Programska implementacija digitalnih regulatora

Jadranko Matuško
Šandor Ileš

Fakultet elektrotehnike i računarstva

19. siječnja 2024.

Programska implementacija digitalnih regulatora

- Na prethodnom predavanju upoznali ste se s dva različita pristupa projektiranja diskretnog regulatora
  - Emulacija analognog regulatora koristeći EMUL1 ili EMUL2 metodu)
  - Diskretizacija procesa za potrebe sinteze regulatora u diskretnom području
- Na današnjem predavanju govorit ćemo o programskoj implementaciji digitalnih regulatora

Podsjetnik: Emulacija analognog regulatora

Postupak projektiranja:

- Projektiranje analognog regulatora $G_R(s)$
- Odabir prikladnog vremena diskretizacije T
- Diskretizacija regulatora uz odabrano vrijeme diskretizacije $G_R(s) \rightarrow G_R(z)$
- Simulacijska provjera diskretnog sustava upravljanja.

Pritom se postupka diskretizacije provodi nekim od numeričkih postupaka:

- Eulerova unaprijedna diskretizacija
- Eulerova unazadna diskretizacija
- Tustinov postupak

Primjer 1: Diskretizacija analognog PI regulatora

- Zbog jednostavnosti promotrimo idealni PI regulator

$$u(t) = K_p e(t) + \frac{K_p}{T_I} \int_0^t e(t)dt \tag{1}$$

- Idealni PI regulator može se zapisati u sljedećem obliku:

$$\frac{du}{dt} = K_p \frac{de}{dt} + K_i e(t). \tag{2}$$

- Ako derivaciju aproksimiramo Eulerovim unazadnim postupkom:

$$\frac{dx}{dt} \approx \frac{x(k) - x(k-1)}{T} \tag{3}$$

- Slijedi:

$$\frac{u(k) - u(k-1)}{T} = K_p \frac{e(k) - e(k-1)}{T} + K_i e(t). \tag{4}$$

Primjer 1: Diskretizacija analognog PI regulatora

- Jednadžba diferencija:

$$\begin{aligned} u(k) &= (K_p + TK_i)e(k) - K_p e(k-1) + u(k-1) \\ &= ae(k) + be(k-1) + u(k-1). \end{aligned} \tag{5}$$

- Diskretizirani regulator

$$G_R(z) = \frac{a + bz^{-1}}{1 - z^{-1}} = \frac{az + b}{z - 1}. \tag{6}$$

Programska implementacija

Programska implementacija

```
e_prev = e;
e = r - y;
u_prev = u;
u = u_prev + a*e + b*e_prev;
```

- Ovakva implementacija je računski najbrža
- Teško je mijenjati parametre regulatora budući da su skriveni unutar koeficijenata *a* i *b*
- Gdje je integrator? Kako riješiti problem namatanja integratora?

Programska implementacija

Programska implementacija

```
e_prev = e;
e = r - y;
u_prev = u;
u = u_prev + a*e + b*e_prev;
```

- Ovakva implementacija je računski najbrža
- Teško je mijenjati parametre regulatora budući da su skriveni unutar koeficijenata *a* i *b*
- Gdje je integrator? Kako riješiti problem namatanja integratora?
- Rješenje: Razdvojiti upravljački signal na komponente koje odgovaraju proporcionalnom i integralnom djelovanju

Programska implementacija

Opcija - A

```
e = r - y;
e_int = e_int + T e ;
u = K_p e + K_i e_int;
```

- Prvo integriramo signal pogreške pa ga množimo pojačanjem
- U ustaljenom stanju nije moguće mijenjati integralno pojačanja bez utjecaja na izlaznu veličinu

Opcija - B

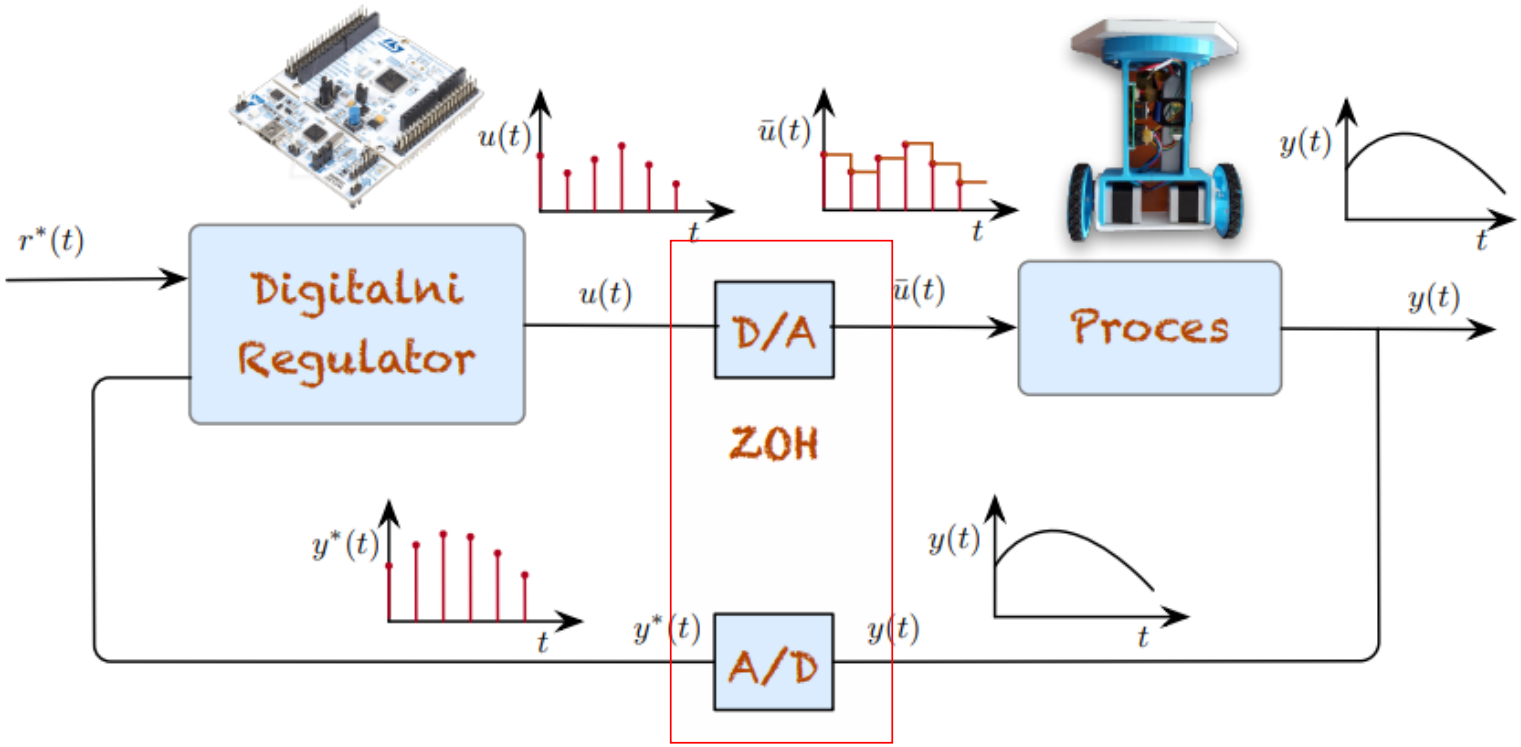
```
u_int = u_int + K_i T e;
u = K_p e + u_int
```

- U ovom slučaju integrator djeluje direktno na upravljački siganl
- Moguće je umnožak K_i i T zamjeniti konstantom
- Moguće je mijenjati integralno pojačanja bez utjecaja na izlaznu veličinu

Implementacija PI regulatora s Antiwindupom

```
if ((sat < 0 && e < 0) || (sat > 0 && e > 0))
 ;
/* do nothing if there is saturation, and error is in the same direction;
 * if you're careful you can implement as "if (sat*e > 0)"
 */
else
 x_integral = x_integral + Ki2*e;
(x_integral,sat) = satlimit(x_integral, x_minimum, x_maximum);

x = limit(Kp*e + x_integral, x_minimum, x_maximum)

;


/* satlimit(x, min, max) does the following:
 * if x is between min and max, return (x,0)
 * if x < min, return (min, -1)
 * if x > max, return (max, +1)


*

 * limit(x, min, max) does the following:
 * if x is between min and max, return x
 * if x < min, return min
 * if x > max, return max
 */
```

Implementacija algoritama u ugradbenim računalnim sustavima



Implementacija algoritama u konačnoj aritmetici

Prilikom analize i sisteze sustava upravljanja obično se pretpostavlja aritmetiku s beskonačnom preciznošću, odnosno pretpostavlja se da su koeficijenti i varijable realni brojevi.

