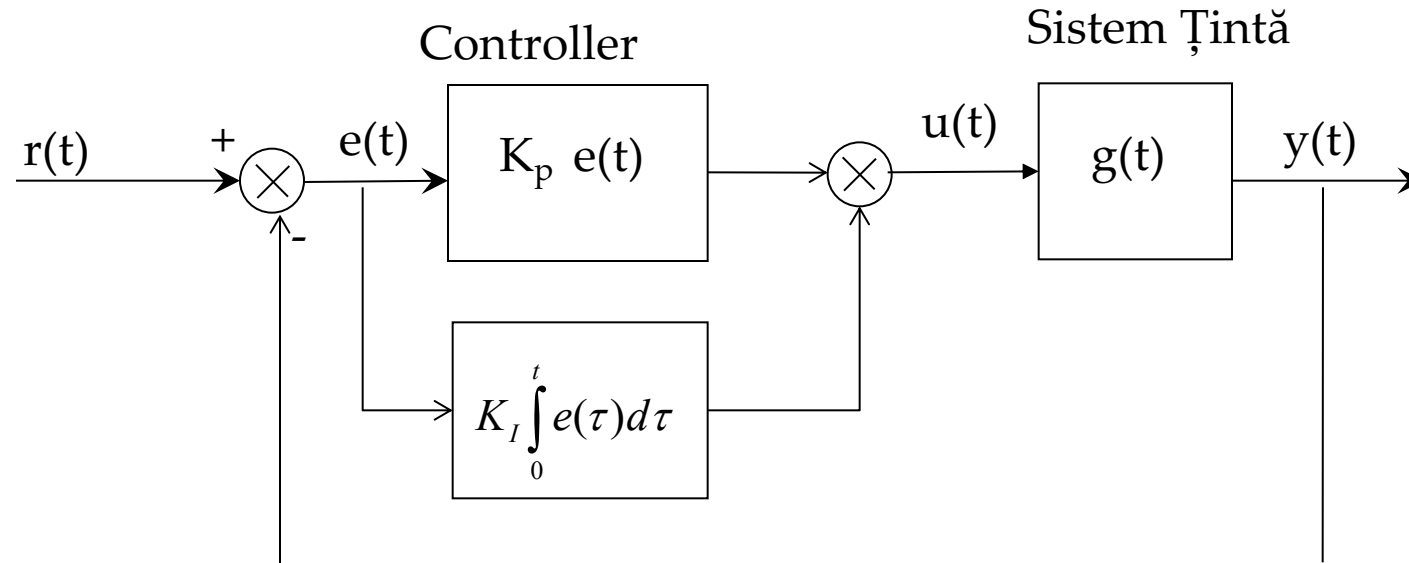


- Răspunsul controlului I este lent și uneori poate produce oscilația sistemului. De cele mai multe ori este preferat controlul PI.

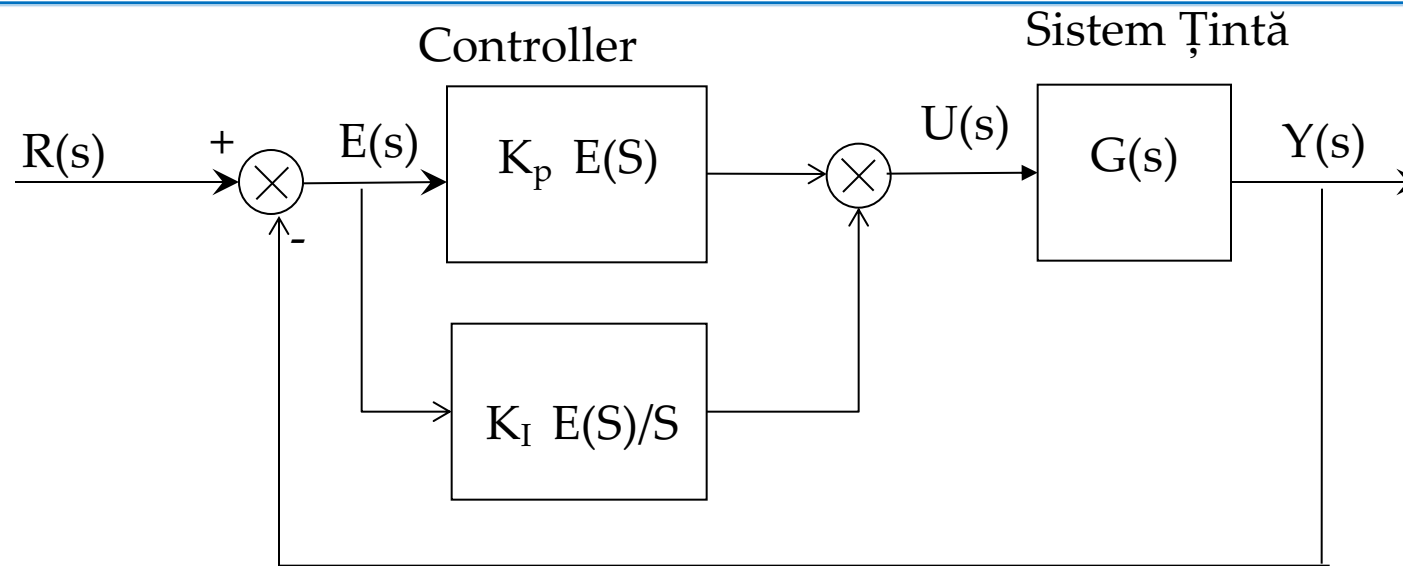
www.embedded.com

Control Proportional-Integral

- ieșirea regulatorului este direct proporțională cu integrala tuturor erorilor trecute.



