Univerza v Ljubljani
Fakulteta za elektrotehniko

Seminarska naloga pri predmetu Gradniki v tehnologiji vodenja

Meritev količine vode v posodi z LSM303DLHC

Kristjan Šoln

2. letnik

Aplikativna elektrotehnika - Avtomatika

Vpisna številka: 64190321

Kozje, december 2020

Povzetek

V tej seminarski nalogi sem se lotil izdelave improviziranega merilnika količine vode v posodi z merilnikom magnetnega polja LSM303DLHC. Merilnik izkorišča prožnost kovinskega traku. Plastenko obesimo na sredino traka, ki se malo ukrivi. Spreminjanje razdalje med trakom in senzorjem izmerimo s pomočjo magneta, ki je pritrjen na trak. Preko izmerjene karakteristike sistema mikrokrmilnik predvidi maso vode v plastenki. Pri tem s povprečenjem odstrani šum na signalu. V najslabšem primeru ima merilnik ±56 gramov odstopanja pri merilnem območju 1500 gramov.

Kazalo vsebine

Kazalo slik

1. Uvod

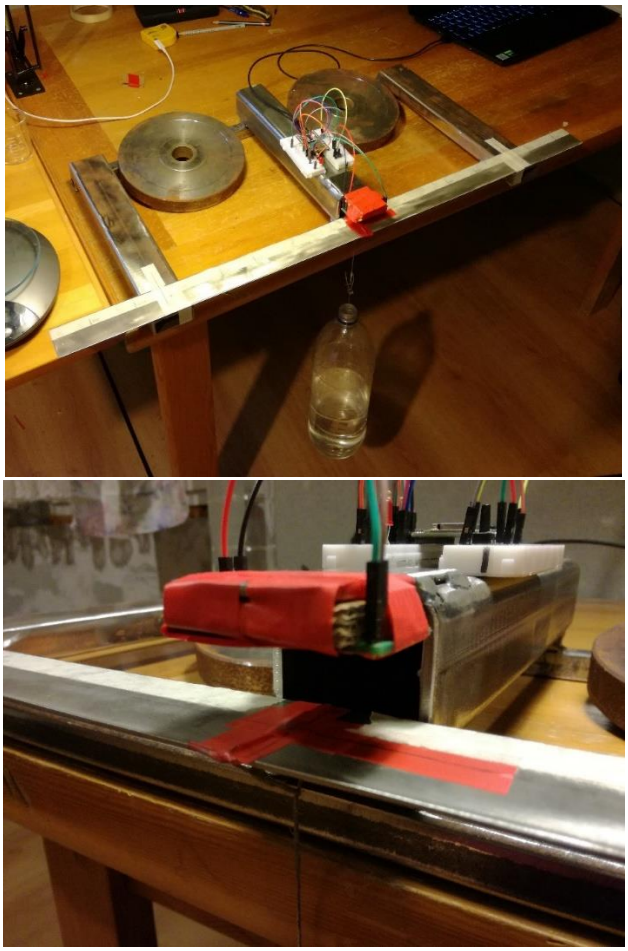
Glavni cilj te seminarske naloge je izdelava improviziranega merilnika količine vode v posodi. Uporabiti sem moral čim več materialov, ki sem jih imel že doma. Izjema je merilnik in mikrokrmilnik; uporabil sem LSM303DLHC. To je MEMS senzor, ki združuje funkcionalnost acelerometra in magnetometra.

Na začetku sem si postavil nekaj zahtev. Želel sem čim boljšo natančnost merilnika. Prav tako sem želel, da so rezultati ponovljivi, torej delovanje brez povzročanja trajnih deformacij merilnika. Želel sem uporabiti princip upogibanja kovinskega traku, enak kot pri četrti laboratorijski vaji. Ker sem pričakoval nelinearen odziv merilnika, sem želel za konec izvesti čim boljšo kompenzacijo nelinearnosti z lookup tabelo ter linearno interpolacijo med posameznimi točkami.