Uzroci pogrešaka prilikom digitalne implementacije upravljačkih algoritama u konačnoj aritmetici su sljedeći:

- Kvantizacija u procesu A/D pretvorbe,
- Kvantizacija koeficijenata algoritma (regulatora),
- Efekti zaokruživanja i preljeva prilikom obavljanja aritmetičkih operacija (zbrajanje, oduzimanje, množenje, dijeljenje), izračuna funkcija kao i drugih operacija.
- Kvantizacija u procesu D/A pretvorbe.

Razina pogrešaka ovisi o:

- duljini riječi,
- vrsti aritmetike (s nepomičnim ili s pomičnim zarezom)
- realizaciji regulatora.

Kvantizacija u procesu A/D i D/A pretvorbe

Rezolucija A/D i D/A pretvornika obično je relativno niska:

- A/D: 10-16 bita,
- D/A: 8-12 bita.

Kvantizacija je nelinearni efekt koji može pruzročiti pojavu graničnog ciklusa (engl. limit cycle) i pomaka (engl. bias).

Prilikom analize tih efekata mogu se koristi sljedeći pristupi:

- Nelinearna analiza
  - Opisna funkcija,
  - Teorija relejnih oscilacija.
- Linearna analiza
  - Modeliranje kvantizacije kao stohastičkog poremećaja.

Primjer: Upravljanje procesom s dvostrukim integratorom

- Pretpostavimo da je proces opisan sljedećom prijenosnom funkcijom:

$$G(s) = \frac{1}{s}, \tag{7}$$

uz vrijeme uzorkovanja:

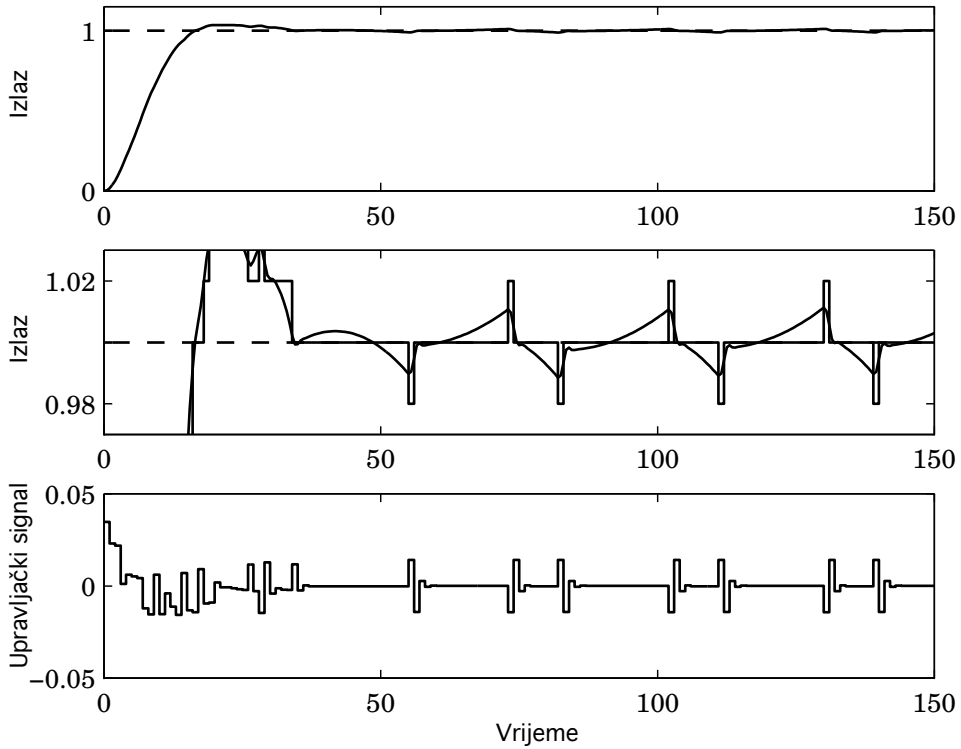
$$T = 1\,[s]. \tag{8}$$

- Neka se proces upravlja digitalnim regulatorom PID strukture opisanim izrazom:

$$C(z) = \frac{0.715z^2 - 1.281z + 0.518}{(z-1)(z+0.188)}. \tag{9}$$

Primjer: Upravljanje procesom s dvostrukim integratorom (2)

Simulacija uz A/D pretvarač rezolucije $\delta y = 0.02$



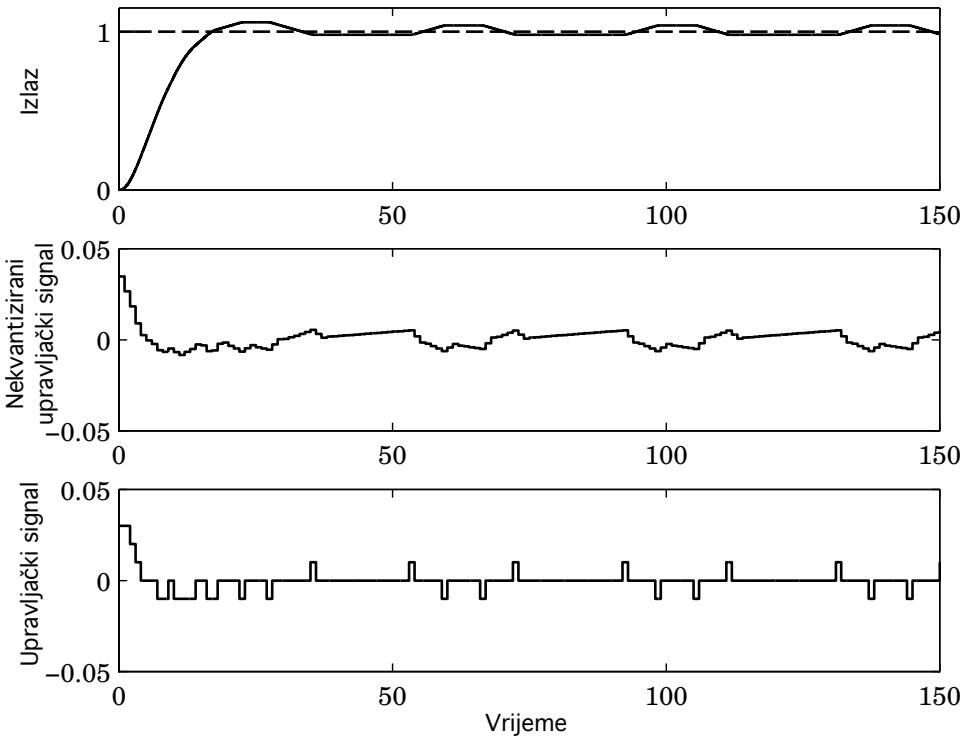
Slika 1: Odziv sustava upravljanja s A/D pretvaračem

Granični ciklus:

- Period: 28 [s]
- Amplituda: 0.01

Primjer: Upravljanje procesom s dvostrukim integratorom (2)