Funcția de transfer PI

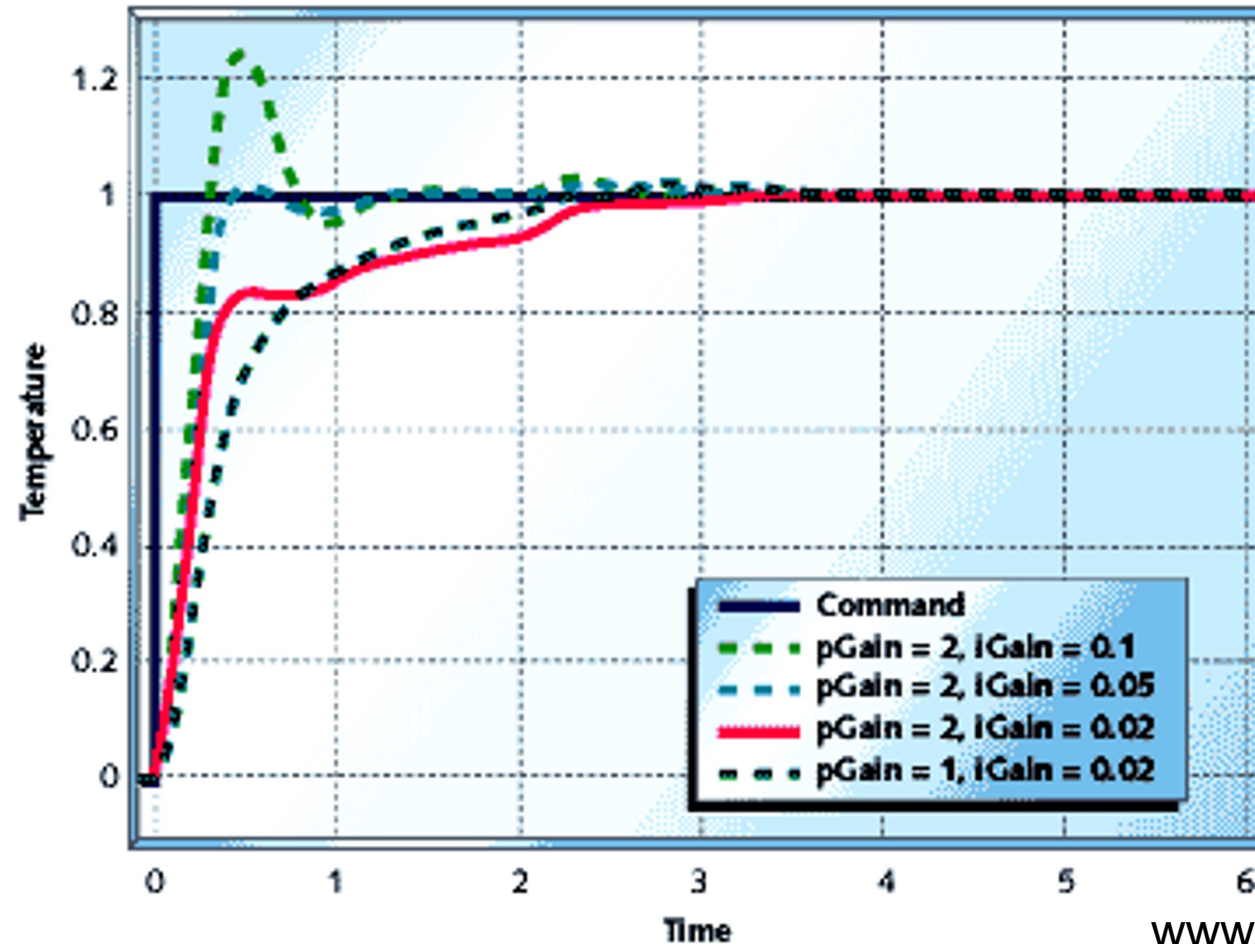


$$E(s) = R(s) - Y(s) \quad U(s) = K_p E(s) + K_I \frac{E(s)}{s} \quad Y(s) = G(s)U(s)$$

$$H(s) = \frac{(K_p + \frac{K_I}{s}) \cdot G(s)}{1 + (K_p + \frac{K_I}{s}) \cdot G(s)}$$

Exemplu

- Pentru regulatorul de temperatură:



Ecuția caracteristică: $1 + \left(K_P + \frac{K_I}{s} \right) \cdot G(s) = 0$

$$e_{ss} = \lim_{k \rightarrow \infty} e(k) = \lim_{s \rightarrow 0} \frac{1}{1 + \left(K_P + \frac{K_I}{s} \right) G(s)}$$

- Eroarea steady-state este zero cât timp sistemul este stabil.

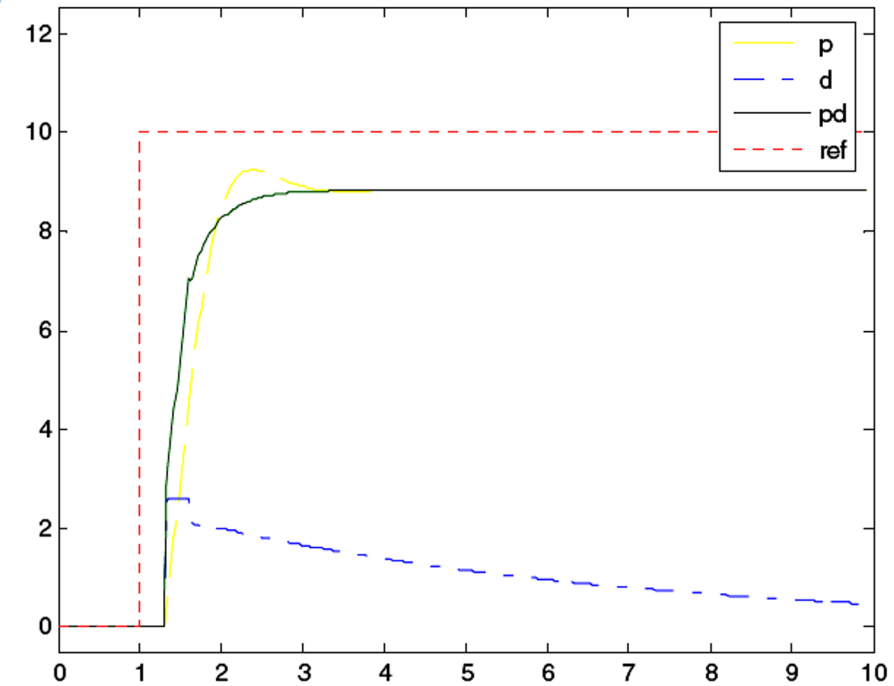
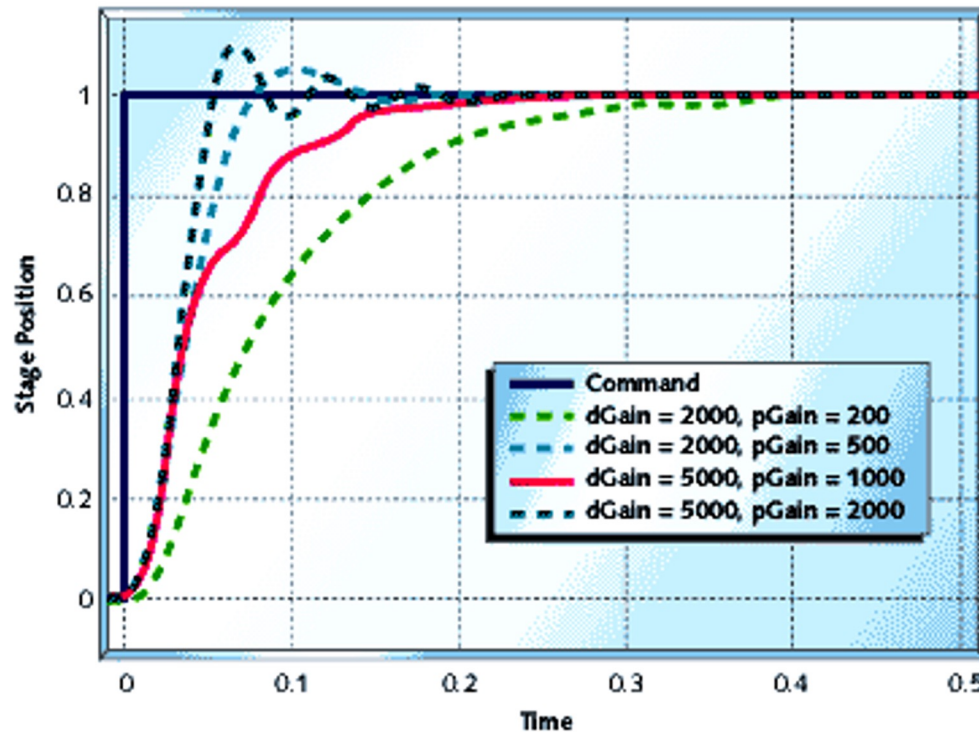
Implementare Regulator PI

```
double iTerm;  
// calculează termenii PI  
// între două limite  
pid->iState += error;  
if (pid->iState > pid->iMax)  
    pid->iState = pid->iMax;  
else if (pid->iState < pid->iMin)  
    pid->iState = pid->iMin;  
iTerm = pid->iGain * iState; // calculează factorul integral
```

