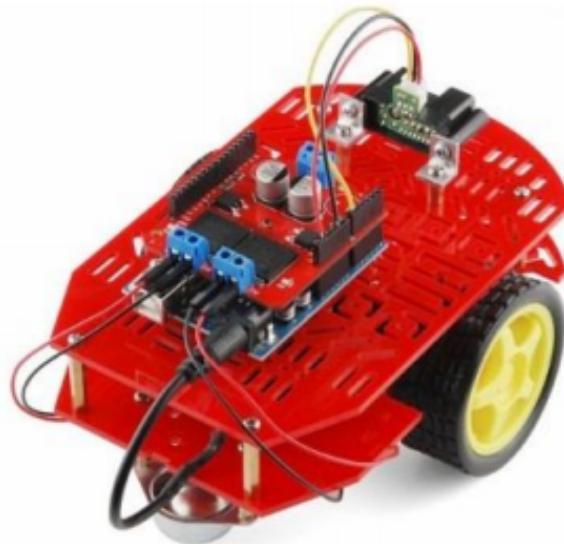


Forårs Semester 2016

Line following robot

Gruppe 4

2. Semester IT-Teknolog



Gruppe medlemmer: Anders Pedersen - Kasper Delfs - Kristian Porsborg

Vejledere: Jesper M. Kristensen og Steffen Vutborg



2. Semester
IT-teknolog
Sofiendalsvej 60
9200 Aalborg SV
<http://www.ucn.dk/>

Titel:

Line following robot

Projekt Periode:

2. Semester | Forårs semester 2016

Projectgruppe:

Gruppe 4

Medvirkende:

Anders Pedersen

Kasper Delfs

Kristian Porsborg

Vejleder:

Jesper M. Kristensen og
Steffen Vutborg

Sideantal: TBD ¹

Appendiks: TBD ²

Færdiggjort: 7/6-2016

Forord

I dette semester projekt skal en robot bygges til at følge en linje. Ud fra problem-formulering er noget hardware stillet til rådighed hvor gruppens formål vil være at implementere den nødvendige hard- og software, så robotten vil være i stand til at manøvre rundt på en oplagt linje.

Produktet udvikles af it-teknologstuderende fra University College Nordjylland på 2. Semester på elektronik linjen. Produktet udvikles for at øge kompetencer og forståelser inden for allerede kendt eletorik- og programmerings viden med forbehold for at anvende det i praksis.

Anders Pedersen

Kasper Delfs

Kristian Porsborg

Indholdsfortegnelse

1	Foranalyse	1
1.1	Indledning	1
1.2	Line Track	1
2	Kravspecifikation	4
3	Hardware	5
3.1	Hardware Overblik	5
3.2	Mikrocontroller	5
3.3	Sensor	6
3.4	Low-Pass filter	7
4	Software	9
4.1	Software introduktion	9
4.2	Arduino kode med én sensor	10
4.3	Test og delkonklussion	11
4.4	Arduino kode med flere sensore	12
4.5	PID regulering	14
4.6	Test og delkonklussion	15
5	Embedded	16
5.1	Introduktion til embedded systems	16
5.2	ADC	16
5.3	PWM	18
6	Test	20
7	Konklusion	21
8	Perspektivering	22
	Bibliography	23
9	Appendix	24

Foranalyse 1

1.1 Indledning

Automatisering er en stor del af samfundet den dag i dag. Fabrikker anvender automatisering for at kunne øge produktion og salg.

En stigende automatisering ses især i masseproduktion, heriblandt eksempelvis produktionen af biler.

Sikkerhedsmæssige aspekter bliver udviklet i takt med automatiseringen hvor der i dag ses eksempler på biler som bremser selv hvis den forankørende kommer for tæt. Dette er med til at skabe fokus på blandt andet sikkerhed hvor computerstyring er stærkt tiltagende. Indenfor rammen af automatisering og sikkerhed kan om andre google nævnes med udvikling af deres fuldt automatiseret biler. Google's selvkørende bil (jf. kilde [2])

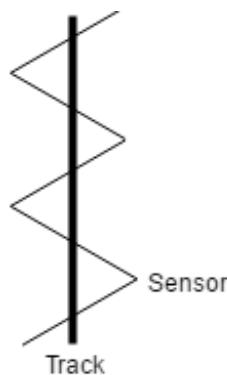
Dette projekt handler om at fremstille et automatiseret produkt i form af en selvkørende bil. Bilen skal injusteres med et line track system, som derved skal få den til at følge en opstillet bane, bestående af en hvid overflade og en sort streg af tape der markerer banens gang.

1.2 Line Track

Den linjefølgende robot følger en bestemt bane ved at anvende princippet om at modtage et konstant signal. Signalet sendes fra lyssensoren med en varierende værdi afhængig af farven på den kørete overflade.

Den forudbestemte bane er defineret ved en sort bane med en minimusbredde på 30 mm omgivet af hvid eller gråtonet omkringliggende farve.

Gruppen har valgt som fremgangsmåde først at anvende én sensor til at sende den værdi som læses fra den kørete overflade.