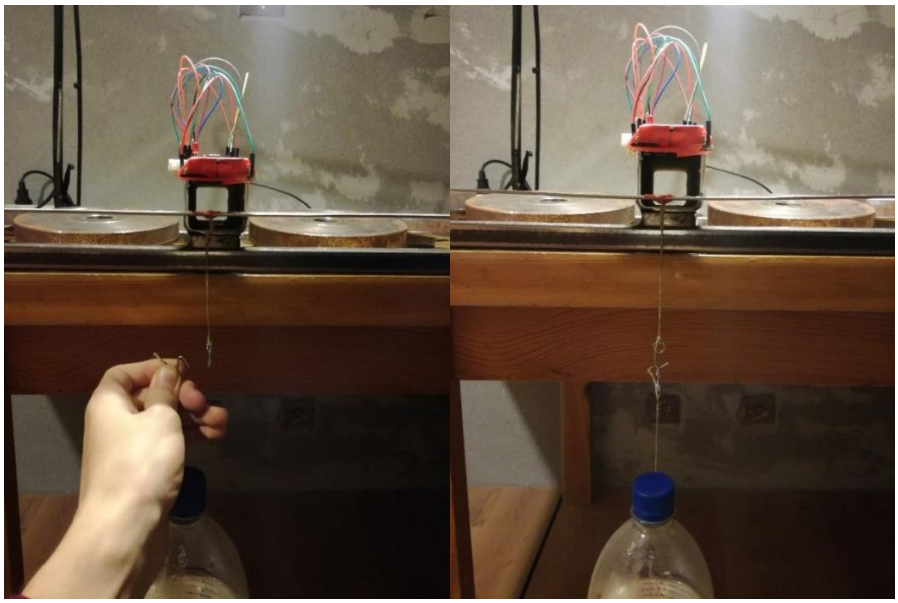
2. Jedro

2.1 Opis rešitve

Merilnik izkorišča upogib kovinskega traku, magnet ter magnetometer, da predvidi, koliko vode je v plastenki.



Slika 1: Levo: Končni izdelek, obremenjen s plastenko vode. Desno: Magnetni senzor, magnet ter vrvica na kovinskem traku. Magnet je skrit pod rdečim izolirnim trakom.



Slika 2: Merilnik pred in po obremenitvi. Vidi se tudi upogib traku.

Na sredino kovinskega traku je privezana plastenka, v kateri je voda. Voda zaradi sile teže povzroči upogib v prožnem traku iz nerjavečega jekla. Nad vrvico, na kovinskem traku je Neodim magnet z izraženim parom polov. Na sliki se ga ne vidi, saj je skrit pod rdečim izolirnim trakom. Nad samim trakom je postavljen senzor. Ta je podložen s kartonom, zaradi nastavljanja razdalje med magnetom in senzorjem.

Ko se trak upogne, se poveča tudi razdalja med magnetom in senzorjem. To senzor zazna kot spremembo gostote magnetnega polja v osi Z. Mikrokrmilnik zaznano vrednost gostote magnetnega polja pretvori v predvideno maso vode v plastenki in jo izpiše na serijski vmesnik.

Izdelal sem kovinsko ogrodje, ki služi kot nosilec za trak ter senzor. Na sliki so vidne tudi uteži, s katerimi sem preprečil, da bi se prekucnil čez mizo.

Dodal sem še gumb za tariranje, ta trenutno vrednost na senzorju postavi kot referenco za 0 gramov. Prav tako sem dodal še svetlečo diodo, ki je bila koristna pri razvijanju programa in merjenju statične karakteristike merilnika.

Program sem napisal v programskem okolju VSCode in PlatformIO. VSCode je veliko bolj fleksibilen kot Arduino IDE, za programiranje mikrokrmilnikov pa sem potreboval tudi PlatformIO. Ta omogoča nalaganje kode, serijski vmesnik, podporo za Arduino funkcije in podobno. Za organizacijo kode sem uporabljal Git ter GitHub. Tako je celoten programski del projekta dostopen na spletnem naslovu https://github.com/kristjansoln/Merilnik_LSM303DLHC. V veji *Characteristics-measurement-code* je program za merjenje karakteristike, v veji *Quintic-regression-approximation* pa končen program za izpisovanje količine vode v posodi.

Uporabil sem knjižnico Adafruit_LSM303DLH_Mag.h, ki je dostopna na portalu GitHub. Za dodatno nastavljanje parametrov senzorja sem se moral poglobiti v funkcije v knjižnici ter v podatkovni list senzorja, da sem vedel, katere registre moram nastavljati.

Za maksimalno obremenitev sem izbral vrednost 1500 gramov, saj le ta pri izbrani dolžini traku povzroči okoli 8 milimetrov upogiba. Večji upogib bi lahko povzročil trajno deformacijo, zato je 1500 gramov zaenkrat maksimalna obremenitev.

Nelinearno karakteristiko sem poskušal popraviti z lookup tabelo izmerjenih vrednosti in linearno interpolacijo med točkami, a sem se kasneje odločil za metodo aproksimacije s polinomom pete stopnje.

Pomembno je bilo tudi odstranjevanje šuma, ki sem ga dosegel z ustreznim povprečenjem signala.