- Controlul Integral pur are un timp de răspuns lent și poate produce oscilația sistemului
- Controlul PI are un timp de răspuns mai mic decât I pur.
- Când e stabil, PI ajunge întotdeauna la valoarea de referință ($e_{ss} = 0$)

- Controlul P tratează comportamentul prezent al sistemului
 - Controlul I tratează comportamentul trecut al sistemului
 - Dacă nu stim cum o să se comporte sistemul, nu avem cum sa-l stabilizăm.
- Anticiparea comportamentului -> stabilitate marită
- Controlul D monitorizează viteza de schimbare a erorii sistemului

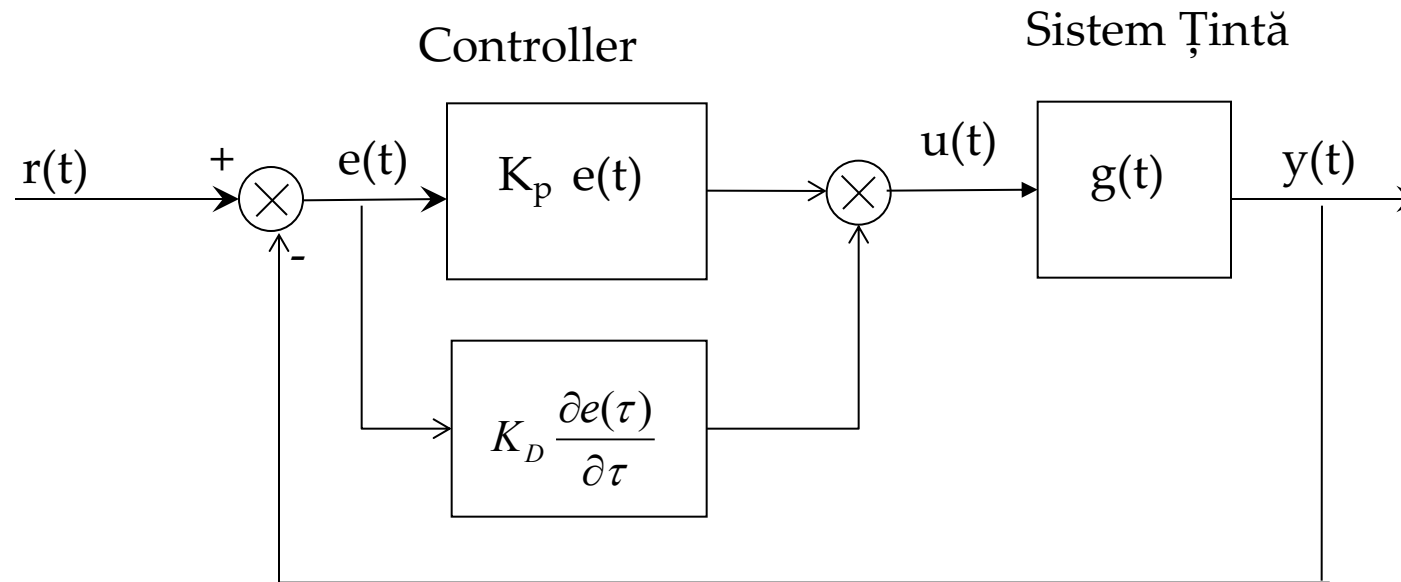
Control PD



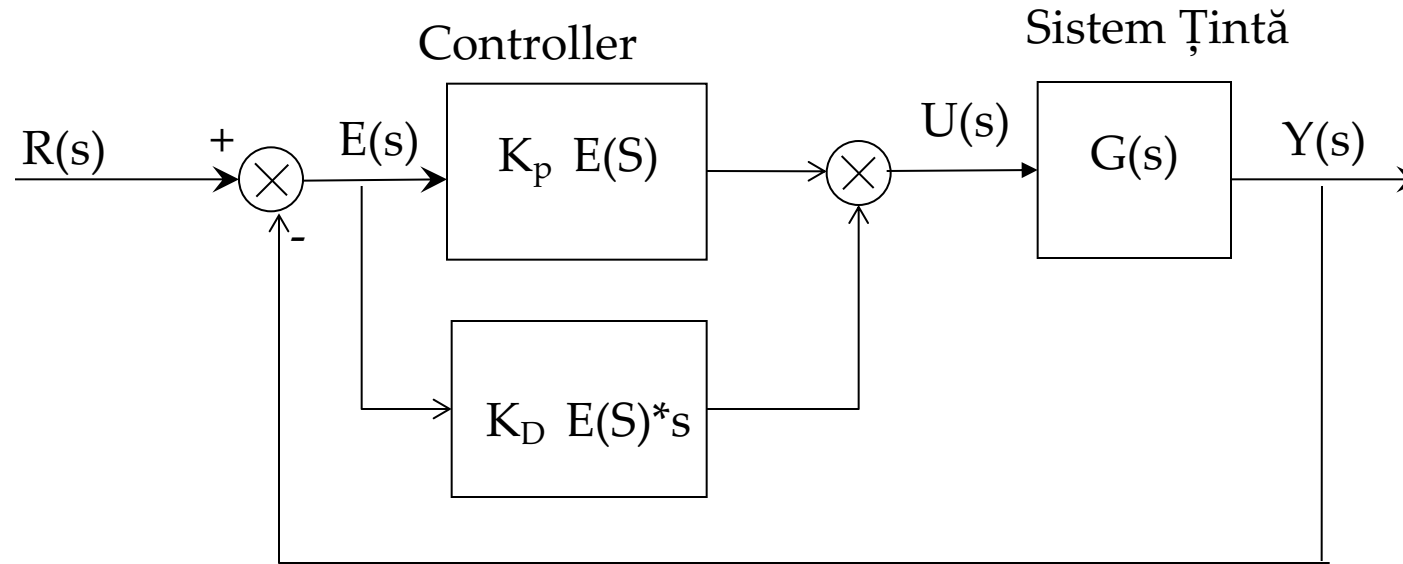
- Controlul diferențial este cel mai puternic dar și cel mai problematic. Sistemul devine instabil la zgomot și la neregularități de eșantionare.