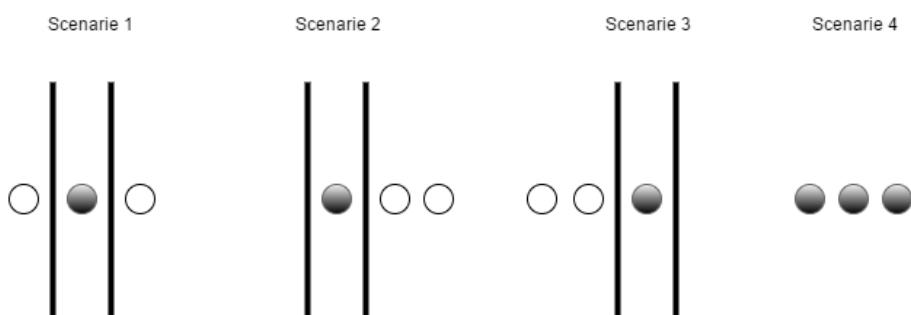
Figur 1.1: Sensor algoritme over én sensor

På ovenstående figur ses det hvordan sensoren registrerer den omkringliggende farve og drejer til højre. Herfra fortsætter bilen indtil den optegnede bane igen registreres og der igen læses en hvidnuanceret farve efter stregen. Da den hvidtonede overflade registreres på ny justerer bilen retning mod venstre for at kunne følge linjen.

Videreudvikling af Line Track

For at gøre bilen i stand til at kunne følge linjen efter bedste og hurtigst mulige evne har gruppen valgt at implementere yderligere sensore. Dette tillader bilen at kunne følge en lige linje samt være i stand til at registrere fremadkomne ændringer i banens forløb.

På nedenstående figur ses hvordan implementering af flere sensorer hjælper bilen i at bestemme banens forløb.



Figur 1.2: Sensoralgoritme over tre implementeret sensorer

Ved anvendelse af tre sensorer skabes fire mulige scenarier som bilen skal være i stand til at forholde sig til.

1. Scenarie

- Her ses det hvordan de tre sensorer tilgår banens forløb. Det antages at bilen følger en lige linje hvor den midterste sensor registrerer den sortnuanceret overflade. De to yderste sensorer registrerer en anden overflade som er ens på begge sider og bilen skal derfor ikke justere retning.

2. Scenarie

- Her registrere venstre sensor banens forløb hvilket indikerer at banens forløb har ændret sig. Den midterste og den højre sensor registrerer begge den hvidtonede overflade og bilen skal derfor justere ind til venstre for at den midterste sensor igen læser den mørke overflade så bilen igen kører en lige linje.

3. Scenarie

- Ligesom 2. scenarie registrerer her én af ydersensorne at banens forløb har ændret sig. Højre sensor registrerer den mørke overflade hvorfra de to andre sensorer registrerer den hvidtonede overflade. Herfra skal bilens retning ligeledes ændres så den midterste sensor læser den mørke overflade igen.

4. Scenarie

- I dette scenaie antages at en uforudset hindring eller hændelse har påvirket sensor eller banens forløb. Dette kan eksempelvis være fejlregistrering fra sensor eller snavs eller misfarvninger på den opstillede bane.
Ingen af sensorerne er i stand til at videresende en registreret måling af den pågældende overflade og bilen kan derved ikke registrere hvor den opstillede bane er. I dette scenarie står bilen derfor stille.

Kravspecifikation 2

I det følgende afsnit gives der indblik i de krav som er sat i problemanalysen samt de krav projektgruppen har sat for at imødegå produktet præsenteret i projektbeskrivelsen.

Generelle krav

1. Projektet skal konstrueres omkring Sparkfun's Magician Chassis hvor to dc motorer er inkluderet.
2. Til projektet skal en mikrocontroller anvendes baseret på mikroprocessoren PIC32MX 32bit.
3. Produktet skal fremvises og demonstreres til projektevalueringen.
4. Projektet skal dokumenteres i form af en rapport.

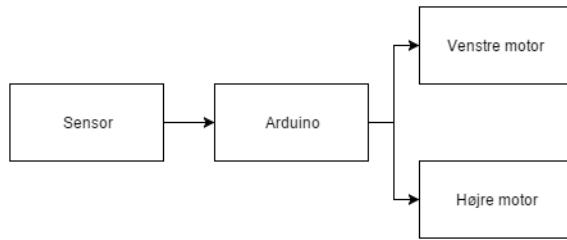
Krav til produktet

1. Produktet skal være i stand til køre på en linje med en minimumsbredde på 30 mm.
 - a) Styring skal foregå ved hjælp af feedback fra en eller flere lyssensorer.
 - b) Farven på linjen skal være sort eller gråtonet over 75%. Den omkring liggende farve skal være hvid eller gråtonet under 50%.
2. Softwaren til produktet skal skrives i MPLAP.

Hardware 3

3.1 Hardware Overblik

I dette afsnit bliver beskrevet hvilket hardware komponenter der blev valgt i forhold til produktet og hvorfor. Endvidere bliver funktionaliteten af de enkelte komponenter afviklet og der gives et overblik over det samlede produkt.



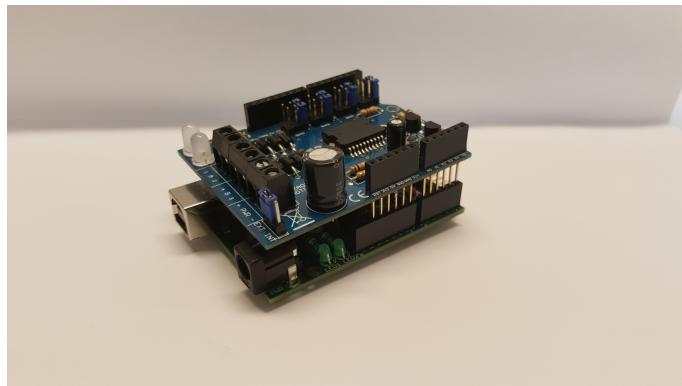
Det ovenstående flowchart viser hvordan produktet fungerer. Sensoren opfanger ved hjælp af linetrack et følsomhedssignal som enten er sort eller hvidt. Dette signal videresendes til arduinoen som konfigurerer signalet og justerer motorene efterfølgende.