Meritev in izračun poteka po naslednjem načinu (opis delovanja kode v veji *Quintic-regression-approximation*):

- Mikrokrmilnik s frekvenco 200 Hz izvede 400 meritev magnetnega polja. Ob tem izpusti prvih 15 meritev ter ponovno izvede vsako neveljavno meritev (torej tisto, ki je izmerila 0,00 uT). Izračuna aritmetično sredino in jo vrne v *main* funkcijo. To se zgodi v funkciji *collectSamplesAndMean(400)* in traja približno dve sekundi. Tako se znebimo šuma.
- Preko enačbe za aproksimacijo karakteristike izračunamo predvideno maso vode v plastenki.
- Na serijski vmesnik izpišemo maso vode, kateri odštejemo taro. Privzeta vrednost tare je 0.
- Če je pritisnjena tipka, se trenutna vrednost zapiše v taro, ki je upoštevana v naslednjem merilnem ciklu.

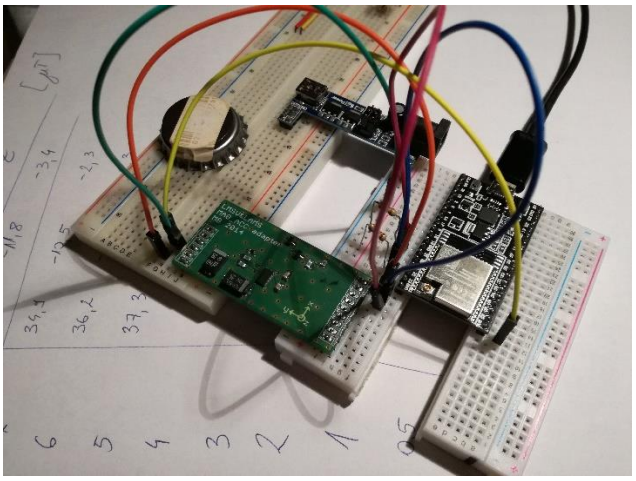
2.2 Opis postopka načrtovanja, izgradnje ter testiranja fizičnega dela merilnika

Začel sem z minimumom. Poskusil sem usposobiti senzor in ustrezno knjižnico. Povzročala je težave, prenos je delal le nekaj trenutkov, ali pa sploh ne. Tudi *IIC scanner* marsikdaj ni zaznal senzorja. Ko sem še enkrat preveril knjižnico, sem opazil, da obstaja novejša. Po posodobitvi knjižnice je senzor delal, kot se spodobi. Na SCL in SDA liniji sta bila potrebna še pullup upora, da je zadeva delala popolnoma. Motil me je PCB okoli senzorja, saj je imel veliko nepotrebnih kontaktov, prav tako nima nobene luknje za lažjo montažo na okvir merilnika. Kasneje sem tu improviziral z izolirnim trakom in kartonom.

Uporabil sem magnet iz hladilnika, kot je vidno na spodnji sliki. Izkazalo se je, da ni primeren, saj ima posebno zasnovo, imenovano Halbach magnetni sklop. Magnet ima mnogo parov polov, jaz pa sem želel le en izražen par polov. Zato sem uporabil raje Neodim magnet, ki sem ga prav tako našel na hladilniku.

Magnetno polje z razdaljo hitro upada, zato sem želel, da bi senzor in magnet postavil čim bližje skupaj. Kasneje sem ugotovil, da potrebujem minimalno razdaljo okoli 20 mm, saj drugače senzor pride v zasičenje. To je bilo pomembno pri načrtovanju ogrodja. Merilno območje je namreč lahko največ 810 uT, v praksi pa se lahko to razširi do nekje 950 uT, Neodim magnet pa hitro preseže to mejo.

Slika 3: Levo: Magnet s Halbachovim magnetnim sklopom. Sredina in desno: Neodim magnet. Posuti železovi opilki prikazujejo obliko magnetnega polja.

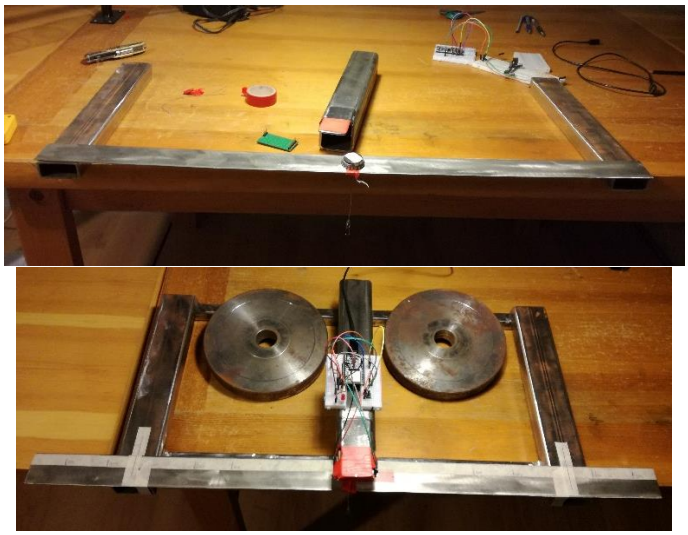


Slika 4: Prvo testiranje senzorja.