Simulacija uz D/A pretvarač rezolucije $\delta u = 0.01$

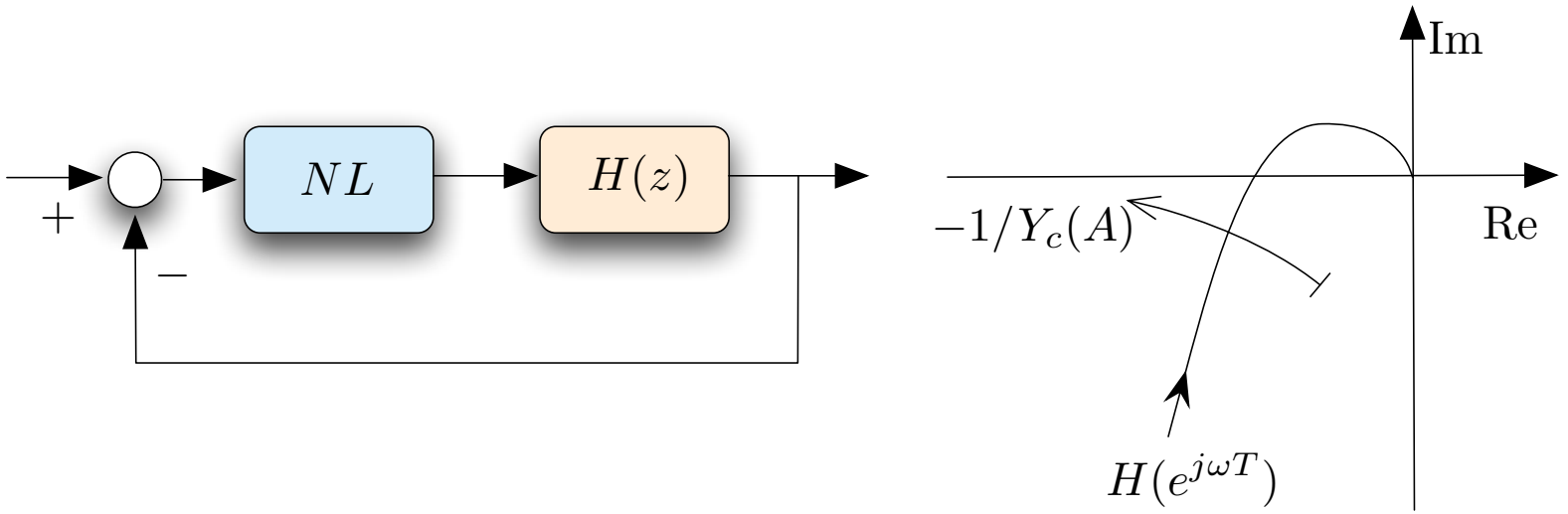


Slika 2: Odziv sustava upravljanja s D/A pretvaračem

Granični ciklus:

- Period: 39 $[s]$
- Amplituda: 0.005

Analiza primjenom opisne funkcije



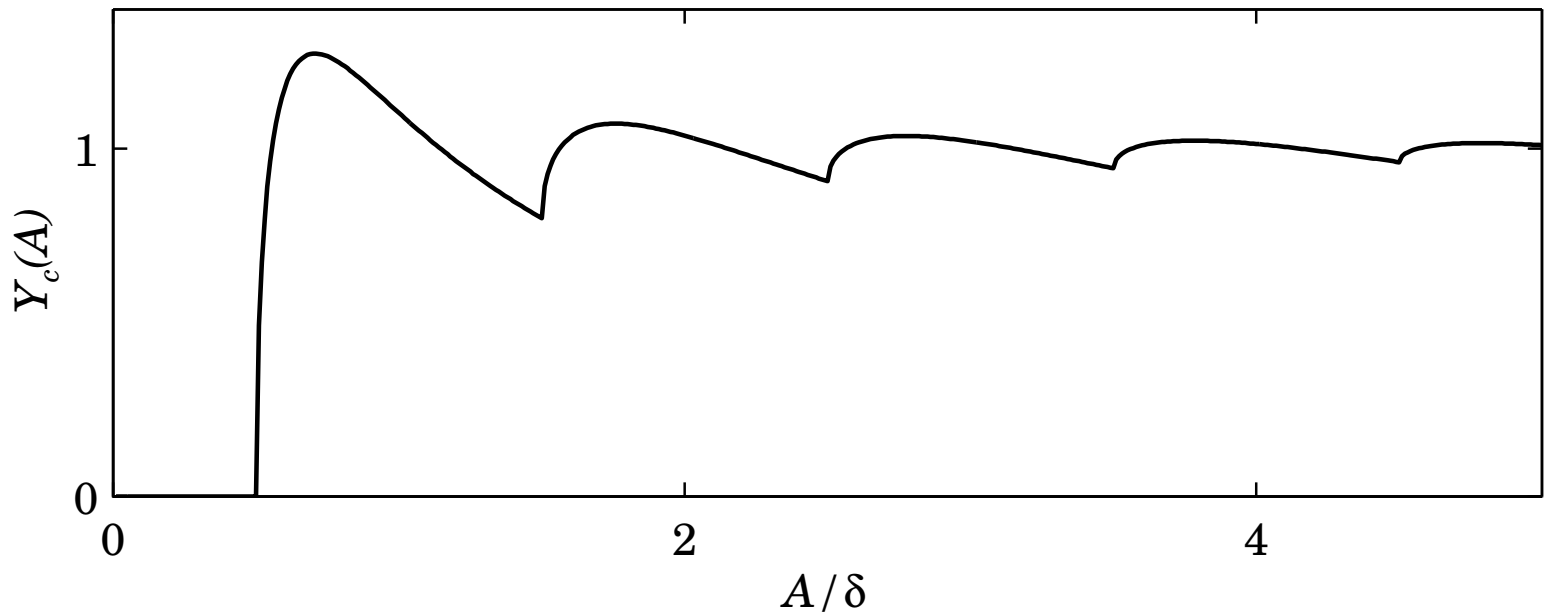
a) b)

- Granični ciklus frekvencije ω_1 i amplitude A_1 pojavit će se ako vrijedi:

$$H(e^{j\omega_1}) = -\frac{1}{Y_c(A_1)}. \tag{10}$$

Opisna funkcija elementa kvantizacije

$$Y_c(A) = \begin{cases} 0 & 0 < A < \dfrac{\delta}{2} \\ \dfrac{4\delta}{\pi A} \displaystyle\sum_{i=1}^{n} \sqrt{1 - \left(\frac{2j-1}{A}\delta\right)^2} & \dfrac{2n-1}{2}\delta < A < \dfrac{2n+1}{2}\delta \end{cases} \tag{11}$$



Aritmetika s pomičnim zarezom

- Sklopovski podržana na modernim mikroprocesorima i mikrokontrolerima.
- Brojevi je prikazuju kao:

$$\pm f \cdot 2^{\pm e} \tag{12}$$

  pri čemu je:
  - f - mantisa (frakcija),
  - e - eksponent.
- Binarni zarez je pomičan (engl. floating) o ovisi o iznosu eksponenta
- Dinamički raspon i rezolucija

Aritmetika s pomičnim zarezom

```
\#include <stdio.h>

main() {
float a[] = { 10000.0, 1.0, 10000.0 };
float b[] = { 10000.0, 1.0, -10000.0 };
float sum = 0.0;
int i;

for (i=0; i<3; i++)
sum += a[i]*b[i];

printf("sum = %f\n", sum);
}
```

- Za razliku od aritmetike s pomičnim zarezom, za operacije u konačnoj aritmetici ne vrijede svojstva asocijativnosti i distributivnosti.

Aritmetika u ugradbenim sustavima

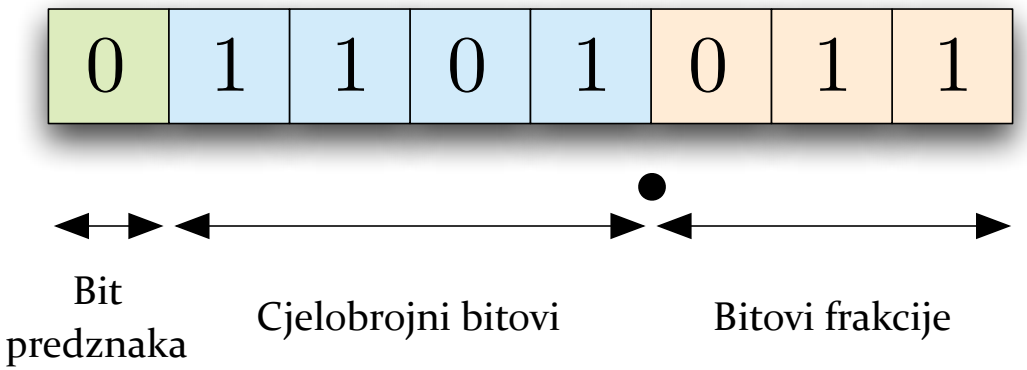
- Mikroprocesori i mikrokontroleri koji se koriste u ugradbenim sustavima obično nemaju sklopovsku podršku za aritmetiku s pomičnim zarezom.

Mogućnosti:

- Softverska emulacija aritmetike s pomičnim zarezom,
  - Velik kod, sporo izvođenje
- Aritmetika s nepomičnim zarezom.
  - Kompaktan kod, brzo izvođenje

Aritmetika s nepomičnim zarezom

- Realni se broj u aritmetici s nepomičnim zarezom prikazuje kao cjelobrojna vrijednost X s $N = m + n + 1$ bitova, pri čemu je:
  - N - duljina riječi,
  - m - broj cjelobrojnih bitova,
  - n - broj bitova frakcije.