Regulator Proportional-Diferențial

- ieșirea regulatorului este direct proporțională cu rata de creștere a erorii.



Funcția de transfer PD



$$E(s) = R(s) - Y(s) \quad U(s) = K_p \cdot E(s) + K_D \cdot sE(s) \quad Y(s) = G(s)U(s)$$

$$H(s) = \frac{(K_p + sK_D) \cdot G(s)}{1 + (K_p + sK_D) \cdot G(s)}$$

Ecuția caracteristică: $1 + (K_P + sK_D) \cdot G(s) = 0$

$$e_{ss} = \lim_{k \rightarrow \infty} e(k) = \lim_{s \rightarrow 0} \frac{1}{1 + (K_P + sK_D)G(s)}$$

- Pentru e_{ss} , PD se aseamănă cu controlul P. Sistemul poate avea probleme în atingerea valorii de referință.

Implementare Regulator PD

```
double dTerm;
```

```
dTerm = pid->dGain * (position - pid->dState);
```

```
pid->dState = position;
```

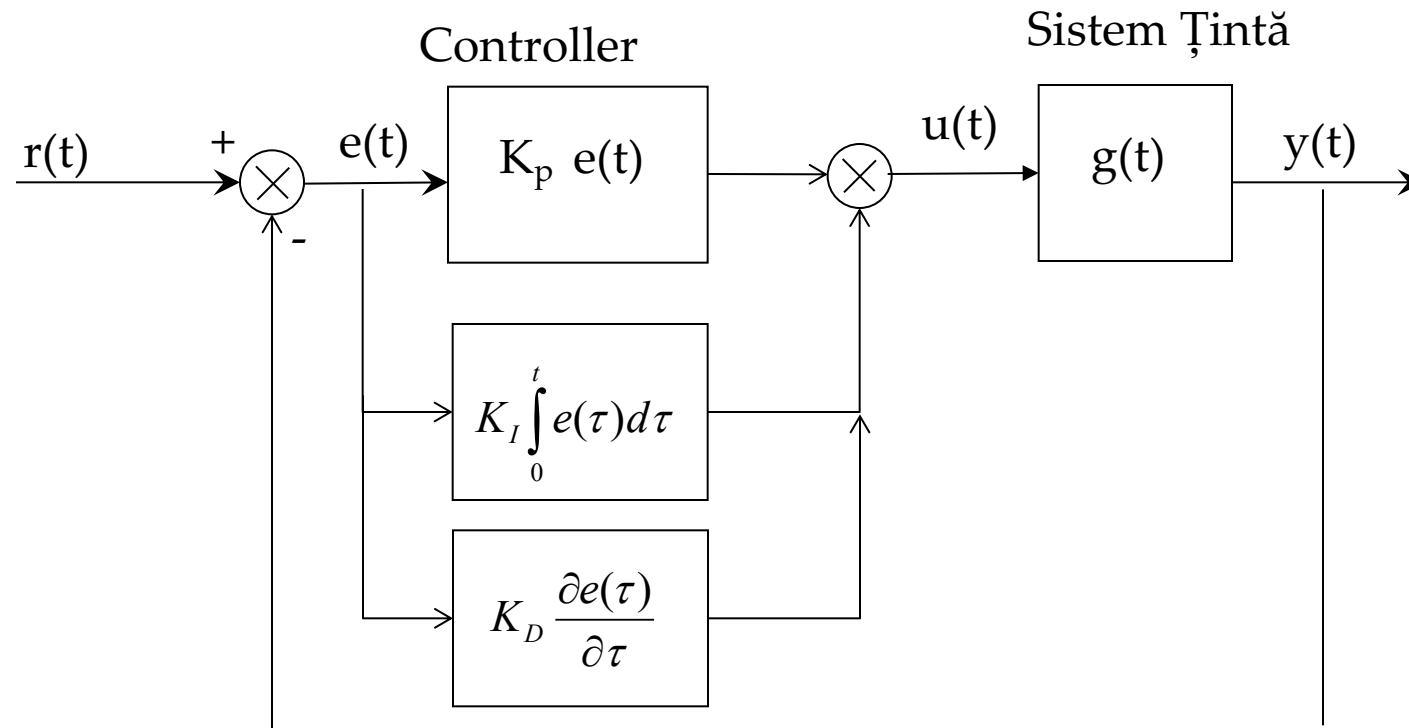
- Controlul PD are un timp de răspuns foarte mic
- Anticipează comportamentul viitor al sistemului folosindu-se de variația erorii măsurate
- Este foarte util într-un sistem de control fin al mișcării/poziției datorită vitezei crescute de răspuns
- Sensibilitate crescută la zgomote și perturbații datorată factorului de amplificare mare.
- Controlul PD pot fi folosit pentru micșorarea depășirii superioare (overshoot) în locul sistemelor cu control P.

- Folosește toate cele trei tipuri de control studiate anterior
- Este cel mai utilizat tip de control în industrie, datorită stabilității lui.
- Performanțe crescute din punctul de vedere al timpului de răspuns, stabilității și erorii de aproximare.