Nato sem se posvetil iskanju kovinskega traku. Želel sem daljšega, saj bi lahko s spreminjanjem razdalje med nosilcema traku nastavljal prožnost traku in posledično nastavljal merilno območje. Odkril sem, da lahko pričakujem največ 15 mm upogiba pri 1 kg, kar je dovolj, da merilnik dobro zazna spremembo razdalje.

Slika 5: Iskanje in preverjanje ustreznosti kovinskega traku

Za tem sem izdelal prototip okvirja merilnika. Po določitvi razdalje med nosilcema traku sem dodal prečne cevi, ki so celotno zadevo fiksirale. Ker nisem želel merilnika vijačiti na mizo, sem dodal še dve uteži, da se težje premika.



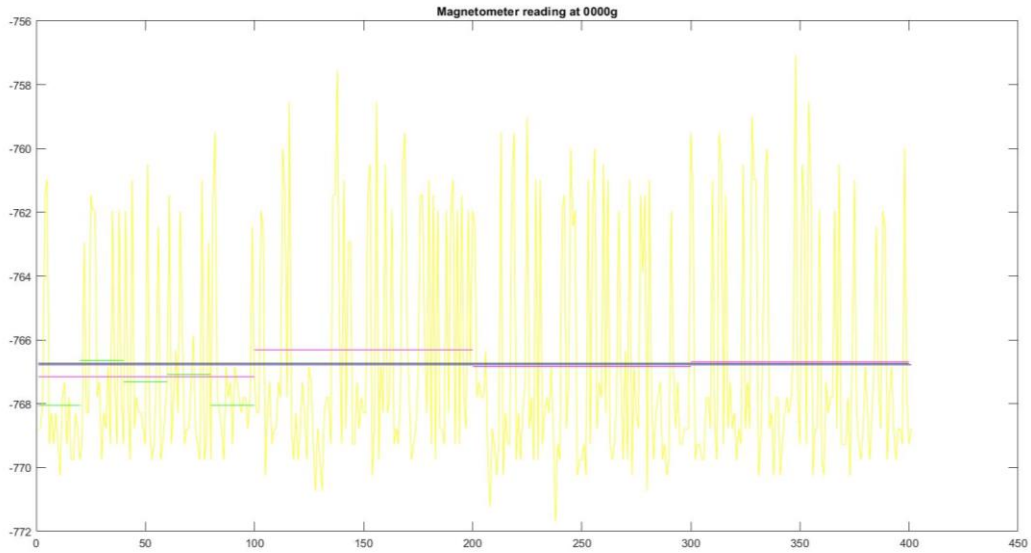
Slika 6: Prototip in končni okvir merilnika.

Dolžino traku in razdaljo med levim in desnim nosilcem sem nastavil na 490 milimetrov, kar pri obremenitvi 1500 gramov povzroči 8 milimetrov upogiba. Po tem, ko senzor ustrezno podložil in prilepil na okvir, sem dobil za območje obremenitve 1500 gramov na senzorju vrednosti med -850 uT in -530 uT. Sklenil sem, da so to uporabne številke in dokončno fiksiral senzor ter kovinski trak.

2.3 Opis razvoja programskega dela

Ko sem imel končano okvir ter vse na svojem mestu, sem se lotil programiranja.

Na merilniku je bil prisoten opazen šum, ki sem ga prikazal na spodnji sliki z rumeno črto. Znebil sem se ga tako, da sem na meritev zajel 400 vzorcev in jih povprečil. Na spodnjem grafu so prikazane tudi povprečne vrednosti na posameznih intervalih. Povprečenje na 200 vzorcev ne odstopa precej od povprečenja na 400 vzorcev. Na grafu je to označeno kot črna in modra črta in vidi se, da ležita skoraj ena na drugi. Tako sem vedel, da je 400 vzorcev več kot dovolj za zanesljivo odstranitev šuma.



