Szervomotor sebességszabályozása

1. A gyakorlat célja

Egyenáramú szervomotor sebességszabályozásának tervezése. A motorszabályozás programvázának felépítése. A sebesség irányítási algoritmus megvalósítása valós időben.

2. Elméleti bevezető

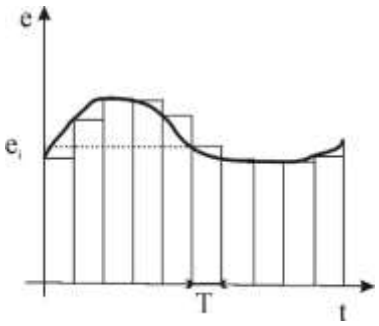
A motor sebességszabályozásának megoldásához PI szabályozót alkalmazunk. A PI szabályozó figyelembe veszi a szabályozási hiba múltbeli alakulását is. A múltbeli hiba összegzett hatását integráló csatornával számítjuk. A folytonos idejű PI szabályozó esetében a beavatkozó jelet az alábbi formában kapjuk:

$$u(t) = K_P \cdot \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau)\, d\tau \right), \quad K_P, T_i > 0 \tag{1}$$

A $T_i>0$ paraméter az integrálási idő.
Az átviteli függvényt az alábbi formában kapjuk:

$$u(s) = K_P \cdot \left(e(s) + \frac{1}{T_i s} e(s) \right)$$

$$H_{PI}(s) = \frac{u(s)}{e(s)} = K_P \cdot \left(1 + \frac{1}{T_i s} \right) \tag{2}$$



1 Ábra: Az integrál megközelítése téglalapokkal

A PI szabályozó kauzális, az integrátor csatorna miatt a pólusa a zéróban van.
A szabályozó mintavételes megvalósításánál az integráló tag megközelítésére a téglalap módszert lehet alkalmazni, az *e(t)* hibafüggvény alatti területet *T* szélességű téglalapokkal

közelítjük meg (lásd 1 Ábra). Az integrál megközelítőleg egyenlő a téglalapok területének összegével a *k*-ik mintavételig:

$$\int_0^T e(t)\,dt = \sum_{i=0}^{k} T \cdot e_i \tag{3}$$

Felhasználva a (3.12) approximációt, a beavatkozó jel számítása a *k*-ik mintavételben:

$$u_k = K_P \cdot \left(e_k + \frac{T}{T_i} \sum_{i=0}^{k} e_i \right) \tag{3.13}$$

Egyszerűbb implementálási forma kapható, ha a beavatkozó jele számítása rekurzívan történik. Felírva a beavatkozó jelet a *k* és *k-1* mintavételben, majd egymásból kivonva:

$$\begin{array}{l} u_k = K_P \cdot \left(e_k + \dfrac{T}{T_i} \displaystyle\sum_{i=0}^{k} e_i \right) \\ \underline{u_{k-1} = K_P \cdot \left(e_{k-1} + \dfrac{T}{T_i} \displaystyle\sum_{i=0}^{k-1} e_i \right)} \quad \Big| - \\ u_k - u_{k-1} = K_P \cdot \left(e_k - e_{k-1} + \dfrac{T}{T_i} e_k \right) \end{array} \tag{4}$$

Így beavatkozó jel a *k*-ik mintavételben:

$$u_k = u_{k-1} + K_P \cdot \left(e_k - e_{k-1} + \frac{T}{T_i} e_k \right) \tag{5}$$

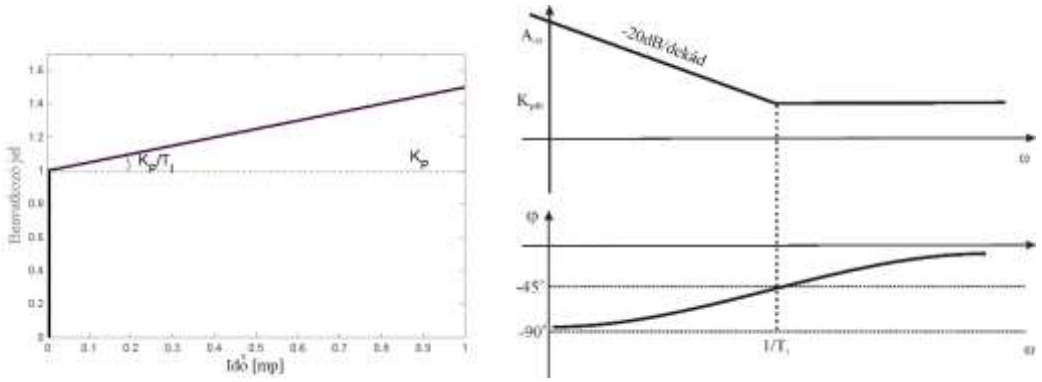
Látszik, hogy a beavatkozó jel értéke mindaddig változni fog, amíg a szabályozási hiba nem válik nullává. Ez jó szabályozóparaméter megválasztás mellett nagy pontosságú szabályozást biztosít.
A mintavételes approximáció átviteli függvénye a *Z* transzformált alkalmazásával számítható:

$$u_k = u_{k-1} + q_0 \cdot e_k + q_1 \cdot e_{k-1} \quad q_0 = K_P\left(1 + \frac{T}{T_i}\right) \quad q_1 = -K_P \tag{6}$$

$$u(z) = u(z)z^{-1} + q_0 \cdot e(z) + q_1 \cdot e(z)z^{-1}$$

$$H_{PI}(z) = \frac{u(z)}{e(z)} = \frac{q_0 z + q_1}{z - 1} \tag{7}$$

A PI szabályozó egységugrásra adott válasza és Bode diagramja a 2. Ábrán látható. Mivel az egységugrás hiba bemenet konstans bármely *t>0* pillanatban, a beavatkozó jel az integráló tag miatt sebességugrás-szerűen nőni fog.