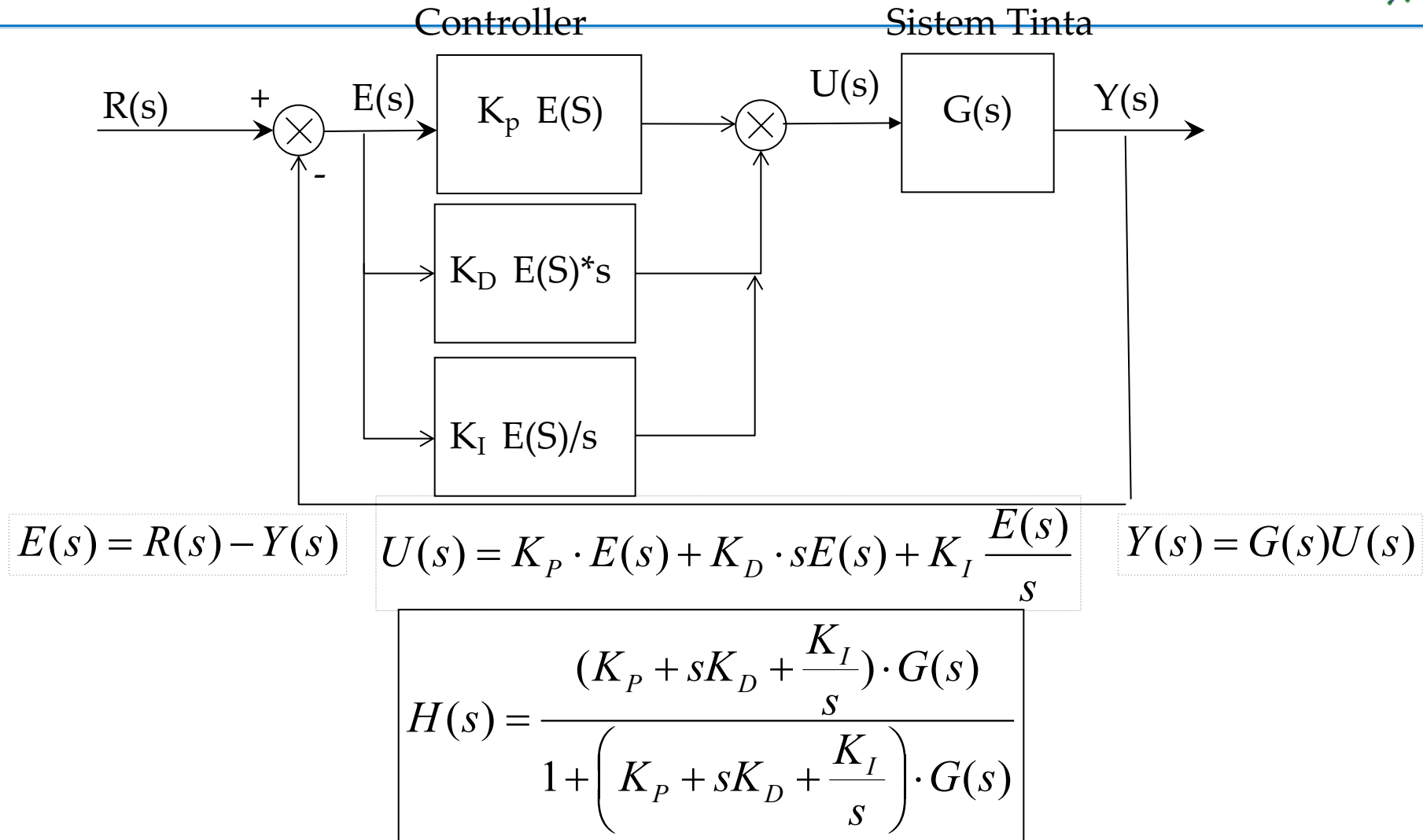
$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt}$$

- P: Contribuie la stabilitatea sistemului, timp de răspuns mediu spre lent
- I: Tracking și rejectia perturbațiilor, timp de răspuns lent. Poate cauza oscilații
- D: Timp rapid de răspuns. Sensibil la zgomot.
- PID: Cel mai folosit tip de controller low-level.
- Notă: stabilitatea nu este garantată.
- Mecanismul de feedback are o abilitate remarcabilă de a tolera incertitudinea indusă de modelul fizic adoptat!

$$u(t) = K_p e(t) + K_D \frac{\partial e(\tau)}{\partial \tau} + K_I \int_0^t e(\tau) d\tau$$

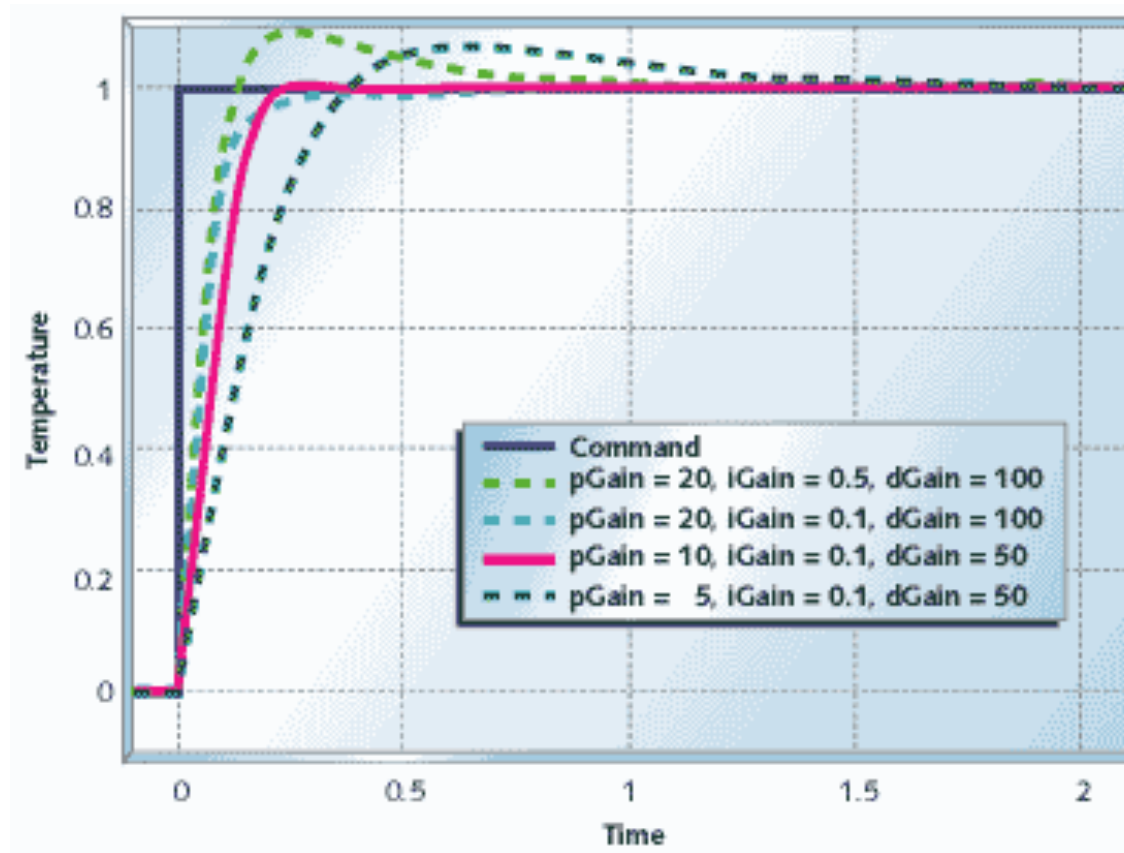


Funcția de transfer PID



Exemplu

- Pentru controllerul de temperatură:



www.embedded.com

Ecuția caracteristică: $1 + \left(K_P + sK_D + \frac{K_I}{s} \right) \cdot G(s) = 0$

$$e_{ss} = \lim_{k \rightarrow \infty} e(k)$$

$$e_{ss} = \lim_{s \rightarrow 0} \frac{1}{1 + \left(K_P + sK_D + \frac{K_I}{s} \right) G(s)}$$