Slika 7: Graf šuma na merilniku in povprečnih vrednosti signala na različnih intervalih

Napisal sem funkcijo mag.init(), kjer inicializiram senzor, nastavim avtomatsko ojačenje senzorja (nastavi se na največje merilno območje) in frekvenco osveževanja senzorja na najvišjo možno, 220 Hz.

Spisal sem funkcijo *collectSamplesAndMean(int sampleCount),* ki je na voljo za ogled v prilogi oziroma na GitHub-u, nahaja se v vseh vejah projekta. Omogoča zanesljivo merjenje magnetnega polja. Najprej naredi 15 meritev, ki jih zavrže. Izkazalo se je, da so prve meritve nezanesljive. Nato opravi 400 veljavnih meritev. Če meritev ni veljavna (prebere vrednost 0,00 uT), se jo ponovi. Na koncu najde aritmetično sredino teh 400 meritev in jo vrne v *return* stavku. Bere s frekvenco 200 Hz, kar omogoči izvedbo meritve v 2 sekundah.

Za merjenje karakteristike sem uporabil kodo v veji *Characteristics-Measurement-Code,* na voljo v prilogi in na portalu GitHub. Dodal sem le tipko, ki izvede funkcijo *collectSamplesAndMean,* ter svetlečo diodo kot indikacijo izvajanja meritev. Izmeril sem vrednost senzorja za območje 1500 gramov, v korakih po 100 gramov, ter izdelal karakteristiko senzorja. Kasneje sem jo aproksimiral s pomočjo polinoma pete stopnje. To je natančneje opisano pod točko 2.4.

Za meritev in pretvorbo v maso, torej končno delovanje merilnika, sem uporabil kodo, shranjeno v veji *Quintic-regression-approximation,* na voljo v prilogi in na portalu GitHub. V vsakem ciklu funkcije *void loop()* krmilnik iz magnetnega polja preko aproksimacije izračuna maso vode v plastenki. Dodal sem še funkcijo tariranja, ki se nahaja v *void loop().*

2.4 Merjenje karakteristike in odpravljanje nelinearnosti

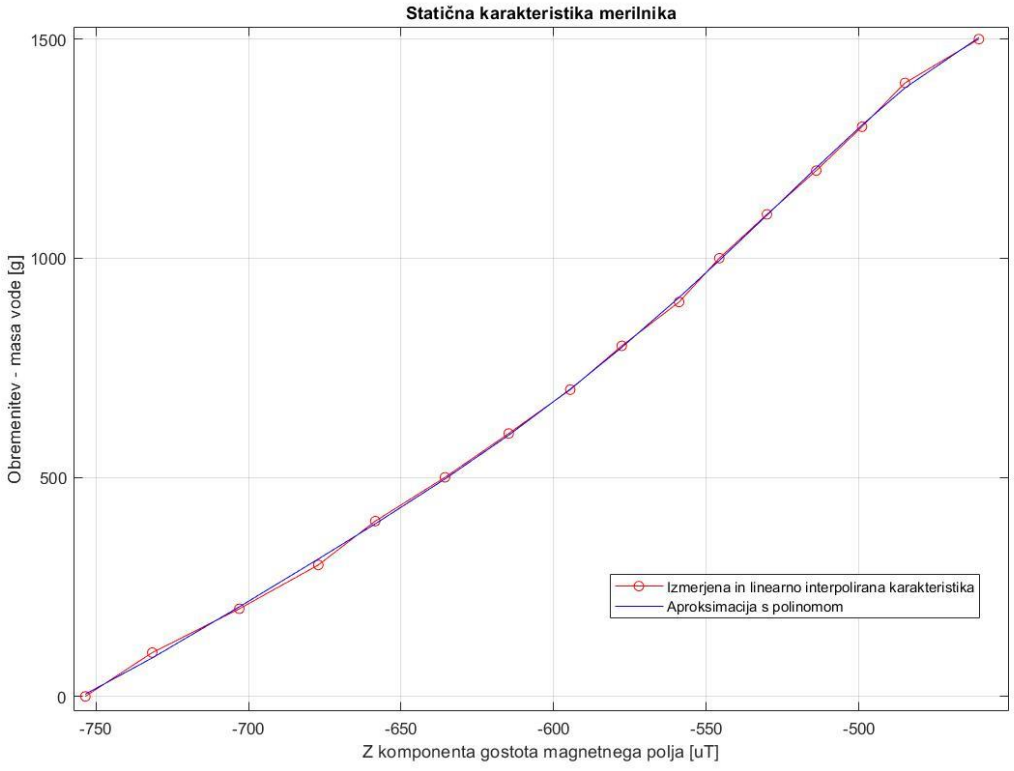
Izmeril sem karakteristiko merilnika na območju 1500 gramov s korakom 100 gramov. Zavoljo preprostosti sem namesto linearne interpolacije med točkami raje odločil za aproksimacijo s polinomom pete stopnje, ki sem ga izračunal s pomočjo spletnega orodja MyCurveFit, dostopnega na www.MyCurveFit.com. Nameraval sem uporabiti Matlab in Curve Fitting Toolbox, vendar mi je spletno orodje dalo boljši rezultat na preprostejši način. Problem je ležal v tem, da nisem dobro razumel vseh opcij v Matlabovem orodju. Na spodnji sliki je izmerjena karakteristika, izrisana v Matlabu, ter enačba, ki aproksimira to karakteristiko. Navedena je tudi tabela izmerjenih vrednosti, ki sem jo uporabljal za izhodišče pri risanju karakteristike.