2 Ábra: Az ideális PI szabályozó egységugrásra adott válasza és Bode diagramja

3. A mérés menete

3.1 A program vázának elkészítése

A *MotorControl* tervben építsük fel a program vázát: A motor csak akkor indulhat el, hogyha az *ENABLE CONTROL* kétállású gomb le van nyomva. Az időzítő metódusban valósítsuk meg az alább programstruktúrát:

Mérési adatok beolvasása, kalibrálása (valós idő, frekvencia, pozíció, sebesség)
Ha (ENABLE CONTROL)
    ENABLE 1
    Ha (OPEN LOOP)
        Beolvasott beavatkozó jel kiküldése
    Máskülönben, ha (VELOCTY CONTROL)
        Sebességszabályozás
    Máskülönben, ha (POSITION CONTROL)
        Pozíciószabályozás
    Máskülönven, ha (TRACKING CONTROL)
        Pályakövetés
Máskülönben
    ENABLE 0

3.2 A sebességszabályozás tervezése

Legyen a motor dinamikáját leíró egyenlet:

$$J\dot{\omega} + A_m\omega = K_u u \tag{8}$$

ahol J jelöli a terhelés tehetetlenségi nyomatékát, A_m az elektromágneses állandója, K_u a bemeneti feszültség erősítése.

A motor átviteli függvénye:

$$H_M(s) = \frac{\omega(s)}{u(s)} = \frac{K_u}{Js + A_m} \tag{9}$$

A (9) és (2) összefüggések alapján, a $T_i = J/A_m$ választással a nyílt rendszer:

$$H_N(s) = K_P \frac{T_i s + 1}{T_{is}} \frac{K_u}{Js + A_m} = \frac{K_u K_P}{Js} \tag{10}$$

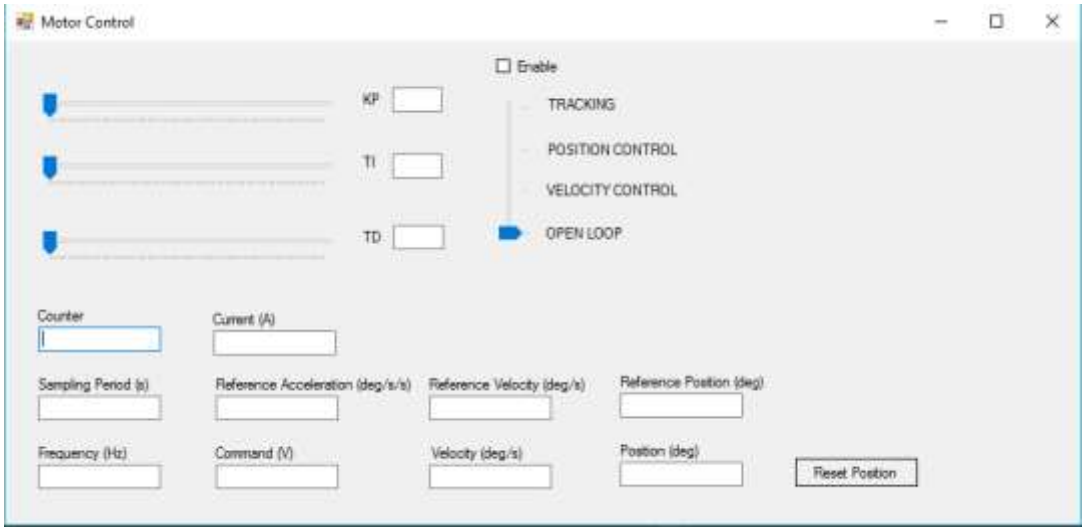
A zárt rendszer:

$$H_o(s) = \frac{H_N(s)}{1 + H_N(s)} = \frac{1}{\frac{J}{K_u K_P} s + 1} \tag{11}$$

Látszik, hogy minél nagyobb a szabályozó erősítése a rendszer válasza annál gyorsabb. Ugyanakkor az integrátor miatt egységugrásra garantált a nulla állandósult állapotbeli hiba.

3.3. A sebességszabályozás megvalósítása

A sebességszabályozáshoz az előírt sebességet a DESIRED VELOCITY ablakelemből olvassuk be radián/másodperc mértékegységben. A proporcionális erősítést és az integrálás időt a *KP* és *TI* csúszkákról olvassuk le a *getTextBoxVal(string box, out int val)* függvénnyel. A *TI* csúszkáról leolvasott értéket el kell osztani 1e5-tel, a KP értékét pedig 100-zal (lásd 3 Ábra).



3. Ábra: A program interfésze

Az implementáláshoz először a sebességszabályozási hibát számítjuk ki: $e_k=w_{ref}-w_k$. A beavatkozó jel kiszámításához az (5) összefüggést alkalmazzuk.
A beavatkozó jel kiküldésénél annak értékét korlátozni kell ±10 V közé.
A kiküldés az alábbi módon történik:

- Kiküldjük a beavatkozó jel abszolút értékét az NI USB-6001 kártya feszültségkimenetén.
- A beavatkozó jel előjelének függvényében kiküldjük a motornak az irány bitet (0-s kimeneti port 0-s bitje). Pozitív beavatkozó jel esetén 1-et, negatív beavatkozó jel esetén 0-t küldünk ki.

4. Kérdések és feladatok

1. Határozza meg a PI szabályozó implementálási formáját, ha a mintavételes megvalósításhoz a trapéz módszert alkalmazzuk.
2. Módosítsa a programot úgy, hogy a sebességértéket nem radiánban, hanem fokban számoljuk. Mennyi az így elérhető pontosság?