- Dacă parametrii controlului PID sunt aleși incorect, sistemul poate să devină instabil și să devieze de la răspunsul optim cu sau fără oscilație.
- Mai multe metode de estimare:
 - Estimare Manuală
 - Ziegler-Nichols
 - Software Tuning
 - Cohen Coon

- Obținerea matematică a K_P , K_I și K_D nu poate fi posibilă
 - Sistemul este prea complex pentru a fi modelat matematic
 - Costurile de modelare sunt prea mari
- Metoda ad hoc pentru obținerea unor valori rezonabile ale K_P , K_I , și K_D
 - Inițial K_P foarte mic, $K_I = K_D = 0$
 - Mărește K_P până când apar oscilații
 - Setează K_P la jumătate din valoarea obținută
 - Mărește K_I până când răspunsul sistemului este egal cu valoarea dorită
 - K_I mare poate induce oscilații
 - Mărește K_D până când sistemul are un timp de răspuns suficient de mic pentru orice perturbații

- Produce rezultate acceptabile, dar nu optime
- Algoritm:
 - Inițial K_P foarte mic, $K_I = K_D = 0$
 - Mărește K_P până când apar oscilații (K_C – câștigul critic)
 - Măsoară perioada oscilațiilor sistemului (T_C)
 - Determină coeficienții folosind următoarele relații:

	K_P	K_I	K_D
P	$0.5 \cdot K_C$	-	-
PI	$0.45 \cdot K_C$	$1.2 K_p / T_c$	-
PID	$0.6 \cdot K_C$	$2 K_p / T_c$	$K_p T_c / 8$

- Buclă principală, ciclează la infinit
 - Măsoară variabila de ieșire a sistemului
 - De cele mai multe ori convertor AD
 - Măsoară valoarea de referință curentă
 - Apelează PidUpdate pentru a calcula comanda efectorului
 - Setează valoarea curentă pentru efector
 - Convertor DA

```
void main()
{
    double sensor_value, actuator_value, error_current;
    PID_DATA pid_data;
    PidInitialize(&pid_data);
    while (1) {
        sensor_value = SensorGetValue();
        reference_value = ReferenceGetValue();
        actuator_value =
            PidUpdate(&pid_data, sensor_value, reference_value);
        ActuatorSetValue(actuator_value);
    }
}
```

- Pgain, Dgain, Igain sunt constante stabilite cu un algoritm de tuning
- sensor_value_previous
 - Pentru controlul Diferențial
- error_sum
 - Pentru controlul Integral

```
typedef struct PID_DATA {  
    double Pgain, Dgain, Igain;  
    double sensor_value_previous; // find the derivative  
    double error_sum; // cumulative error  
}
```

Calcularea coeficienților

- $$u_t = P \cdot e_t + I \cdot (e_0 + e_1 + \dots + e_t) + D \cdot (e_t - e_{t-1})$$

```
double PidUpdate(PID_DATA *pid_data, double sensor_value,
                double reference_value)
{
    double Pterm, Iterm, Dterm;
    double error, difference;

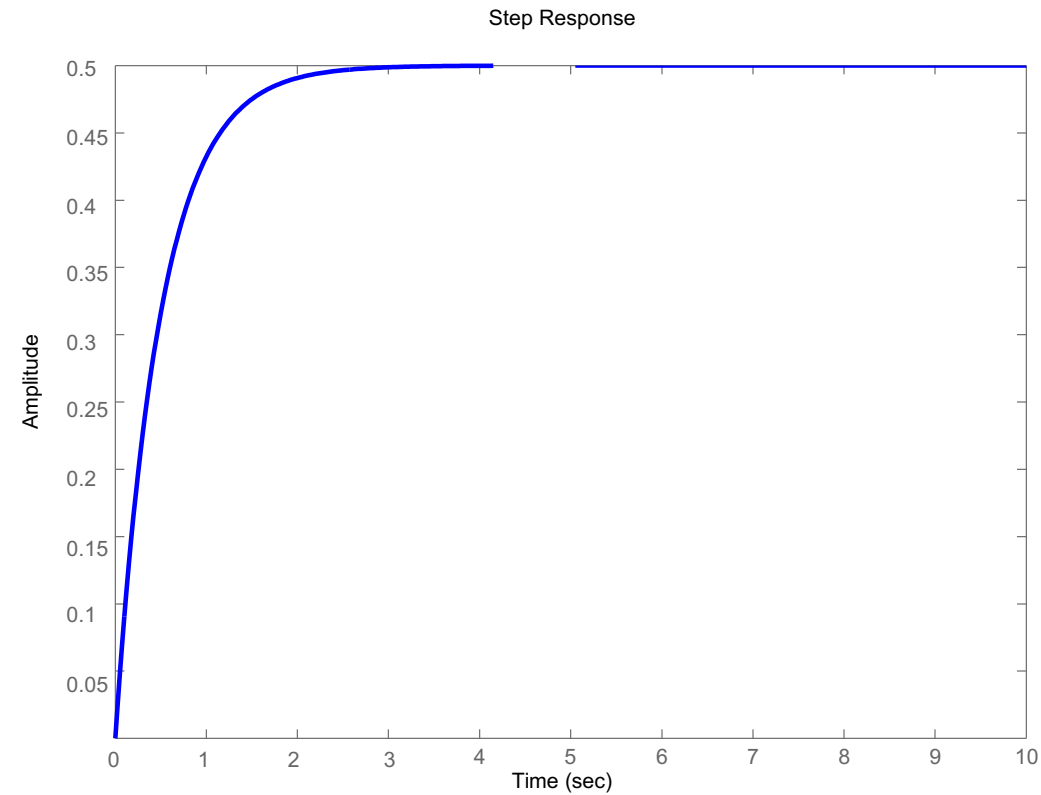
    error = reference_value - sensor_value;
    Pterm = pid_data->Pgain * error; /* proportional term*/
    pid_data->error_sum += error; /* current + cumulative*/
    // the integral term
    Iterm = pid_data->Igain * pid_data->error_sum;
    difference = pid_data->sensor_value_previous -
                sensor_value;
    // update for next iteration
    pid_data->sensor_value_previous = sensor_value;
    // the derivative term
    Dterm = pid_data->Dgain * difference;
    return (Pterm + Iterm + Dterm);
}
```