3.2 Mikrocontroller

Til styring af robotten har gruppen valgt at anvende et UCN board. Dette besidder 128 kb flash memory og 16 pins til in- og output. Fem af disse er analoge pins som fungere i et spændingsinterval mellem 2.3-3.6 V.

Derudover har UCN boardet to UART pins (hardware serial ports). Disse har gruppen valgt at anvende under test af ADC3.2 for at verificere at ADC'en virker efter hensigten. Til mikrocontrolleren anvendes et motorshield hvilket muliggør anvendelsen af flere pins samt styring af de påmonteret DC motorer på Sparkfun's Magician Chassis.

UCN boardet's mikroprocessor er en pic32mx250f128b som er en del af pic32 familien og har en maks clock frekvens på 50MHz. Strømmen tilføres mikrokontrolleren via et 7.2 V. ekstern batteri gennem et Power Jack som er monteret på UCN boardet's PCB(Printed circuit board).



Figur 3.1: UCN board med påmonteret motorshield til styring af robotten.

3.3 Sensor

Sensoren der anvedes er en "Line Sensor Breakout - QRE1113" fra sparkfun.com[5]. Det er en analog sensor som sidder på et breakout board i en spændingsdeling. Dette betyder at der blot skal aflæses spænding på en pin for at få en værdi der svarer til en lysstyrke fra sensoren.



Specifikationerne gældende for sensoren:

- 5VDC operating voltage
- 25mA supply current
- Optimal sensing distance: 0.125" (3mm)
- Dimension: 0.30 x 0.55 " (7.62 x 13.97 mm)

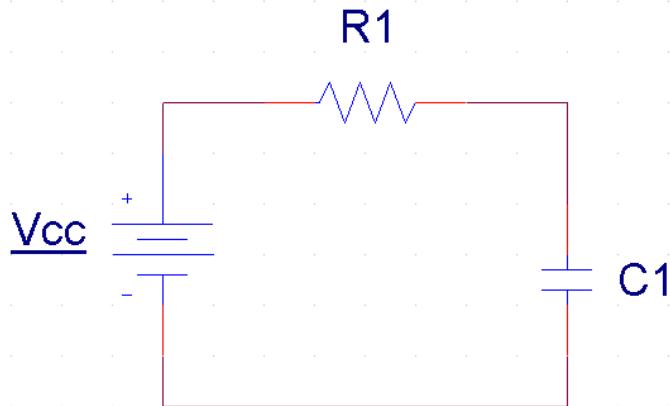
(jf. kilde [4])

For at kunne anvende lyssensoren med en arduino skal noget software skrives. Sensoren sidder i en spændingsdeling og outputtet fra lyssensoren tilkobles en pin på arduinoen. Herfra foretages en analog måling med ADC'en (analog til digital konverter) på arduinoen. Dette gøres ved at anvende analogRead() i softwaren.

I sensorens spændingsdeling sidder en transistor som kan generere høj frekvens støj. For at filtrere støj'en væk anvendes et low-pass filter.

3.4 Low-Pass filter

ADC'en på Arduino UNO's mikroprocessor har en samplingfrekvens på 10k hvor UCN boardet sampler ved 1100. Her kan forekomme alaising og for at undgå dette implementeres et low-pass filter. Et low-pass filter er et filter som sorterer høje frekvenser fra men samtidig tillader DC (jævnstrøm) igennem.



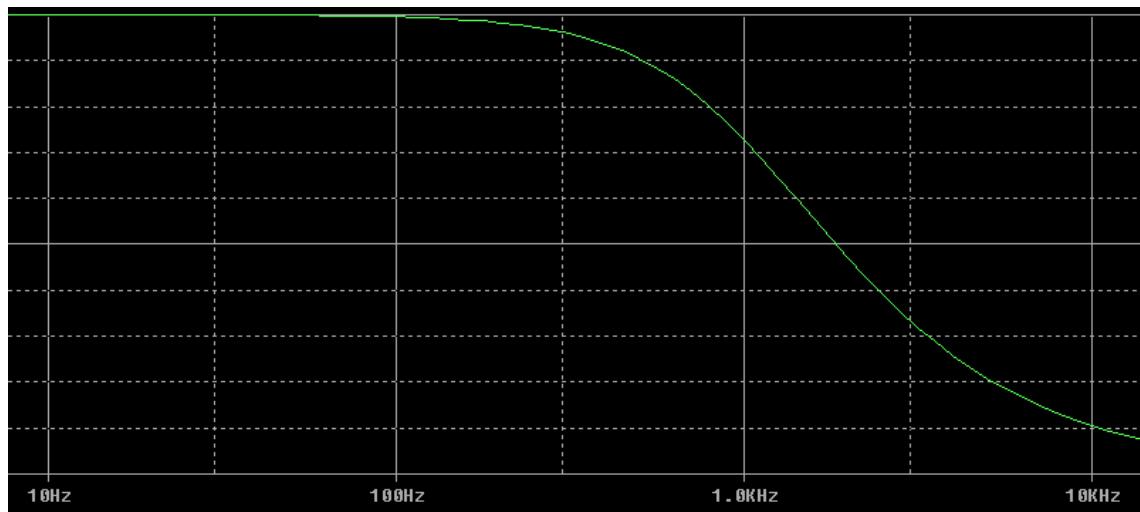
Et low-pass filter kan konstrueres med en modstand og en kondensator i serie. Kondensatoren er forbundet til stel så den skaber en AC-kortslutning (vekselsstrøm) og derved filtrer AC væk fra signalet. Lave frekvenser filtreres ikke væk, fordi kondensatoren bruger tid til at lade op og derved ikke længere fungerer som stel fra de frekvenser.