Format:

masa vode [g], z-komponenta magnetnega polja [uT];

0000, -767.73; *// brez plastenke*
0000, -753.41; *// s plastenko*
0100, -731.50;
0200, -702.98;
0300, -677.10;
0400, -658.47;
0500, -635.62;
0600, -614.77;
0700, -594.60;
0800, -577.74;
0900, -558.97;
1000, -545.80;
1100, -530.19;
1200, -514.05;
1300, -499.07;
1400, -484.98;
1500, -460.79;

$$m(B_z)\,[g] = -154785.2 - 1288.846 * B_z - 4.17052 * B_z^2 - 0.006647351 * B_z^3 - 0.000005248848 * B_z^4 - 1.645575e-9 * B_z^5$$



Slika 8: Graf karakteristike merilnika in njegove linearne aproksimacije

2.5 Rezultati in komentar postopkov

Izpostavil bi rad občutljivost merilnika. V tabeli karakteristike senzorja se vidi, da sama plastenka spremeni izmerjeno vrednost za 14.3 uT, kar me je zelo presenetilo. Plastenka tehta 56 gramov. Merilnik je veliko bolj občutljiv, kot sem pričakoval. V povprečju sprememba za 100 gramov pomeni spremembo za 19.53 uT na senzorju.

Motnje zaradi šuma na senzorju so, po mojem mnenju, po povprečenju 400 vzorcev, zanemarljivo majhne, saj se večkratne meritve ujemajo do dveh decimalk natančno.

Resolucija merilnika magnetnega polja je, glede na podatkovni list senzorja, enaka 0.2 uT. Tudi to je po mojem mnenju zanemarljivo majhna negotovost.

Problem pri tako veliki natančnosti so majhni neželeni premiki prožnega kovinskega traku. To povzroči, da magnet ni več ustrezno poravnan z Z-osjo merilnika, kar močno spremeni rezultat. Eksperimentalno sem ugotovil, da lahko trak ročno popravim tako, da je začetna vrednost enaka ± 2 uT začetne vrednosti v karakteristiki.

Pri merjenju karakteristike senzorja sem si pomagal s kuhinjsko tehtnico, ki ima ločljivost enako 1 gram. To pomeni, da sem referenčno količino vode lahko nastavil do ± 1 gram natančno.

Velik pogrešek na nekaterih mestih povzroča aproksimacija s polinomom**.** Četudi je senzor zelo občutljiv, ta aproksimacija povzroči nenatančno preslikavo v nelinearno karakteristiko. Če senzor izmeri -460 uT, se to pretvori v 1506 g, kar je zelo blizu dejanske vrednosti. Ko izmeri -677 uT, izračuna 314 g, kar je 14 g preveč. Če pogledamo karakteristiko, izgleda, da je največji pogrešek ravno pri 300 g, zato lahko rečem, da aproksimacija povzroči največ ±15 g odstopanja.

Največji pogrešek predstavlja nenatančna karakteristika oziroma težka ponovljivost rezultatov zaradi velike občutljivosti merilnika na položaj kovinskega traku in neznatnih premikov med merjenjem. Neobremenjen senzor vedno pokaže okoli -767 uT s ±3 uT odstopanja, kar je pravilno in razmeroma natančno. Ko pa ga obremenim s 1500 g, izmeri med 465 in 470 uT, namesto željenih 461 uT, kot piše v karakteristiki. Zato pri 1500 g merilnik pokaže odstopanje do ±40 g. Pri meritvi 1400 g pokaže -480,25 uT in 1414,78 g, namesto 484 uT in 1400 g. Pri obremenitvi 1200 g izmeri -511,37 uT in 1224,10 g, namesto -514 uT in 1200 g. Pri obremenitvi 1000 g pokaže -542,60 uT in 1015,94 g, namesto 545 uT in 1000 g. Eksperimentalno sem ugotovil, da ima največ pogreška zadnja meritev, 1500 g. Sklepam, da je pri merjenju karakteristike tu prišlo do neznatnega premika prožnega traku, ali pa do nenatančnosti referenčne količine vode. Zato lahko sklepam, da ta težava prinese največ ±40 g odstopanja od dejanske vrednosti.