Cruise-Controller (Again)

$$\dot{x} = \frac{c}{m} u - \gamma x \quad c = 1, m = 1, \gamma = 0.1, r = 1$$

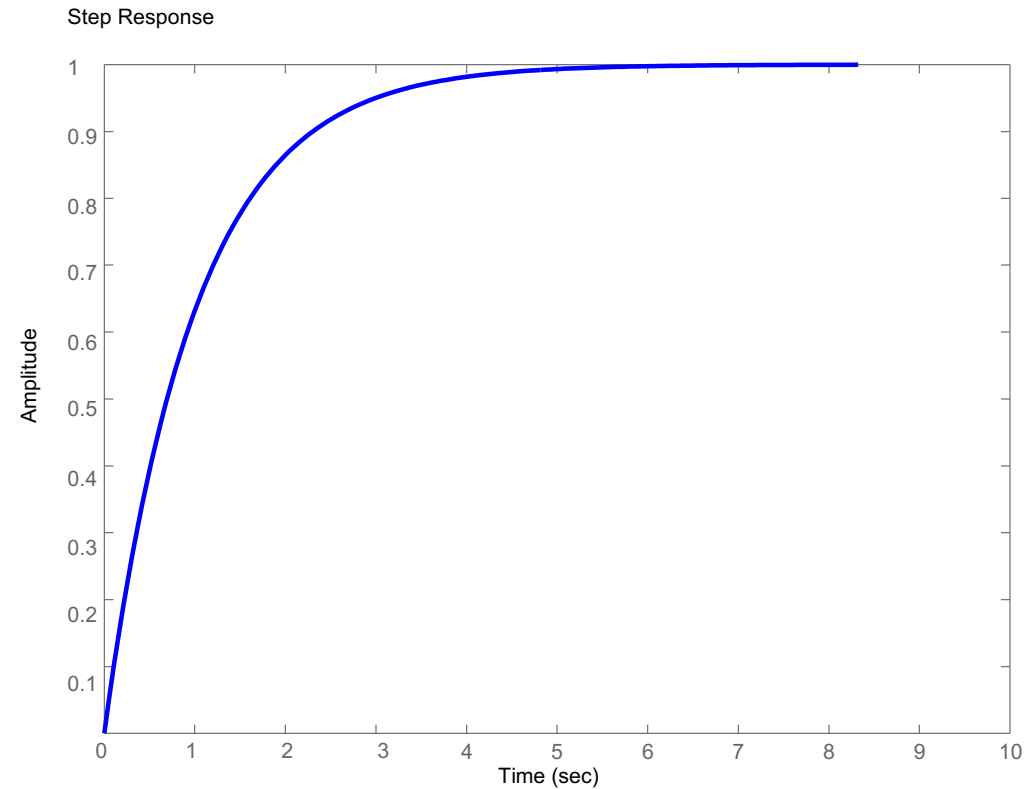


Cruise-Controller (Again)



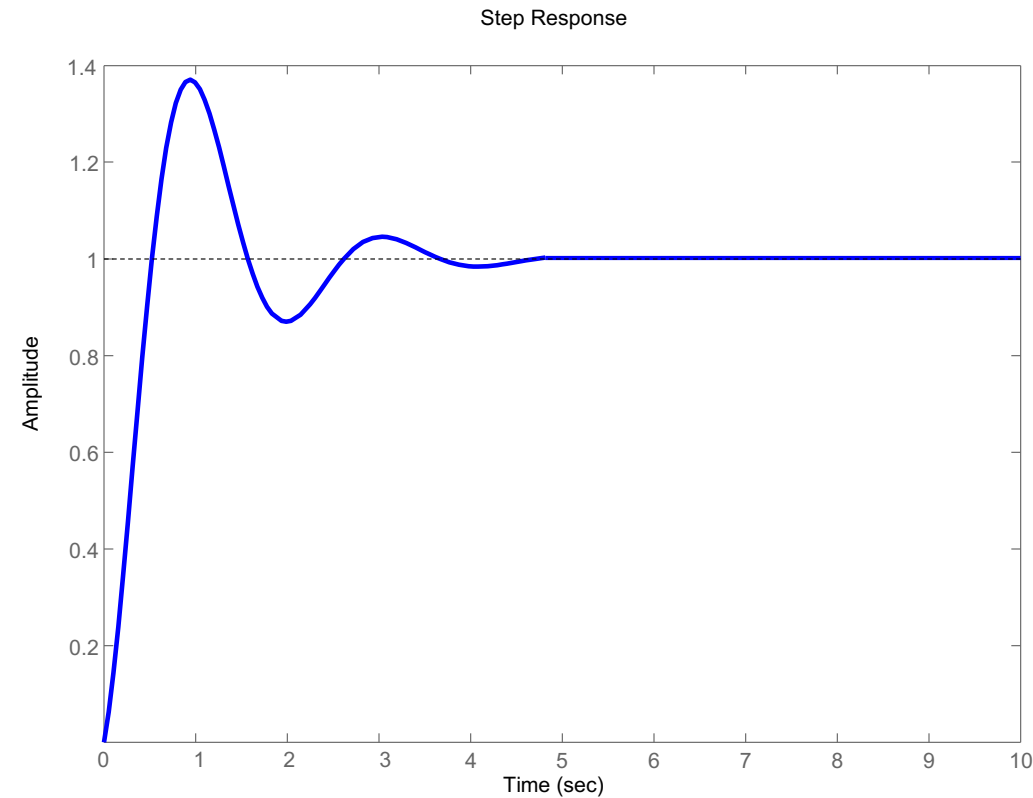
$$k_P = 1, \quad k_I = 0, \quad k_D = 0$$

Cruise-Controller (Again)



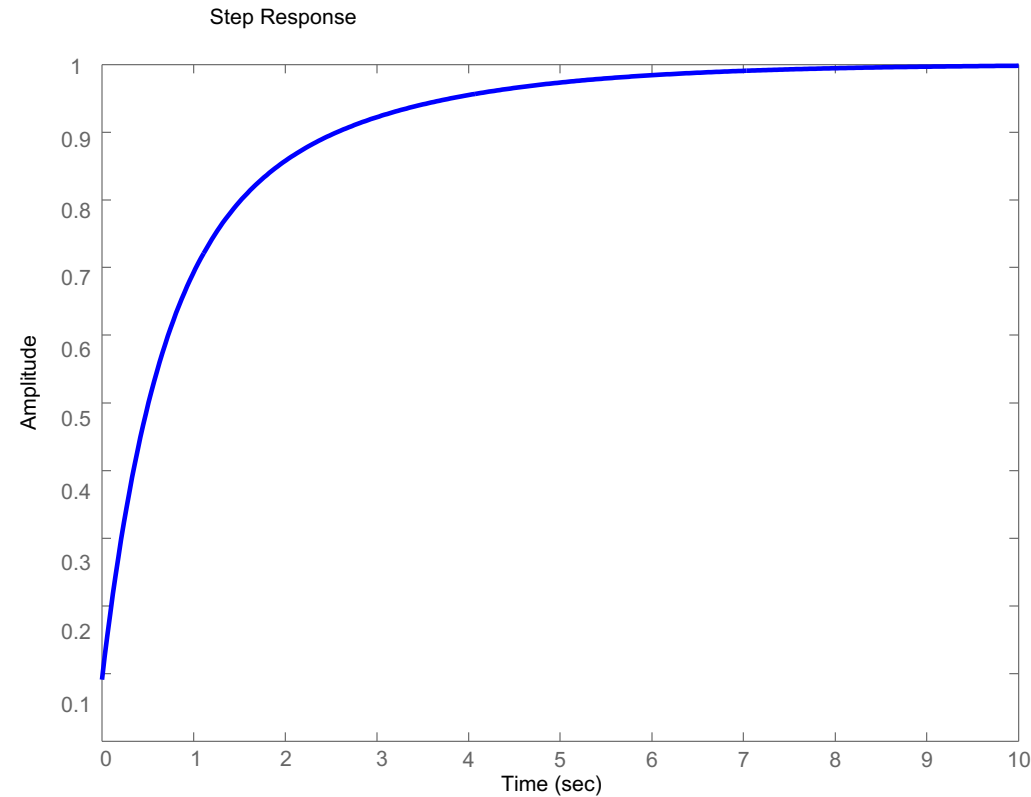
$$k_P = 1, \quad k_I = 1, \quad k_D = 0$$