Cut off frekvensen som er frekvensens afskæringen, kan beregnes ved hjælp af følgende formel.

$$f_c = \frac{1}{2\pi RC} \quad (3.1)$$

$$f_c = \frac{1}{2\pi 1000 * 150 * 10^{-9}} = 1061,03 Hz \quad (3.2)$$

På figur 3.2 ses det hvornår frekvenserne skæres fra ved brug af en 1k modstand i en serie forbindelse med en 150 nanofarad kondensator.



Figur 3.2: Eksempel på low-pass filter med 1k modstand og 1nF kondensator.

Software 4

4.1 Software introduktion

Den indlende idé til en softwareløsning som kan følge en linje er udarbejdet, og ses på figur 4.1.

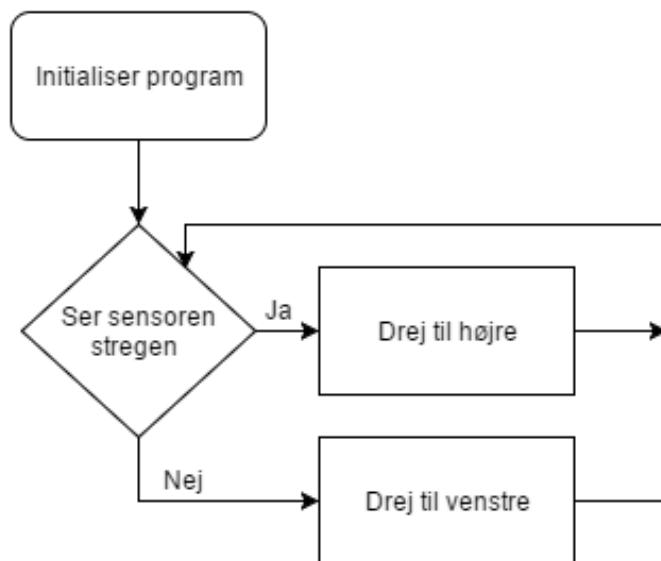
I dette afsnit ses der nærmere på den software som er skrevet i takt med udviklingen af projektet. Her introduceres om andre den indledende løsningsidé samt udviklingen deraf. Herfra ses der nærmere på videreudviklingen ad denne, fra implementering af én til flere sensorer samt betydningen deraf for produktet.

Endvidere præsenteres test i henhold til virkningen af de forskellige stykker software som er konstrueret hertil.

Indledende problemløsning

For at gøre robotten i stand til at kunne følge den opstillede bane er en indledende problemløsning opsat. Her har gruppen valgt sætte fokus på implementering af én sensor som kan videreudvikles fremadrettet.

Softwareen skrevet har to funktioner baseret ud fra én registreret måling. På nedenstående figur ses hvorledes softwarens handling forløber.



Figur 4.1: Den inledende software løsning til linetracking med 1 sensor.

Som det ses på ovenstående figur har softwaren to funktioner. Ses stregen på den opstillede bane, hvis ja drej til højre, hvis nej drej til venstre. Dette resulterer i at robotten er i stand til at kunne følge kanten af stregen ved at registrere de forskellige overfladefarvemålinger.

4.2 Arduino kode med én sensor

Den indledende løsning blev først implementeret på en Arduino Uno mikrocontroller hvorefter konceptet blev testet og verificeret. Koden er derfor i sin indledende fase også skrevet i Arduino.

```
void FollowLine()
{
    while (analogRead(sensor_one_pin) > mean_value) //sensor -> white paper
    {
        analogWrite(left_motor_pin, SLOW);      //turn right
        analogWrite(right_motor_pin, FAST);     //

        Serial.print("Turn right. Sensor data =");
        Serial.println(analogRead(sensor_one_pin));

    }
    while (analogRead(sensor_one_pin) < mean_value) //sensor -> black line
    {
        analogWrite(left_motor_pin, FAST);      //turn left
        analogWrite(right_motor_pin, SLOW);     //

        Serial.print("Turn left. Sensor data =");
        Serial.println(analogRead(sensor_one_pin));

    }
}
```

Figur 4.2: FollowLine() funktion.

