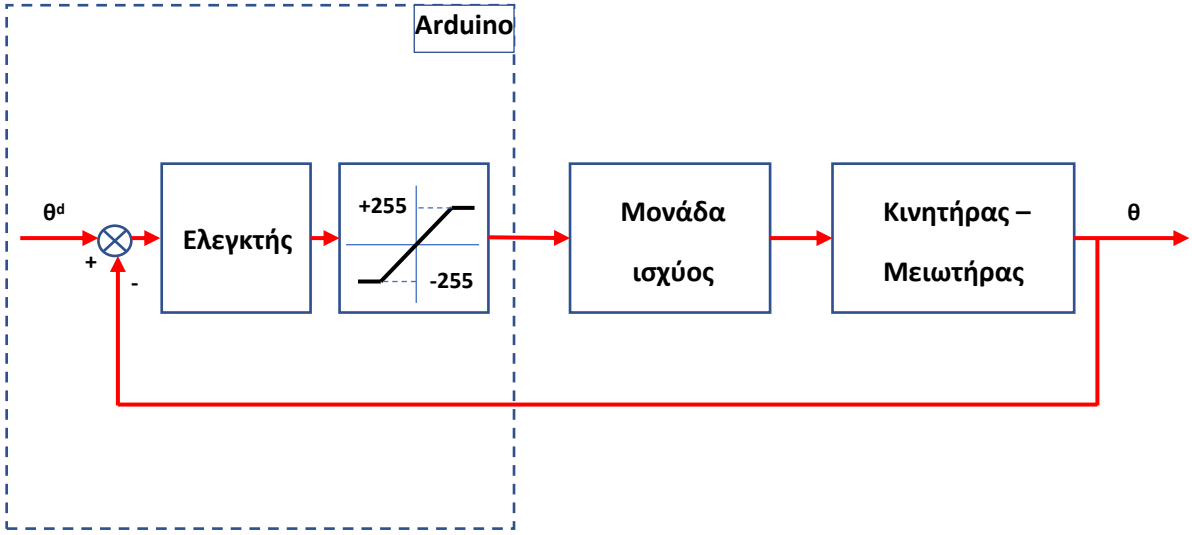
Εργαστηριακή Άσκηση 4

Έλεγχος γωνίας στροφής κινητήρα DC με ενσωματωμένο μειωτήρα

Υλοποίηση Αναλογικού – Διαφορικού Ελέγχου (PD)

Στο Σχήμα που ακολουθεί φαίνεται το σύστημα του εργαστηρίου σε λειτουργία ελέγχου γωνίας.



Θα προσπαθήσομε στην συνέχεια να υλοποιήσομε με την βοήθεια του μικροελεγκτή Arduino Uno κατάλληλους «Ελεγκτές», ώστε να ελέγξομε την γωνία στροφής όσο γίνεται καλύτερα.

Αναλογικός – Διαφορικός ελεγκτής (Proportional - Derivative Control)

Ο Διαφορικός έλεγχος συνεισφέρει στην ευστάθεια του συστήματος. Σε συνδυασμό με τον πάντα παρόντα Αναλογικό, αρκεί για να ελέγξει σωστά ένα σερβοσύστημα, όταν αυτό δεν επηρεάζεται από διαταραχές ροπής.

Η εντολή :

Derivative=(Err-PrevErr)/0.01;

είναι μια καλή αριθμητική προσέγγιση της παραγώγου του σφάλματος – όταν ο "χρόνος δειγματοληψίας και ελέγχου" είναι 0.01 sec.

Η εντολή ελέγχου είναι τώρα :

cmd=Kp*(Err) + Kd*Derivative;

Ρύθμιση του PD Ελεγκτή

Από την θεωρία γνωρίζομε ότι η συνάρτηση μεταφοράς κλειστού βρόχου, με τον PD ελεγκτή είναι :

$$G_c(s) = \frac{\frac{(K_p + K_d\, s)K_{tot}}{T}}{s^2 + (\frac{K_d K_{tot} + 1}{T})s + \frac{K_p K_{tot}}{T}}$$

Παρά το ότι η συνάρτηση αυτή δεν είναι η τυπική συνάρτηση μεταφοράς ενός δευτεροταξίου συστήματος (Βλέπε Θεωρητική Άσκηση), θα χρησιμοποιήσομε τους τύπους που γνωρίζομε – όπως ακριβώς κάναμε και στην περίπτωση του PI ελεγκτή στο σύστημα ελέγχου ταχύτητας:

$$\omega_n^2 = \frac{K_p K_{tot}}{T} \qquad \kappa\alpha\iota \quad 2\zeta\omega_n = \frac{K_d K_{tot} + 1}{T}$$

ώστε από τις «προδιαγραφές» για την κυκλική συχνότητα ω_n και την απόσβεση ζ που θα θέσομε, να υπολογίσομε στη συνέχεια τα κέρδη Kp και Ki.