Cruise-Controller (Again)



$$k_P = 1, \quad k_I = 10, \quad k_D = 0$$

Cruise-Controller (Again)

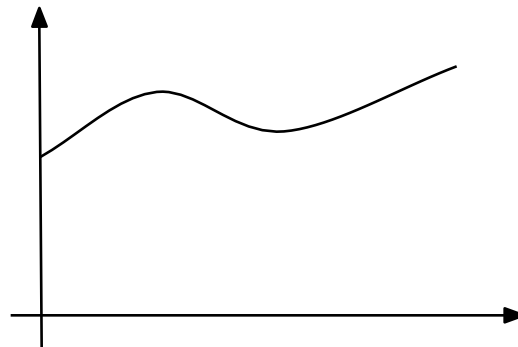


$$k_P = 1, \quad k_I = 1, \quad k_D = 0.1$$

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt}$$

- Cum să transformăm o integrală în ceva implementabil în cod?

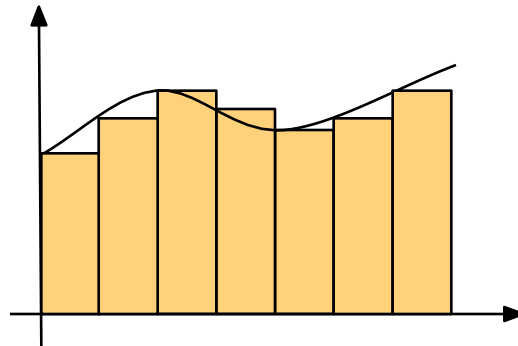
$$\Delta t \quad (\text{sample time}) \quad \dot{e} \approx \frac{e_{new} - e_{old}}{\Delta t}$$



$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt}$$

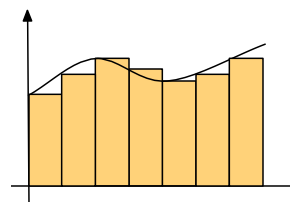
- Cum să transformăm o integrală în ceva implementabil în cod?

$$\Delta t \quad (\text{sample time}) \quad \dot{e} \approx \frac{e_{new} - e_{old}}{\Delta t}$$



$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt}$$

- Cum să transformăm o integrală în ceva implementabil în cod?



Δt (sample time)

$$\dot{e} \approx \frac{e_{new} - e_{old}}{\Delta t}$$
$$\int_0^t e(\tau) d\tau \approx \sum_{k=0}^N e_k \Delta t$$

$$\Delta t E_{new} = \Delta t$$
$$E_{new} = E_{old} + e$$

- De fiecare dată când este apelat regulatorul

```
read e; e_dot=e-old_e; E=E+e;  
u=kP*e+kD*e_dot+kI*E; old_e=e;
```

Notă: Coeficienții includ acum și timpul de eșantionare și trebuie scalați în consecință

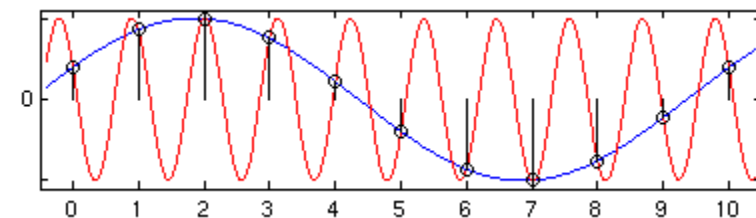
Exemplu: Regulator de altitudine pentru un quadcopter



$$\ddot{x} = cu - g \quad u = k_p e + k_I \int e \, dt + k_D \dot{e}$$

- Erori de reprezentare
 - Nu toate valorile numerice pot fi reprezentate intern datorită limitărilor de memorie
- Overflow
 - Depășirea superioară sau inferioară a domeniului de reprezentare
- Aliasing
- Viteza de procesare

- Două sau mai multe semnale periodice devin indistinctibile atunci când sunt eșantionate la o anumită frecvență.
- Exemplu:
 - Eșantionate la 2.5 Hz, următoarele semnale nu pot fi diferențiate
 - $y(t)=1.0*\sin(6\pi t)$, $f = 3$ Hz
 - $y(t)=1.0*\sin(\pi t)$, $f = 0.5$ Hz
 - Conform teoremei Nyquist, frecvența minimă de eșantionare pentru primul semnal:
 $3*2 = 6$ Hz



- Produce întârzieri inevitabile
 - Acțiunea calculată se petrece mai târziu
- Întârzierile trebuiesc prevăzute și estimate în implementarea oricărui sistem de control.
- Întârzierile hardware sunt de obicei ușor de calculat
 - Structura sincronă a procesoarelor ajută
- Întârzierile software sunt mai greu de prevăzut
 - Codul trebuie organizat a.i. întârzierile să fie minime
 - Software cu întârzieri predictibile
 - Rutine declanșate la intervale regulate de timp
 - Limbaj de programare sincron (Esterel, Chuck)

- Cost redus
 - Un control analogic este foarte scump mai ales dacă este proiectat să fie imun la perturbații
 - Uzură, temperatură, erori de fabricație
 - Controlul numeric înlocuiește circuitele analogice complexe cu programe complexe
- Programabilitate
 - Controlul poate fi “upgradat”
 - Schimbări la tipul controlului, câștig, sunt ușor de făcut
 - Se adaptează la schimbările sistemului
 - Datorate îmbătrânirii, temperaturii etc.
 - “future-proof”
 - Ușor de adaptat la un alt standard