På figur 4.2 ses funktionen follow_line(). Koden fungerer sålades at når sensoren registrerer den hvide omkringliggende farve drejes der til højre og når den sorte linje registreres drejes der til venstre. Robotten kører derved kun på kanten af linjen. Ved brug af én sensor har gruppen valgt at anvende et referencepunkt til sammenligning med målinger. Dette referencepunkt kaldet mean_value i koden, er gennemsnittet for den sorte streg og den hvide baggrund. Herefter hvis målingen er under gennemsnittet, drejes til den ene siden, og er den over drejes til den anden side.¹

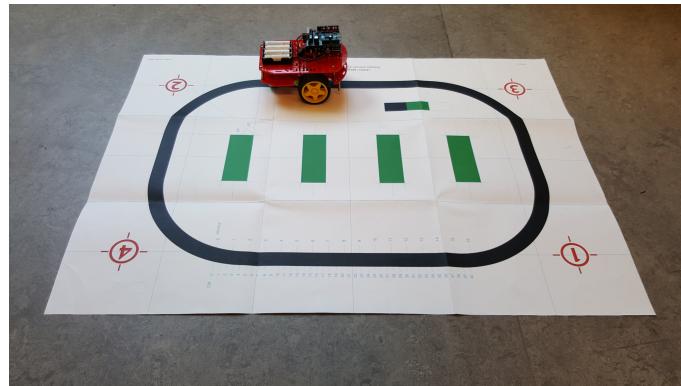
¹FIXME Note: find ud af om det ikke er for tæt på det der står ved flowchartet

4.3 Test og delkonklusion

Testen er udført ved hjælp af en legomindstorm bane i oval form. Ud fra det introducerede linetrack system blev konceptet testet ved brug af én sensor på den opstillede bane.

Robotten fuglte den sorte streg

Ud fra denne test kan det konkluderes at softwaren virker i henhold til den indledende problemløsning. Robotten er i stand til at følge kanten af linjen ved hjælp af de forskellige overfladefarveværdier som sensoren registrerede gennem testen.

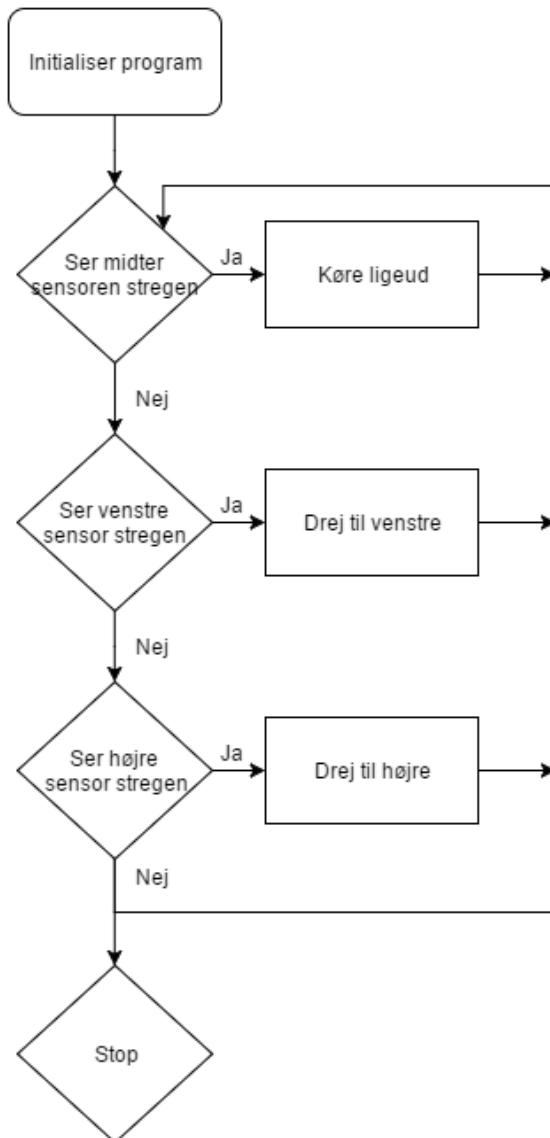


Figur 4.3: Den inledende software løsning til linetracking med 1 sensor i et superloop.

4.4 Arduino kode med flere sensore

For at robotten er i stand til at kunne følge linjen hurtigst og bedst muligt implementeres yderligere sensore. Dette gøres for at robotten kan navigere nøjagtigt og præcist. Sensorene er placeret ud fra en skabelon som gruppen har 3D printet for at skabe de bedste omstændigheder muligt for at få så præcise måliner som muligt. Her anvendes konceptet vist i figur 1.2 i Line Track.

Herfra ændres softwaren fra den indledende problemløsning til at håndtere tre sensore i stedet for én.



Figur 4.4: Software flowchart over problemløsningen ved hjælp af tre sensore i et superloop.

Som det ses på figur 4.4, er der fire mulige scenarier som blev præsenteret i figur 1.2. Som vist i flowchartet registrerer sensorene om stregen er visuel for sensoren eller ej. Såfremt sensoren registrerer en værdi som fortæller om den er på stregen, kører robotten ligeud indtil en anden værdi måles.

Hvis en anden sensor end den midterste afgiver en måling af den opstillede bane angiver dette at banens forløb har ændret sig, og robotten anvender derved de to funktioner anvist i flowchartet hvorvidt der skal justeres til venstre eller højre.

Såfremt ingen fejlmålinger fra sensoren eller snavs stopper robotten fortsætter denne i et superloop.

Ved videre implementering af yderligere sensore kan målinger fra den enkelte sensor sammenlignes med gennemsnittet af de andre.