Η κυκλική συχνότητα σχετίζεται άμεσα με την «ταχύτητα απόκρισης» του συστήματος. Η τελευταία, εκτιμάται με την βοήθεια του «χρόνου ανόδου» (rise time, Tr). Για τον χρόνο ανόδου όμως, στην περίπτωση των δευτεροτάξιων συστημάτων, δεν διαθέτομε αναλυτική σχέση. Θα χρησιμοποιήσομε εναλλακτικά τον χρόνο κορυφής, Tp, για τον οποίο (και πάλι για το τυπικό δευτεροτάξιο σύστημα με υποαπόσβεση) ξέρομε :

$$T_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}}$$

Θέσετε προδιαγραφές. Π.χ :

Tp=0.1 sec (σημαίνει ότι το σύστημα σε 0.1 sec θα έχει πιάσει την υψηλότερη τιμή του, αρκετά γρήγορα δηλαδή)

ζ = 0.6 (μια αξιοπρεπής απόσβεση)

και υπολογίσετε στην συνέχεια, με βάση τους παραπάνω τύπους και τα χαρακτηριστικά της εγκατάστασης που βρήκατε κατά την ταυτοποίηση (δηλαδή Ktot και T), τα κέρδη Kp και Kd του ελεγκτή.

Δώστε τις τιμές αυτές στο πρόγραμμα στον Arduino και δείτε το αποτέλεσμα σε είσοδο βαθμίδας 200 παλμούς.

Υπολογίσετε εκ νέου Kp και Kd τέτοια που να δίνουν τον ίδιο χρόνο κορυφής αλλά λόγο απόσβεσης ζ=0.2 και παρατηρείστε ότι το σύστημα είναι τώρα πολύ ταλαντωτικό.

Και για τις δύο παραπάνω περιπτώσεις, σχεδιάσετε την καμπύλη απόκρισης της γωνιακής θέσης σε είσοδο βαθμίδας 200 παλμούς.

Διαταραχές ροπής

Προσπαθήστε να εκτρέψετε τον άξονα από την θέση ισορροπίας του. Τι παρατηρείτε;

Σφάλμα μόνιμης κατάστασης

Ισορροπεί το σύστημα στην θέση που του ζητούμε όταν δεν εξασκείται σ' αυτό εξωτερική ροπή; Αν όχι, ποιος είναι ο λόγος; Η θεωρία προβλέπει για την περίπτωση αυτή ότι το σφάλμα έπρεπε να είναι μηδέν.

Κώδικας

```
#include <Encoder.h>

Encoder Enc(2, 3);

volatile long pos =0, start_pos=0, set_point=0;

long  newP, vel, PrevErr=0, Err=0, cmd=0;

float Integral=0, Derivative=0, Kp=2.8, Kd=0.24, Ki=0;


void setup() {

//Setup timer

cli();                    //stop interrupts

TCCR1A = 0;                // set entire TCCR1A register to 0

 TCCR1B = 0;               // same for TCCR1B

 TCNT1  = 0;               //initialize counter value to 0;

 // set timer count for 100 Hz increments

 OCR1A = 19990;

 // turn on CTC mode

 TCCR1B |= (1 << WGM12);

 // Set CS11 bit for 8 prescaler

 TCCR1B |= (1 << CS11);

 // enable timer compare interrupt

 TIMSK1 |= (1 << OCIE1A);

 sei();//allow interrupts

 //END TIMER SETUP


Serial.begin(9600);

pinMode(13, OUTPUT);            //DIRECTION Motor Channel B

pinMode(8, OUTPUT);             // BRAKE Motor Channel B

digitalWrite(8, LOW);           //Disengage the Brake for Channel B
```

```
pos=0; vel=0;
Enc.write(0);
}

ISR(TIMER1_COMPA_vect)          //Interrupt at freq of 100 Hz to execute control law
{
//Read encoder and calculate ...
  pos = Enc.read();
  Err=set_point-pos;
  Integral=Integral+(Err+PrevErr)*0.005;
  Derivative=(Err-PrevErr)/0.01;
  PrevErr=Err;

//Compute control law
cmd=Kp*(Err)+Ki*Integral+Kd*Derivative;

//Issue command
if (cmd<0)  {cmd=-cmd; digitalWrite(13, LOW);}
        else {digitalWrite(13, HIGH);}
if (cmd>255)  cmd=255;
analogWrite(11, cmd);

  Serial.println (pos);
}
void loop()
{
 set_point=300;         // κινήσου στην θέση 300
 delay(5000);
 set_point=0;           // κινήσου στην θέση 0
 delay(5000);
}
```