FLOT FORSIDE

WOW

LAV ET BILLEDE

Indholdsfortegnelse

[Indledning 1](#_Toc430259343)

[Prioritering af rute 1](#_Toc430259344)

[Baggrund og analyse af banen 2](#_Toc430259345)

[Starten 2](#_Toc430259346)

[Første flaskeopsamling 2](#_Toc430259347)

[Vippen 2](#_Toc430259348)

[Fire linjer 2](#_Toc430259349)

[Målskiven 2](#_Toc430259350)

[Flaske forhindring og bump 2](#_Toc430259351)

[Mål 2](#_Toc430259352)

[Robotdesign 3](#_Toc430259353)

[Robot version 1 3](#_Toc430259354)

[Robot version 2 3](#_Toc430259355)

[Robot version 3 3](#_Toc430259356)

[Delbeskrivelse 4](#_Toc430259357)

[EV3 Brick 4](#_Toc430259358)

[Gyro 4](#_Toc430259359)

[Test 1 5](#_Toc430259360)

[Motorer 5](#_Toc430259361)

[Linjefølger 6](#_Toc430259362)

[Linjen 6](#_Toc430259363)

[EV3 farvesensor 6](#_Toc430259364)

[Farvemåling 6](#_Toc430259365)

[Reflekteret lys fra egen lyskilde 6](#_Toc430259366)

[Reflekteret lys fra omgivelserne 7](#_Toc430259367)

[Valg af måle metode 7](#_Toc430259368)

[Test af Målingsttilstande 8](#_Toc430259369)

[Planlægning af kørsel 10](#_Toc430259370)

[Fejlhåndtering 10](#_Toc430259371)

[Program 11](#_Toc430259372)

[Main task 11](#_Toc430259373)

[Musik task 12](#_Toc430259374)

[Farvekalibrering – manualCallibColor() 12](#_Toc430259375)

[CheckIfLost(float lostTimer, bool direction) 12](#_Toc430259376)

[Mission1 (M1) 12](#_Toc430259377)

[Mission6 (M6) 13](#_Toc430259378)

# Indledning

RoboCup-projektet går ud på at konstruere og programmere en robot, til at komme igennem en given bane. Banen har et pointsystem, som skalerer efter hvor hård hver enkelt forhindring er. Point bliver givet, for alt, fra at gennemføre en forhindring, spille musik, have et funktionelt og flot design, til at komme hurtigst i mål. Til dette projekt er der udleveret et LEGO Mindstorm sæt og et programmeringsprogram, kaldet RobotC. Inkluderet i LEGO Mindstorm sættet er CPU’en til robotten, samt motorer, sensorer og div LEGO byggeklodser. Målet for gruppen er at gennemføre banen og alt efter, hvilke forhindringer der viser sig mest lukrative mht. risiko/point, vil det blive forsøgt at klare disse. Dette gøres ved at analysere banen og det udleverede materiale grundigt, for herefter at lægge den bedst mulige taktik for at vinde RoboCup 2015!

# Prioritering af rute

Ruten er blevet gennemgået og en plan af prioriteringer er blevet fremstillet.

Batteri-porten: Hver gennemført port giver 5 point og hvis man ser bort fra ’Vippen’ og dens 2 porte, har banen i alt 12 porte, hvilket er 60 point. Da vi kommer igennem alle 12 porte, ved at følge banens grå linje og da første prioritering er at gennemføre banen, ses det som et mål at samle 60 point her.

Første forhindring – Brudt streg: Den brudte streg giver 10 point, samt 5 point for en batteri-port der er placeret midt på den brudte streg. Dette ses som en relativ nem forhindring og burde være 15 lette point at samle.

Anden forhindring – Flyt flaske: Her vælges først at skubbe flasken over stregen og herefter at lade robotten snore om sin egen akse og smide flasken fra sig, så hårdt som muligt, da der gives 20 bonuspoint for at have den mest destruktive robot. Det ses som en fordel ikke at løfte flasken, hvis disse 20 bonuspoint skal opnås.

Tredje forhindring – Vippen: Vippen ignoreres, da det vurderes, at det ikke er risikoen værd at satse på at gå efter de 30 point, når den første prioritering er at gennemføre hele banen.

Fjerde forhindring – Parallelle streger: Denne forhindring er nødvendig at klare for at kunne gennemføre banen og 10 point samles her.

Femte forhindring – Flaske i målskive: Denne forhindring er blevet nedprioriteret, da det overordnede mål er at komme igennem banen, men vil blive kigget på såfremt det overordnede mål nås.