Videreudvikling af Arduino kode

```
void FollowLine2()
{
    if (analogRead(sensor_middle_pin) > (analogRead(sensor_right_pin) + analogRead(sensor_left_pin)) / 2) {
        drive_straight(); // middle sensor on the line --> *|*|
    }

    if (analogRead(sensor_right_pin) > (analogRead(sensor_middle_pin) + analogRead(sensor_left_pin)) / 2) {
        turn_right(); // right sensor on the line --> **|
    }

    if (analogRead(sensor_left_pin) > (analogRead(sensor_middle_pin) + analogRead(sensor_right_pin)) / 2) {
        turn_left(); // left sensor on the line --> |*||
    }
}
```

Figur 4.5: Uddrag af den anvendte Arduino kode til problemløsningen med tre sensorer.

²Fixme Note: skriv noget efter figuren?

4.5 PID regulering

For at opnå en mindskning i fejl målingerne fra sensoren, kan implementeres en PID. PID står for Proportional Integral Derivative controller. Formålet med en PID er at man konstant mäter en fejl værdi som består af differensen mellem en ønsket værdi og en målt værdi. PID'en forsøger at minimere fejlmålingerne ved at justerer fejl målingen over tid.

PID kan opskrives på følgende måde.

$$u(t) = K_p e(t) + K_i \int_0^t e(T) dT + K_d \frac{de(t)}{dt} \quad (4.1)$$

hvor,

K_p , er proportional ledet som beskriver den nuværende værdi, hvilket er den ønskede værdi trukket fra den målte værdi.

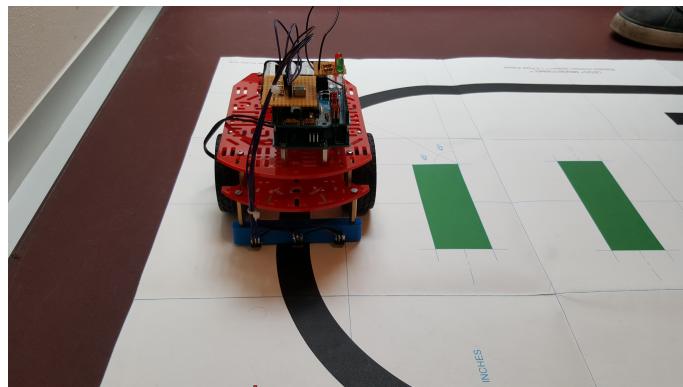
K_i , er integralledet som beskriver den forhændsværende værdi, hvilket betyder at fejlværdien bliver justeret ind over tid, dette gøres ved hjælp af det ønskede signal i forhold til det givne signal, som drager et areal af hver periode.

K_d , er differentiale ledet som beskriver en mulig fremtidig fejl, som er vudret ud fra forhændsværende værdier. Dette gøres ved at mæle hældningen på det givne signal for at forsøge at justere det ind efter den ønskede værdi.

³FiXme Note: beslut hvad der skal gøres med afsnittet

4.6 Test og delkonklusion

Testen er udført på samme bane og med samme fremgangsmåde som testen med anvendelse af én sensor. Robotten foretog adskellige omgange uden at registrere målinger som afsporede robotten. Det kan konkluderes at afstanden imellem sensorene var for stor, hvilket medførte at robotten reagerede sent på målinger fra sensorene i siderne. Dette resulterede i at robotten ikke er i stand til at følge banen hurtigst og bedst muligt. PID ville forbedre dette problem men ikke løse det. Gruppen har valgt at frapriotere denne mulighed på grund af manglende tid og derfor har valgt at lægge fokus på implementering af software til embedded.



Figur 4.6: Test af Arduino kode med tre anvedte sensore.

5.1 Introduktion til embedded systems

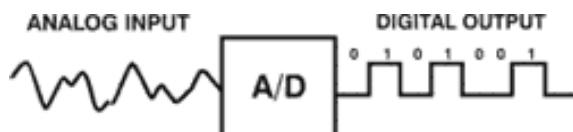
Med henblik på videreudvikling af produktet har gruppen valgt at lægge fokus på at overføre robottens hard- og software fra Arduino til et embedded system. Dette er gjort med forbehold for læringsmålene sat i projektbeskrivelsen, hvor ét af fokusobjekterne er på øvelse og repetition af implementering og udvikling af software i embedded systemer.

Medvidere her ses også mulighed for prisreducering i henhold til hardware gældende for produktet. Under den indledende process blev et Arduino Uno board anvendt for at teste den indledende problemløsning gruppen havde opstillet. Herfra har gruppen lagt fokus på at overføre softwaren skrevet i Arduino til MPLAP's platform. Dette har betydet at mikrocontrolleren som først blev anvendt blev udskiftet med et UCN board ?? som gruppen konstruerede under 1. semester.

De følgende afsnit introducere hvorledes gruppen anvender og implementere ADC(analog til digital konverter) og PWM (pulsbreddemodulation) for at robotten skal virke bedst muligt i henhold til projektbeskrivelsen.

5.2 ADC

ADC eller A/D converter er en enhed der benyttes til at omdanne et elektrisk analog signal til et digitalt signal som kan viderebearbejdes i software. Den ADC som sidder på PIC32mx250f128b [3] er en 10 bit ADC, som kan have op til 9 analoge inputs. På UCN boardet er 6 mulige inputs hvoraf gruppen anvender 3 af disse til lyssensorerne.



Figur 5.1: ADC diagram.[1]

Den anvendte ADC er implementeret ved hjælp af ADC reference manualen fra microchip, som er et generelt datablad for PIC32 processorene. Her findes de nødvendige opsætninger som skal foretages for at anvende ADC'en efter den ønskede hensigt.