- Prikazani format zapisa ponekad se naziva Q-formatom i označava $Qm.n$

Raspon i rezolucija u aritmetici s nepomičnim zarezom

Pretvorba iz i u aritmetiku s nepomičnim zarezom

- Pretvorba realnog broja u broj u aritmetici s nepomičnim zarezom:

$$X := \text{round}(x \cdot 2^n) \tag{13}$$

- Pretvorba broja zapisanog u aritmetici s nepomičnim zarezom u realni broj:

$$x := X \cdot 2^{-n} \tag{14}$$

- Primjer: Broj $x = 13.4$ prikazuje se u formatu $Q4.3$ kao:

$$X = \text{round}(13.4 \cdot 2^3) = 107 (= 01101011_2) \tag{15}$$

Izvedba regulatora

- Linearni regulator:

$$C(z) = \frac{b_0 + b_1 z^{-1} + \ldots + b_n z^{-n}}{1 + a_1 z^{-1} + \ldots + a_n z^{-n}}, \tag{16}$$

može se realizirati različite načine:

- Direktna izvedba,
- Serijska (kaskadna) izvedba,
- Paralelna izvedba,
- Kanonička izvedba.

Direktna izvedba

- Direktna izvedba slijedi iz zapisa regulatora u obliku jednadžbe diferencija:

$$u(k) = \sum_{i=0}^{n} b_i y(k-i) - \sum_{i=1}^{n} a_i u(k-i). \tag{17}$$

- Osnovna svojstva ovakve izvedbe su sljedeća:
  - Izvedba je neminimalna, tj. koriste se svi ulazi i izlazi),
  - Izvedba je vrlo osjetljiva na efekt kvantizacije koeficijenta,

  te iz tih raloga ovu izvedbu treba izbjegavati.

Osjetljivost položaja polova na pogreške koeficijenata

- Kako pogreške koeficijenata, prouzročene efektom njihove kvantizacije, utječu na položaj polova? Neka je karakteristični polinom sustava dan izrazom

$$A(z) = 1 - \sum_{k=1}^{n} a_k z^{-k} = \prod_{i=1}^{n} (1 - p_k z^{-1}), \tag{18}$$

  pri čemu su a_k koeficijenti i p_k korijeni polinoma (polovi sustava).
- Osjetljivost položaja polova sustava određena je sljedećom funkcijom osjetljivoti:

$$S_{p_i}^{a_k} = \frac{\partial p_i}{\partial a_k} \tag{19}$$

Osjetljivost položaja polova na pogreške koeficijenata

- Primjenom pravila o derivaciji složene funkcije slijedi:

$$\frac{\partial A(z)}{\partial a_k} = \frac{\partial A(z)}{\partial p_i}\frac{\partial p_i}{\partial a_k} \tag{20}$$

odakle se dobije

$$\frac{\partial p_i}{\partial a_k} = \frac{\frac{\partial A(z)}{\partial a_k}}{\frac{\partial A(z)}{\partial p_i}} = -\frac{z^{-k}}{\prod_{j=1,j=i}^{n}(1-p_j z^{-1})}. \tag{21}$$

- Uvrštenjem $z = p_i$ slijedi:

$$\frac{\partial p_i}{\partial a_k} = -\frac{p_i^{n-k}}{\prod_{j=1,j\neq i}^{n}(p_j - p_i)} \tag{22}$$

Osjetljivost položaja polova na pogreške koeficijenata - 2

Što se može zaključiti na temelju izraza

$$\frac{\partial p_i}{\partial a_k} = -\frac{p_i^{n-k}}{\prod_{j=1, j\neq i}^{n} (p_j - p_i)}$$

- Ako su polovi sustava blizu jedna drugome tada je osjetljivost na promjene iznosa koeficijenata vrlo velika.
- Najveća je osjetljivost položaja polova na iznos parametra a_n, uz pretpostavku stabilnog regulatora ($|p_i| < 1$).
- Konjugirano kompleksni polovi blizu realne osi predstavljaju također nepovoljan slučaj.
- Analogno razmatranje vrijedi i za osjetljivost nula sustava.

Ilustracija osjetljivosti položaja polova i nula

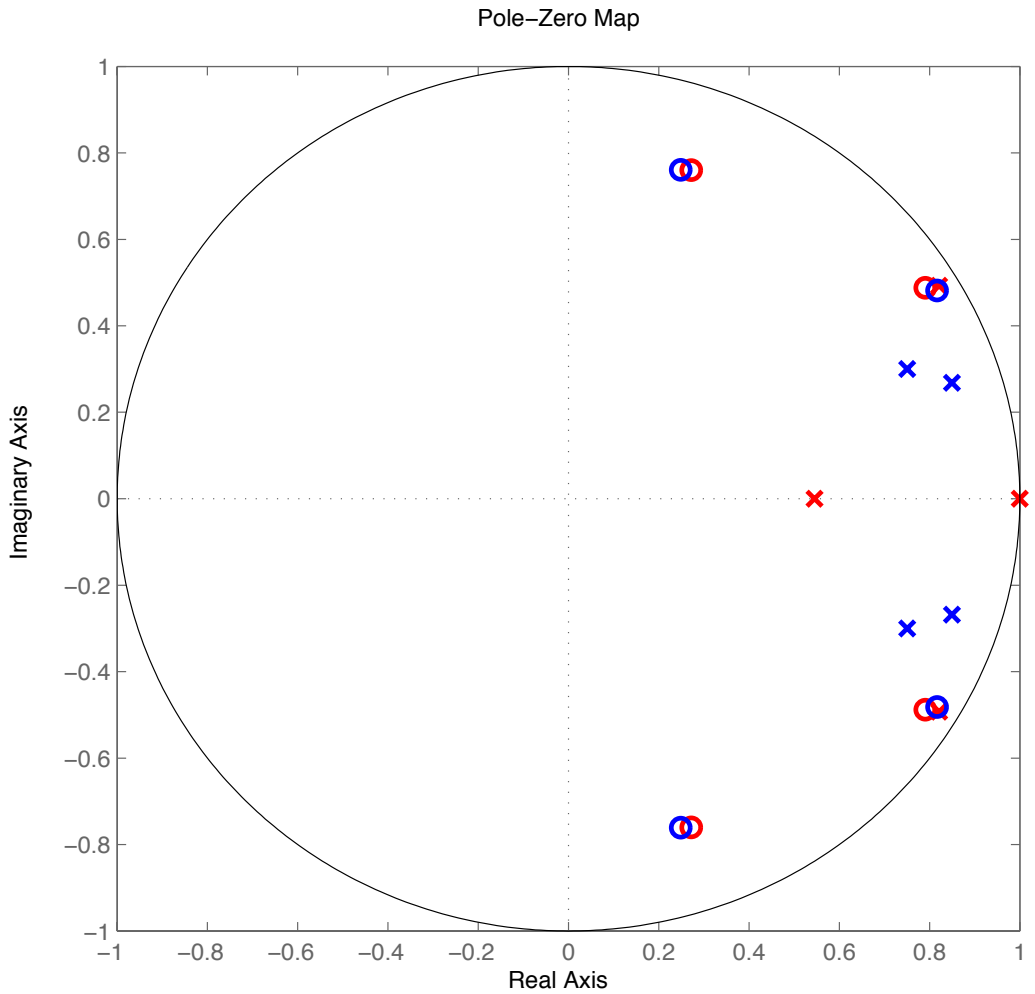
- Neka je idealna prijenosna funkcija regulatora koju je potrebno implementirati u mikrokontroleru s aritmetikom s nepomičnim zarezom $Q8.4$ dana izrazom:

$$C(z) = \frac{z^4 - 2.13z^3 + 2.351z^2 - 1.493z + 0.576}{z^4 - 3.2z^3 + 3.9971z^2 - 2.301z + 0.5184} \tag{23}$$

- Kvantizirana prijenosna funkcija glasi:

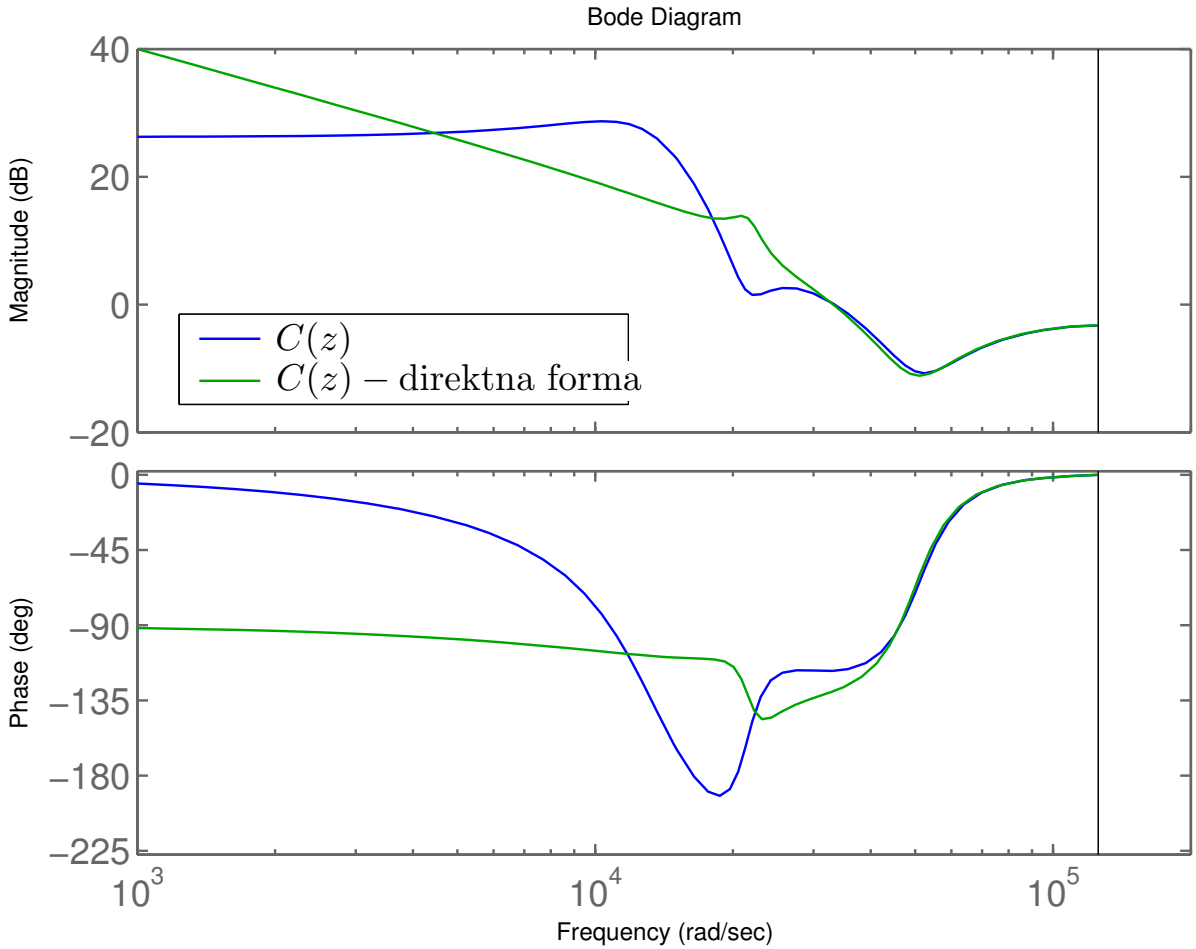
$$C_q(z) = \frac{z^4 - 2.125z^3 + 2.375z^2 - 1.5z + 0.5625}{z^4 - 3.188z^3 + 4z^2 - 2.312z + 0.5} \tag{24}$$

Realizacija u direktnoj formi



Slika 3: Položaj polova i nula regulatora i direktnoj izvedbi (plava- idealni regulator, crvena - regulator s kvantiziranim koeficijentima.

Realizacija u direktnoj formi u aritmetici *Q*8.4



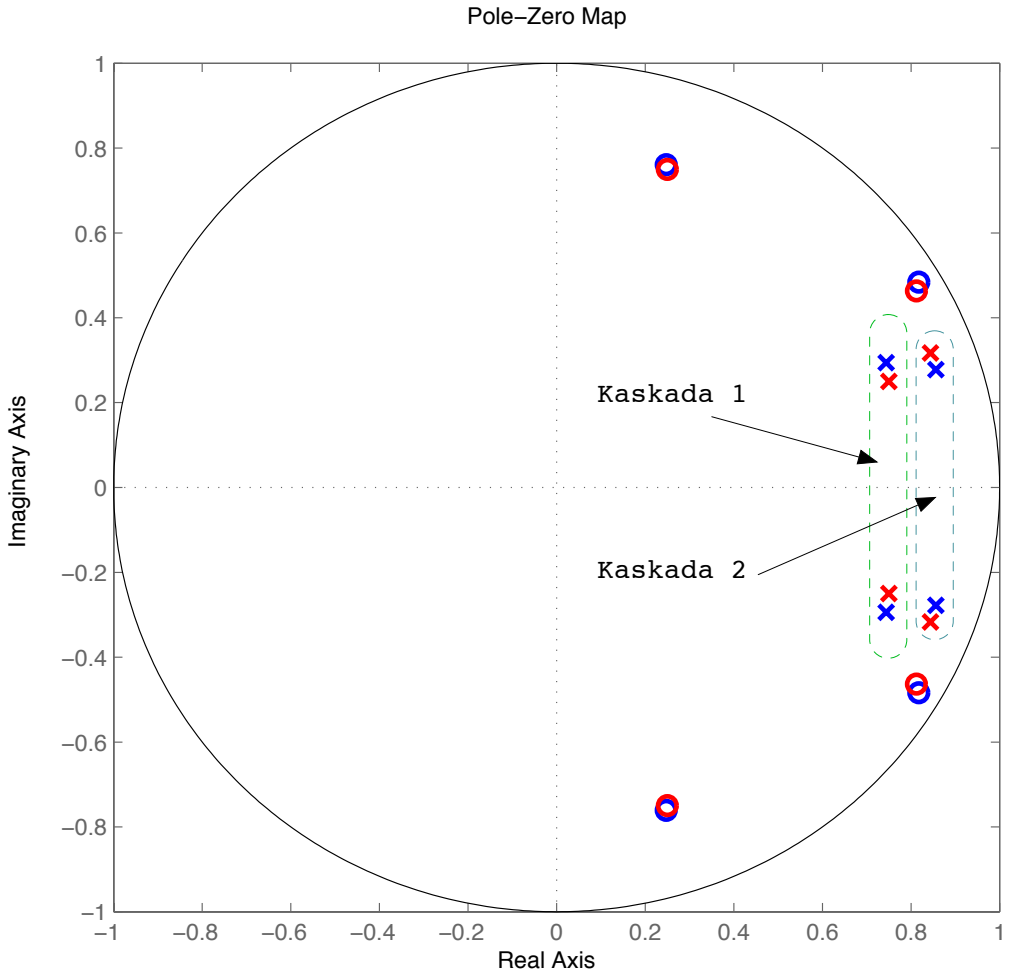
Slika 4: Bodeov dijagram regulatora u direktnoj izvedbi.

Zapis prijenosne funkcije u različitim formama

- Polaznu prijenosnu funkciju moguće je zapisati u serijskoj (kaskadnoj) ili u paralelnoj formi kako slijedi:

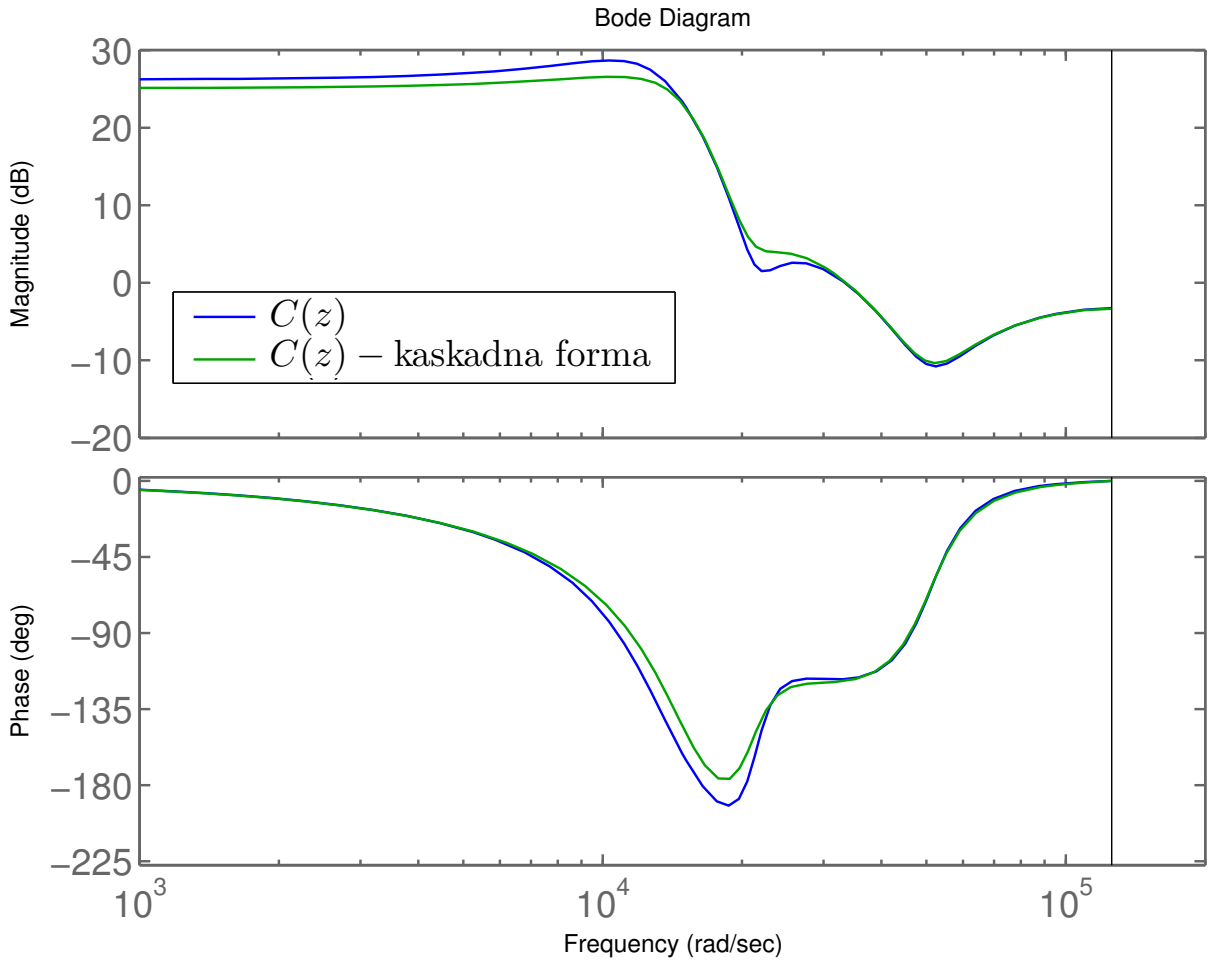
$$\begin{aligned} C(z) &= \frac{z^4 - 2.13z^3 + 2.351z^2 - 1.493z + 0.576}{z^4 - 3.2z^3 + 3.9971z^2 - 2.301z + 0.5184} \\ &= \left(\frac{z^2 - 1.635z + 0.9025}{z^2 - 1.712z + 0.81}\right)\left(\frac{z^2 - 0.4944z + 0.964}{z^2 - 1.488z + 0.64}\right) \\ &= 1 + \frac{-5.396z + 6.302}{z^2 - 1.712z + 0.81} + \frac{6.466z - 4.907}{z^2 - 1.488z + 0.64} \end{aligned} \tag{25}$$

Realizacija u kaskadnoj formi



Slika 5: Položaj polova i nula regulatora u kaskadnoj izvedbi (plava- idealni regulator, crvena - regulator s kvantiziranim koeficijentima.

Realizacija u kaskadnoj formi u aritmetici *Q*8.4



Slika 6: Bodeov dijagram regulatora u kaskadnoj izvedbi.

Usporedba različitih izvedbi regulatora

- Prednost kaskadne i paralelne izvedbe proizlazi iz činjenice da se zasebno realiziraju pojedini segmenti prijenosne funkcije regulatora.
- Svaki od segmenata realizira se u direktnoj izvedbi pa za njih vrijede relacije za osjetljivost polova i nula o iznosu koeficijenata.
- Razlog velike osjetljivosti direktne izvedbe leži u činjenici da su polovi regulatora međusobno blizu.
- Stoga ih je nužno razmjestiti u različite kaskade kako bi se eliminirao njihov međusobni negativni utjecaj.

Izvedba regulatora u modalnoj formi

- Ova izvedba osigurava jednaku (uniformnu) osjetljivost položaja polova i nula.
- Kako bi se to postiglo prijenosna se funkcija pretvara u odgovarajući model po varijablama stanja.
- Koristeći činjenicu da prikaz po varijablama stanja nije jedinstven odabire se tzv. modalna forma zapisa po varijablama stanja kod koje su dijagonalni članovi jednaki realnim polovima ili kvadratne podmatrice dimenzije za konjugirano-kompleksne polove, kao npr.:

$$\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \sigma & \omega \\ 0 & -\omega & \sigma \end{bmatrix} \tag{26}$$

  za slučaj da su polovi regulatora $p_1 = \lambda_1$ i $p_{23} = \sigma \pm j\omega$.
- Kao matrica transformacije u modalnu formu koristi se matrica svojstvenih vektora sustava:

$$x = Tz. \tag{27}$$

Izvedba regulatora u modalnoj formi

- Regulator zapisan u prostoru stanja u modalnoj formi realizira se paralelno, odnosno zaseno se realiziraju pojedini realni polovi i paravi konugirano-kompleksnih polova.
- Pretpostavimo da regulator ima n_r različitih realnih polova i n_c konjugirano kompleksih parova polova, tada se on realizira na sljedeći način:

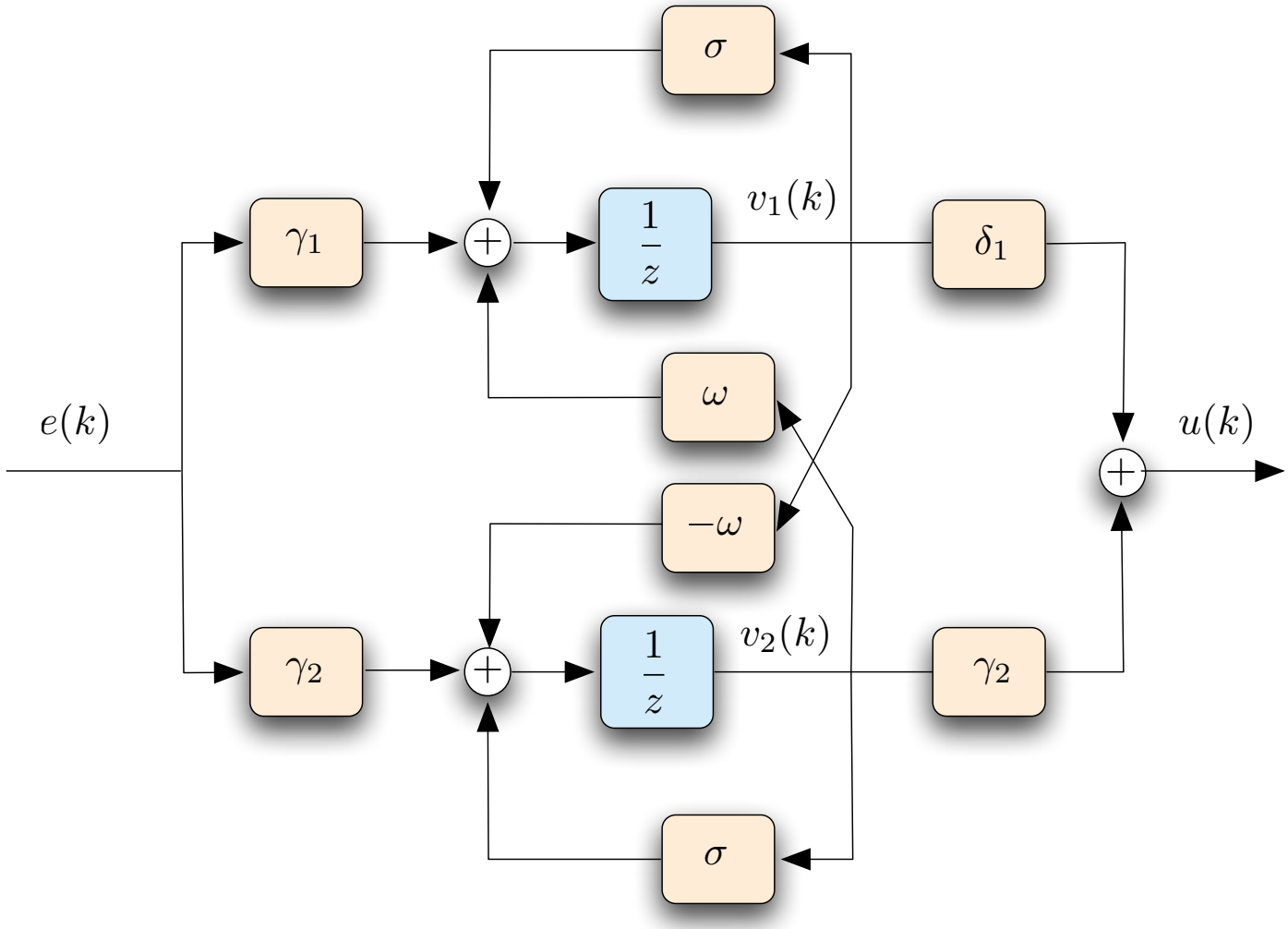
$$z_i(k+1) = \lambda_i z_i(k) + \beta_i e(k), \;\; i = 1, 2, ..., n_r \tag{28}$$

$$v_i(k+1) = \begin{bmatrix} \sigma_i & \omega_i \\ -\omega_i & \sigma_i \end{bmatrix} v_i(k) + \begin{bmatrix} \gamma_{i1} \\ \gamma_{i2} \end{bmatrix} e(k), \;\; i = 1, 2, ..., n_c \tag{29}$$