Sjette forhindring – Rundt om flaske: Denne forhindring ligger ligesom den brudte streg og de parallelle streger på ruten og 10 point samles.

Syvende forhindring – Bump/hjørne: Grundet robottens design vil det ikke være muligt at klare bumpet, så hjørnet ses som det klare mål her og 25 point samles.

Ottende forhindring – Rundt om flaske (modsat første flaske): Denne forhindring ligger ligesom den brudte streg og de parallelle streger på ruten og 10 point samles.

Niende forhindring – Landingsbanen: Prikken over i’et ville være at samle 50 point her, hvilket vil blive prioriteret højt, da banen her er gennemført og der dermed ikke eksisterer nogen risiko.

Ud over de ovenstående 9 forhindringer er der bonuspoint at skrabe sammen. Der gives op til 50 point for at spille musik på turen, hvilket der vil blive lagt en del kræfter i at opnå. Derudover kan 50 point samles for hurtigste robot igennem banen, hvilket ikke er et mål i sig selv. Dog vil der løbene blive optimeret på koden til robottens line-follower, hvilket evt. kan resultere i en hurtig gennemgangstid.

# Baggrund og analyse af banen

For at danne et overblik over banen, deles den ind I forskellige sektioner. Det er derfor muligt at fokusere på en sektion af gangen, både med henhold til design af robotten, pointfordeling og selve softwaren. Her bliver der beskrevet hvilke færdigheder robotten skal have for at gennemføre de forskellige sektioner til fulde. Der vil så kunne tages stilling til hvilken rute der bedst vil kunne betale sig på baggrund af tid, point og sværhedsgrad.

## Starten

Her skal robotten kunne følge linjen, skifte over til en anden linje og tilbage igen.

## Første flaskeopsamling

Her er der placeret en flaske ned af en ”side linje.” Derfor skal robotten skal kunne navigere ned af den linje. Robotten skal herefter kunne samle flasken op og sætte den ned bag et markeret punkt. Det er også muligt at skubbe flasken, dog for mindre points. Efter flaskeopgaven ska robotten kører tilbage til den første linje og fortsætte til næste del.

## Vippen

Her er der to ruter robotten kan tage. Den ene rute går udenom den placerede vippe og her skal robotten bare kunne følge en linje, dog med et lille sving. Den anden rute går hen over en vippe. Vippen har en lille kant og robotten skal derfor være i stand til at kunne køre hen over den og få sig placeret midt på vippen. Vippen ville være den optimale rute point mæssig, men det er risikabelt da robotten kan vælte ned fra siden.

## Fire linjer

Der er fire parallelle linjer, hvor det kun er den tredje af dem som fortsætter. Robotten skal derfor være i stand til at finde den rigtige linje og følge den.

## Målskiven

Her skal der placeres en flaske så tæt på midten af en målskive som muligt. Robotten skal derfor være i stand til at kunne navigere sig frem til flasken og samle den op eller skubbe den. Herefter skal robotten kunne placere flasken så tæt på midten af målskiven som muligt og så finde tilbage til linjen igen.

## Flaske forhindring og bump

Her er der placeret to flasker og et bump. Robotten skal kunne identificere flasken og køre rundt om den. Herefter skal den kunne køre hen over et bump, eller udenom ved hjælp af to plader placeret i en 90 graders vinkel, her er der ikke nogen linje at følge. Efter det skal robotten identificere den anden flaske og køre den modsatte vej rundt om, som den gjorde ved den første.

## Mål

På mållinjen gælder det om at placere sig så tæt på midten som muligt. Robotten skal derfor være i stand til at finde midtpunktet og stoppe på det.

# Robotdesign

Der har været mange overvejelser omkring robottens design undervejs. Det har ledt til flere forskellige versioner. Her beskriver vi de forskellige versioner af robotten og forskellene mellem dem.

## Robot version 1

Version 1 af robotten er opbygget af en simpel ramme, der er sidder rundt om EV3 klodsen. På siden sidder der to motorer, som hvert har et hjul monteret. Robotten er 22cm bred, 19cm lang og 8cm høj.

Da der er to motorer er det muligt at dreje robotten ved at køre hjulene i hver deres retning. Bag på robotten er der monteret et kuglelege, som fungere som det tredje hjul. Selvom robotten har 3 hjul er den stadig meget stabil, og den vil ikke være i stand til at vælte under normale forhold.