ADC'en er implementeret ved hjælp af fremgangsmåden anvist i databladet. Her sættes en række registre op efter den ønskede opsætning.

¹

¹ FIXme Note: kilde til datablad

OPERATION SEQUENCE	
Sample MUX A Inputs: AN0	Convert AN0, Write Buffer 0x0
Sample MUX A Inputs: AN1	Convert AN1, Write Buffer 0x1
Sample MUX A Inputs: AN2	Convert AN2, Write Buffer 0x2
	Interrupt; Change Buffer
Sample MUX A Inputs: AN0	Convert AN0, Write Buffer 0x8
Sample MUX A Inputs: AN1	Convert AN1, Write Buffer 0x9
Sample MUX A Inputs: AN2	Convert AN2, Write Buffer 0xA
	Interrupt; Change Buffer
	Repeat

Figur 5.2: Operations sekvens.

Den angivne sekvens viser på figur 5.2, hvilken rækkefølge den udfører sine funktioner i. Softwaren gældende for ADC'en er bygget op efter samme princip hvor der først foretages en sampling gennem en positiv målt værdi fra AN0, som er den pin som er tilknyttet vores MUX A(multiplexer). Herfra konverteres den registrerede værdi og tilskrives den digitalt til bufferen. Denne sekvens gør sig gældende for alle 3 sensorer, hvorefter bufferen ændres og et interrupt foretages.

Buffer Address	Buffer @ 1st Interrupt	Buffer @ 2nd Interrupt
ADC1BUF0	AN0 sample 1	AN0 sample 5
ADC1BUF1	AN2 sample 2	AN2 sample 6
ADC1BUF2	AN1 sample 3	AN1 sample 7

Figur 5.3: Tilskrivning til ADC buffer.²

Denne sekvens foretages kontinuerligt af ADC'en, hvor de konverterede målinger skrives til bufferen som vist på figur 5.3.

En ADC kan generere noget kaldet "Aliasing" som er kopier af signalet, som bliver genereret ved samplings frekvensen. Dette filtreres væk vha. et low-pass filter3.4.

ADC deltest

For at verificere funktionaliteten på ADC'en er en test blevet udført. Testen er udført vha. et FTDI kabel (se figur 5.4) som opretter en serial forbindelse på computeren hvor målte ADC værdier kan skrives til. Derved kan det visuelt bekræftes at ADC'en mäter en værdi som ligger i indenfor den det forventede område for en 10 bit ADC, som er 1024, eller fra 0 til 1023.



Figur 5.4: ftdi kabel til oprettelse af serial forbindelse.

Testen blev udført ved at skrive ADC bufferne ud til den serielle port hvor værdierne blev aflæst. Der blev ikke taget højde for hvilke analoge pins der var forbundet til hvad, blot at ADC bufferne blev skrevet til den serielle port på PC'en.

```

while(1)
{
    if(IFSO & 0x10000000)
    {
        IFSOCLR = 0x10000000;
        x = ADC1BUF0;
        UART1Write("B0 = ");
        UART1Write(itoa(data_A0,x,10));

        y = ADC1BUF1;
        UART1Write("      B1 = ");
        UART1Write(itoa(data_A1,y,10));

        z = ADC1BUF2;
        UART1Write("      B2 = ");
        UART1WriteLn(itoa(data_A4,z,10));
    }
}

```

Figur 5.5: ADC deltest.

Testen blev foretaget med et potentiometer tilkoblet de 3 ADC indgange (AN3, AN4, AN5) og der blev justeret fra min til maksimum. Det blev observeret at den udskrevne værdi gik fra 0 til 1023, hvilket også er opløsningen på ADC'en. Derved kan det konkluderes at testen var succesfuld for ADC'en.

5.3 PWM

PWM eller pulsbreddede modulation er en måde et firkant signal, hvor tiden signalet er højt kan justeres. Procentdelen hvor signalet er højt, bliver udregnet i forhold til én periode, hvilket er længden imellem 2 frekvenser. Duty cyclen på et PWM signal er mellem 0 og 100%. F.eks. svarer 0% til at der intet signal er og 50% svare til at

signalet er høj halvdelen af tiden. Dvs. et PWM signal med en duty cycle på 50% og en amplityde på 10 V, vil således være et PWM signal på 5 V. Frekvensen på PWM signalet skal blot være tilstrækkelig høj, så belastningen ikke blive påvirket af det svingende signal.

PWM kan genereres af modulet Output compare på microprocessoren. Da ADC blev konfigureret ved at sætte de enkelte registre, valgte gruppen at prøve at bruge plib eller peripheral library til at opsætte PWM som et alternativ til at opsætte de individuelle registre. Plib er et bibliotek som giver simpel tilgang til hardware funktioner uden at skulle skrive til specifikke registre³.

```
void PWM_plib_setup() {
    CloseOC1(); //close output compare modules while doing setup
    CloseOC2();
    OpenTimer2(T2_ON, 0x550); // timer used for PWM frequency
    RPB7R = 0b0101;           // OC1 B7 -> pin 10 on UCN BOARD
    RPB11R = 0b101;           // OC2 B11 -> pin 9 on UCN BOARD
    /* Enable OC | 32 bit Mode | select Timer2 | Continuous O/P | OC Pin High, S Compare value, Compare value*/
    OpenOC1(OC_ON | OC_TIMER_MODE32 | OC_TIMER2_SRC | OC_CONTINUE_PULSE | OC_LOW_HIGH, 0x550, 0x500);
    OpenOC2(OC_ON | OC_TIMER_MODE32 | OC_TIMER2_SRC | OC_CONTINUE_PULSE | OC_LOW_HIGH, 0x550, 0x500);
}
```

Figur 5.6: PWM setup kode.