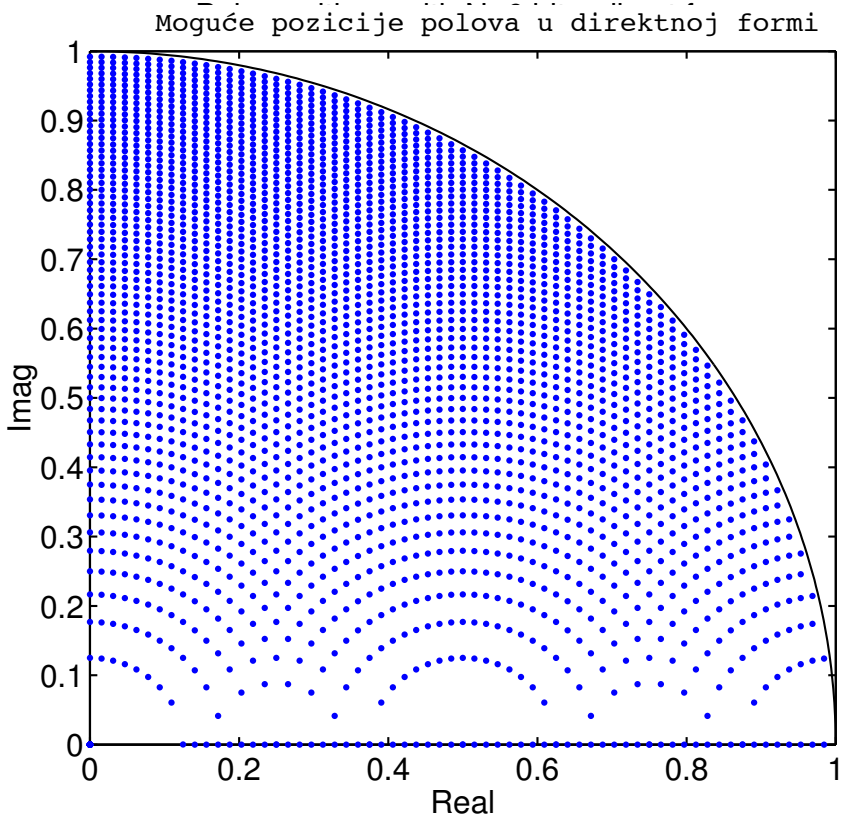
$$u(k) = \sum_{i=1}^{n_r} \gamma_i z_i(k) + \sum_{i=1}^{n_c} \delta_i v_i(k) + De(k) \tag{30}$$

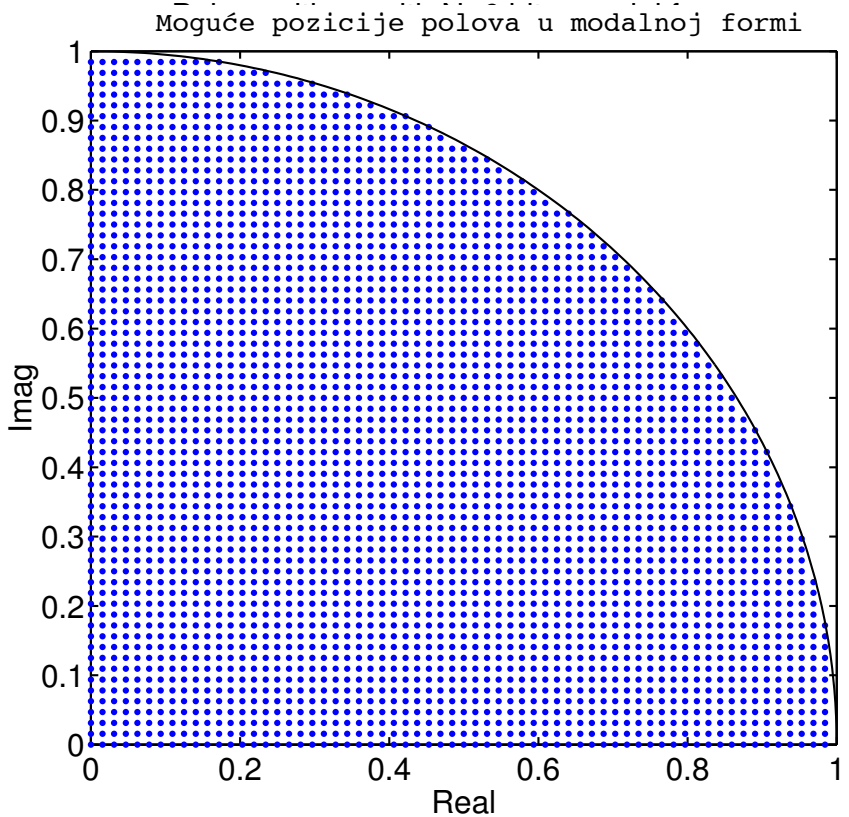
- Uniformna osjetljivost položaja polova postiže se kod modalne realizacije zahvaljujući činjenici da su koeficijenti koji se koriste pri izvedbi regulatora zapravo njegovi polovi.

Realizacija konjugirano-kompleksnog para polova u modalnoj formi



Usporedba mogućih pozicija polova u direktnoj i u modalnoj izvedbi





Primjer: PID regulator

- Za proces:

$$G_s(s) = \frac{K_s}{(1 + T_1 s)(1 + T_2 s)(1 + T_\Sigma s)}, \tag{31}$$

  potrebno je projektirati PID regulator prema tehničkom optimumu postupkom emulacije kontinuiranog regulatora, uzimajući u obzir utjecaj A/D i D/A pretvorbe.

- Kontinuirani PID regulator:

$$G_R(s) = K_R \frac{(1 + T_I s)(1 + T_D s)}{T_I s(1 + T_\nu s)}, \tag{32}$$

  uz $T_I = T_1 = 2$ s i $T_D = T_2 = 1$ s

Primjer: PID regulator - nastavak

$$G_{ZOH} \approx e^{-sT/2} \approx \frac{1}{1+sT/2},$$
$$G_{A/D} \approx e^{-sT/2} \approx \frac{1}{1+sT/2}.$$

U tom slučaju suma nedominantih vremenski konstanti, uz pretpostavku vremena uzorkovanja $T = 50$ ms iznosi:

$$T_\Sigma^* = T_\Sigma + T_\nu + T/2 + T/2 = 0.25 \text{ s}.$$

Pojačanje regulatora određeno je izrazom:

$$K_R = \frac{1}{2K_s}\frac{T_1}{T_\Sigma^*} = 4.$$

Projektiranje za diskretni slučaj T = 50 ms

Diskretizacijom PID regulatora primjenom Tustinovog postupka dobije se sljedeća diskretna prijenosna funkcija PID regulatora:

$$G_R(z) = \frac{33.21z^2 - 63.98z + 30.81}{z^2 - 1.6z + 0.6}$$

Uz pretpostavku implementacije u aritmetici s nepomičnim zarezom Q16.8 dobije se sljedeća prijenosna funkcija:

$$G_{R,FP}(z) = \frac{33.2109375z^2 - 63.98046875z + 30.80859375}{z^2 - 1.6015625z + 0.6015625}$$