Če seštejem vsa odstopanja skupaj, pridem do končne številke ±56 gramov odstopanja od dejanske vrednosti.

3. Zaključek

Sledi vrednotenje zastavljenih hipotez. Zdi se mi, da sem, glede na potrebno improvizacijo, izdelal precej natančen merilnik količine vode. Odstopanje za ±56 gramov pri 1500 g obremenitve po mojem mnenju izpolni zahtevo po natančnosti merilnika.

Zahtevo o ponovljivosti merilnih rezultatov nisem preveč dobro izpolnil. Potrebno bi bilo preveriti izmerjeno karakteristiko. Prav tako bi bilo zelo koristno fiksirati kovinski trak, ne da bi ob tem oviral

upogibanje. To bi zelo pripomoglo k natančnosti. Po drugi strani pa se merilnik do zdaj še ni trajno deformiral, zato lahko rečem, da sem delno izpolnil zahtevo o ponovljivosti merilnih rezultatov.

Uporabil sem princip upogibanja kovinskega traku, prav tako sem izvedel kompenzacijo nelinearnosti, čeprav z drugačno metodo, kot je bilo na začetku mišljeno.

Močno sem bil presenečen nad občutljivostjo merilnika, saj je preko upogiba kovinskega traku, debelega 1,5 mm, zaznal, če imam na merilnik obešeno **prazno** plastenko. Tudi programska rešitev problema s šumom se mi je zdela elegantna. V kolikor popravim karakteristiko in fiksiram kovinski trak, ima merilnik tudi dober praktičen potencial, zato se bi splačalo vložiti še nekaj truda v ta izdelek. Kar se tiče uporabljenih postopkov pri izdelavi, pa mislim, da imajo veliko splošno uporabnost. To vključuje vse od aproksimacije nelinearne karakteristike, torej linearne aproksimacije, do programskega odstranjevanja šuma, pa vse do strojniških vprašanj, povezanih s prožnostjo materiala. Vsekakor je bil to zanimiv projekt.

Seznam literature in virov

Online Curve Fitting. Dostopno prek: https://mycurvefit.com/ (28.12.2020)

C pow() - C Standard Library. Dostopno prek: https://www.programiz.com/c-programming/library-function/math.h/pow (28.12.2020)

Tare weight – Wikipedia. Dostopno prek: https://en.wikipedia.org/wiki/Tare_weight (26.12.2020)

Neodim magnet - Wikipedija, prosta enciklopedija. Dostopno prek: https://sl.wikipedia.org/wiki/Neodim_magnet (26.12.2020)

Halbach array – Wikipedia. Dostopno prek: https://en.wikipedia.org/wiki/Halbach_array (26.12.2020)

Adafruit LSM303DLH Magnetometer library. Dostopno prek: GitHub - adafruit/Adafruit_LSM303DLH_Mag (26.12.2020)

LSM303DHLC datasheet. Dostopno na: https://cdn-shop.adafruit.com/datasheets/LSM303DLHC.PDF (26.12.2020)