PWM opsætning er skrevet med udgangspunkt i et eksempel fundet i "pic32-lib-help" fra microchip.⁴

På figur 5.6 ses opsætning af PWM ved brug af plib registreret. Det første der foretages er, at output compare modulet bliver slukket, således det senere kan åbnes med de valgte opsætninger.

Her har gruppen valgt at bruge pin 9 og 10 på UCN bordet til PWM. Disse anvendes ved at skrive til output pin selection registreret (RPnR register) da disse ikke bliver sat i plib.⁵

Af tidsmæssige årsager valgte gruppen at benytte plib, for også at få lidt erfaring med en anden implementerings metode. Dette er gjort frem for at sætte enkelte registre. Koden kan skrives nemmere, da opsætningsmulighederne er samlet et sted, og funktioner er allerede konstrueret. Dog mangler opsætningen af hvilken pin output der bruges, derfor kan registre ikke helt ungåes.

³FiXme Note: kilde lige under

⁴FiXme Note: evt. kilde til plib

⁵FiXme Note: kilde til side 132 pic32 family

Test 6

Under videreudviklingen og overførslen til MPLAB's platform mødte gruppen hindring i implementeringen deraf. En samlet test var ikke mulig, da der var problemer med opsætning af PWM modulet. ADC modulet fungerede efter hensigten, med korrekte målinger på 3 pins. Opsætningen af PWM modulet blev ikke færdig udviklet pga. manglende tid. Dette betød at en samlet test ikke var mulig at udføre.

Som afslutning på projektet blev et race track event afholdt, hvor robottens evne til at følge en linje testet. Til denne test valgte gruppen at benytte den funktionelle arudino kode, som allerede var testet. Dog var koden ikke blevet testet i 90 grader sving, så derfor blev koden tweaket så dette var muligt.

Konklusion

7

I dette projekt havde gruppen til formål at designe og implementere den nødvendige hard- og software med henblik på at konstruere en automatisk linjefølgende robot.

Produktet er udviklet ud fra Magician chassis som blev leveret af Sparkfun med en dertil forudbestemt lyssensor. Produktet blev udviklet over flere stadier hvoraf den indledende fase var at implementere en sensor til en Arduino og få robotten til at følge en linje ud fra den feedback som kom fra lyssensoren.

Herfra har gruppen implementeret yderligere 2 lyssensorer til robotten i form af videreudvikling af produktet. Projektgruppen har foretaget test ved brug af én og 3 sensorer tilknyttet en Arduino Uno for at reducere mulige fejl.

Efter succesfulde test af produktet ved brug af Arduino, har gruppen valgt at overføre projektet til MPLAB's platform. Her er der lagt særlig fokus på implementering af ADC og PWM og gjort overvejelser omkring videreudviklingen af produktet. Her er test foretaget for at verificere at ADC og PWM virker efter hensigten, desværre kan det konstateres at gruppen ikke har formået at få det til at fungere sammen. Det kan konkluderes at på grund af manglende tid har gruppen fraprioteret implementering af PID med forbehold for at et fuldt funktionelt produkt kan fremvises.¹

Herfra har gruppen foretaget test af robotten ved brug af 3 sensorer hvor det kan konkluderes at robotten virker efter den ønskede hensigt. I følge kravspecifikationen opstillet i projektbeskrivelsen, kan det konkluderes at afstanden i mellem de 3 sensorer er for stor og robotten kan derfor ikke følge banens korteste forløb. På grund af tidsmangel er en ny 3D skabelon fraprioteret, hvilket ville løse dette problem.

¹FiXme Note: ret

Perspektivering 8

Ved udviklingen af den linjefølgende robot har projektgruppen valgt at fokusere på at udvikle produktet skridt for skridt. Ved at anvende denne fremgangsmåde har det betydet at meget tid er gået med udviklingen og dokumentation af de forskellige stadier. Arduino kunne have været benyttet til blot at verificere at hardwaren fungere, og flytte fokus til implementering af software i MPLAB.

Det kunne være interessant at anvende yderligere sensorer, samt at sænke afstanden mellem de monteret sensorer for få højere præcision, når roboten krydser linjen.

Bibliography

- [1] “ADC figur”. In: (2016). URL: http://www.infocellar.com/networks/fiber-optics/glossaries/a_files/A-D_converter.gif.
- [2] Google. “Link til Google”. In: (). Sidst besøgt 2016-05-06. URL: <https://www.google.com/selfdrivingcar/>.
- [3] Microchip. “PIC32MX250F128B Microcontroller”. In: () .
- [4] sparkfun. “link til sensor”. In: () .
- [5] sparkfun. “link til sparkfun”. In: () .

Appendix 9

Rettelser

Note: indsæt sideantal	i
Note: indsæt sideantal for appendiks	i
Note: find ud af om det ikke er for tæt på det der står ved flowchartet	10
Note: skriv noget efter figuren?	13
Note: beslut hvad der skal gøres med afsnittet	14
Note: kilde til datablad	16
Note: kilde til datablad	17
Note: kilde lige under	19
Note: evt. kilde til plib	19
Note: kilde til side 132 pic32 family	19
Note: ret	21