Projektiranje za diskretni slučaj T = 10 ms

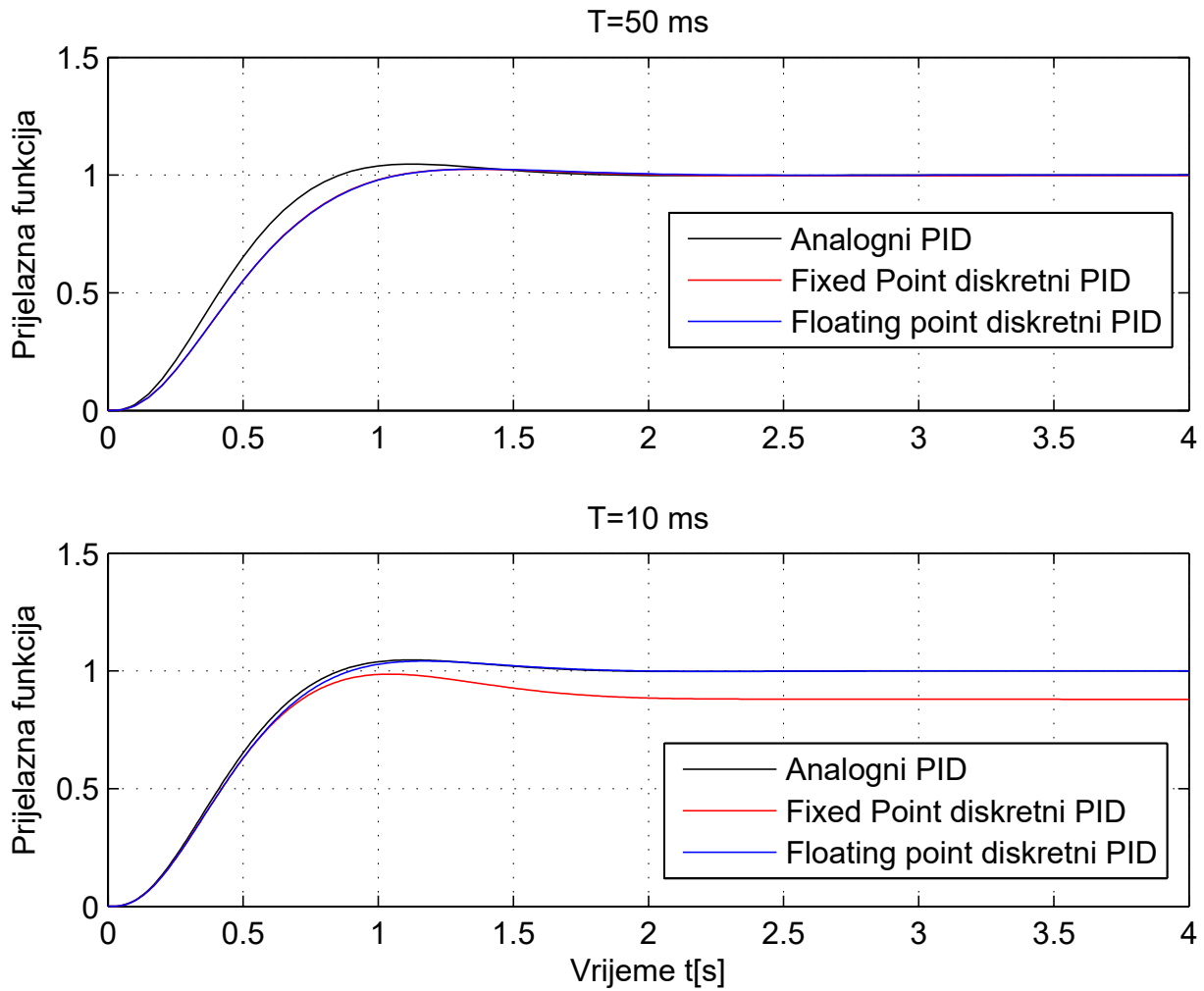
Diskretizacijom PID regulatora primjenom Tustinovog postupka dobije se ijedeća diskretna prijenosna funkcija PID regulatora:

$$G_R(z) = \frac{45.6921768707482 98z^2 - 90.701814058956913z + 45.011904761904759}{z^2 - 1.904751904761905z + 0.904751904761905}$$

odnosno:

$$G_{R,fp}(z) = \frac{45.69140625z^2 - 90.703125z + 45.01171875}{z^2 - 1.90625z + 0.90625}$$

Simulacijski rezultati



Utjecaj vremena uzorkovanja na efekt kvantizacije koeficijenata

- Neka je regulator dan u prostoru stanja izrazom:

$$x(k+1) = \Phi x(k) + \Gamma e(k) \tag{33}$$

$$u(k) = Cx(k) \tag{34}$$

- Pritom je Φ sustavska matrica definirana kao:

$$\Phi = e^{AT}. \tag{35}$$

- Razvidno je da je matrica Φ bliska jediničnoj matrici kada vrijeme diskretizacije T ima vrlo mali iznos:

$$\lim_{T \to 0} \Phi = I. \tag{36}$$

- Ovo povlači da će karakteristične vrijednosti matrice Φ biti smještene vrlo blizu jedinične kružnice te da će osjetljivost na promjenu iznosa koeficijenta biti značajna.

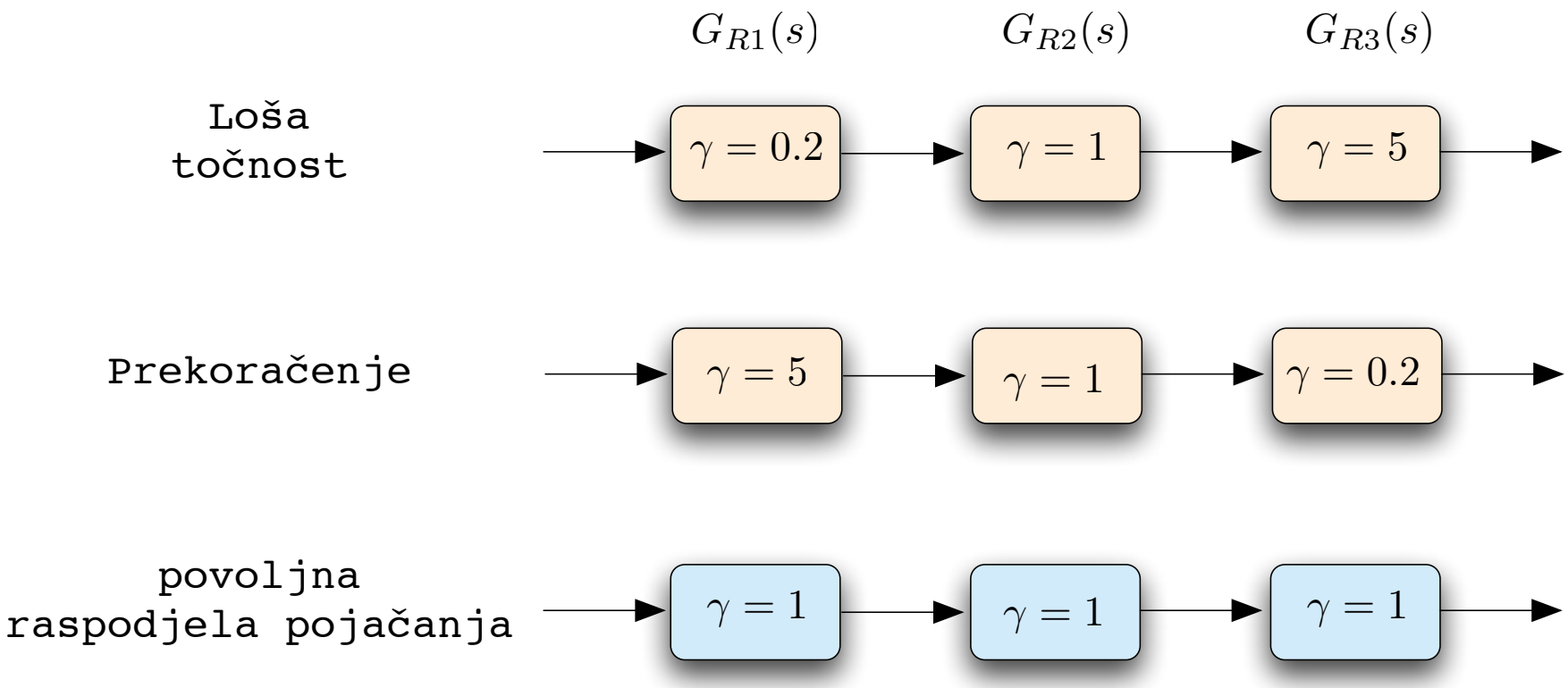
Utjecaj vremena uzorkovanja na efekt kvantizacije koeficijenata (2)

- Iz navedenog je razloga povoljnije regulator zapisati u sljedećoj formi:

$$x(k+1) = (\Phi - I + I)x(k) + \Gamma e(k) = x(k) + (\Phi - I)x(k) + \Gamma e(k), \quad (37)$$

  čime se postiže značajno manja osjetljivost na promjene koeficijenata.
- Ovakav način prikaza diskretnog sustava (regulatora) naziva se δ-formom.

Kako preurediti prijenosnu funkciju regulatora



γ - pojačanje sustava u očekivanom frekvencijskom području ulaznog signala.

- Premali γ problem s točnošću,
- Preveliki γ problem s prekoračenjem