Priloga: Koda za merjenje karakteristike senzorja

Branch: Characteristics-measurement-code.
https://github.com/kristjansoln/Merilnik_LSM303DLHC/tree/Characteristics-measurement-code

```
#include <Adafruit_LSM303DLH_Mag.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

// Function header
float collectSamplesAndMean(int sampleCount);
void collectSamples(int sampleCount);
void displaySensorDetails(void);
void magInit(void);

// LSM303DLHC object declaration
Adafruit_LSM303DLH_Mag_Unified mag = Adafruit_LSM303DLH_Mag_Unified(12345);

// Global variables declaration
float x, y, z = 0;
char ledPin = 14;
char buttonPin = 12;

void setup(void)
{

#ifndef ESP8266
  while (!Serial)
    ; // will pause Zero, Leonardo, etc until serial console opens
#endif

  Serial.begin(9600);
  magInit();
  displaySensorDetails();

  // IO config
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop(void)
{
  if(digitalRead(buttonPin) == 0)
    {
      float totalMean = 0;
      digitalWrite(ledPin, HIGH);
      Serial.println("Button pressed, measuring");
      for (int i = 0; i < 5; i++)
      {
        totalMean += collectSamplesAndMean(400);
        delay(500);
      }
      totalMean = totalMean / 5.0;
      Serial.println("5 measurements average: " + (String)totalMean);
      digitalWrite(ledPin, LOW);
```

```
      // na roke izračunam odstopanje (približno)
    }
}

// FUNKCIJE

float collectSamplesAndMean(int sampleCount)
{
    uint16_t cycleCounter = 0;
    float mean = 0;
    sensors_event_t event;
    while (1)
    {
      /* Get a new sensor event */
      mag.getEvent(&event);
      z = event.magnetic.z;

      // skip first 15 measurements, collect "sampleCount" valid measurements
      cycleCounter++;

      if (cycleCounter > 15)
      {
        // validate
        if ((z > 0.01) | (z < -0.01))
        {
          mean += z;
        }
        else
        {
          // if invalid, repeat the measurement
          cycleCounter--;
        }
      }

      if (cycleCounter >= (sampleCount + 15))
      {
        // divide sum of values by number of values => calculate mean
        mean = mean / sampleCount;
        Serial.println("Mean: " + String(mean));
        return mean;
      }

      // delay - read frequency: 13 Hz - 75 ms
      //                         200 Hz - 5 ms
      delay(5);
    }
}

void magInit(void)
{
    // Initialise the sensor
    while (!mag.begin())
    {
      Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
      delay(1000);
    }
    // Gain
    //mag.enableAutoRange(false);
    //mag.setMagGain(LSM303_MAGGAIN_1_9);
    mag.enableAutoRange(true);

    // Rate
    mag.setMagRate(LSM303_MAGRATE_220);

    return;

}
```

Priloga: Koda za merjenje in izpisovanje količine vode v posodi

Branch: Quintic-regression-approximation

https://github.com/kristjansoln/Merilnik_LSM303DLHC/tree/Quintic-regression-approximation

```
#include <Adafruit_LSM303DLH_Mag.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>
#include <math.h>

// Function header
double collectSamplesAndMean(int sampleCount);
void collectSamples(int sampleCount);
void displaySensorDetails(void);
void magInit(void);

// LSM303DLHC object declaration
Adafruit_LSM303DLH_Mag_Unified mag = Adafruit_LSM303DLH_Mag_Unified(12345);

// Global variables declaration
double x, y, z, finalWeight, finalMagVal;
char ledPin = 14;
char buttonPin = 12;

void setup(void)
{
```

```
  Serial.begin(9600);
  magInit();
  displaySensorDetails();

  // IO config
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
}









void loop(void)
{
  static double tare = 0; // tara

  finalMagVal = collectSamplesAndMean(400);
  finalWeight = -
154785.2 - 1288.846 * pow(finalMagVal, 1) - 4.17052 * pow(finalMagVal, 2) - 0.006647351 * pow(finalMagVal, 3) - 0.000005248848 * pow(fina
lMagVal, 4) - 1.645575e-9 * pow(finalMagVal, 5);

  Serial.println("Masa: " + ((String)(finalWeight - tare)) + "g   B_z: " + (String)finalMagVal + "uT    Tara: " + (String)tare);

  if (digitalRead(buttonPin) == 0) // tariranje
  {
    tare = finalWeight;
  }

  delay(200);
}

//////////////////////////// FUNKCIJE ////////////////////////

double collectSamplesAndMean(int sampleCount)
{
  uint16_t cycleCounter = 0;
  double mean = 0;
  sensors_event_t event;

  //Serial.println("---- begin collectSamplesAndMean ---------------");
  while (1)
  {
    /* Get a new sensor event */
    mag.getEvent(&event);
    z = event.magnetic.z;

    // skip first 15 measurements, collect "sampleCount" valid measurements
    cycleCounter++;

    if (cycleCounter > 15)
    {
      // validate
      if ((z > 0.01) | (z < -0.01))
      {
        mean += z;
      }
      else
      {
        // if invalid, repeat the measurement
        cycleCounter--;
      }
    }

    if (cycleCounter >= (sampleCount + 15))
    {
      // divide sum of values by number of values => calculate mean
      mean = mean / sampleCount;
      return mean;
    }

    // delay - read frequency: 13 Hz - 75 ms
    //                         200 Hz - 5 ms
    delay(5);
  }
}


void magInit(void)
{

  // Initialise the sensor
  while (!mag.begin())
  {
    Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
    delay(1000);
  }
  // Gain
  //mag.enableAutoRange(false);
  //mag.setMagGain(LSM303_MAGGAIN_1_9);
  mag.enableAutoRange(true);
```

```
  // Rate
  mag.setMagRate(LSM303_MAGRATE_220);

  return;
}
```