For at kunne finde vej er der monteret en farvesensor foran, så robotten er i stand til at aflæse banens linje. Under robotten er der monteret en gyro, så robotten er i stand til at måle rotation.

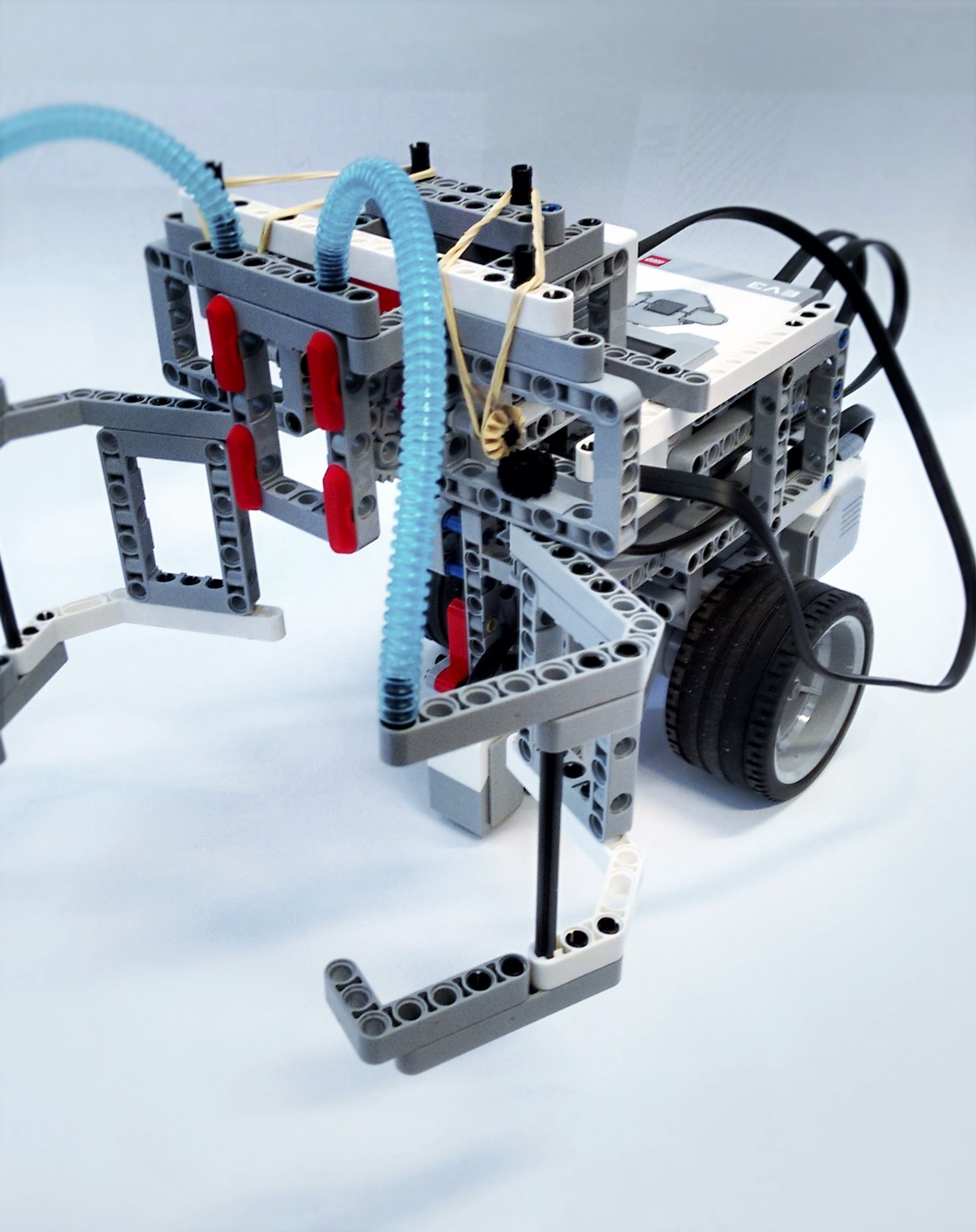
Placeringen af farvesensoren udgør et problem, da den ligger lige midt i omdrejningspunktet. Dette gør at når robotten skal søge efter linjen, ender en med at dreje rundt om sensoren. Dette gør at robotten har utrolig svært ved at lokalisere linjen. Så hvis robotten kommer for langt væk fra linjen har den ikke mulighed for at finde tilbage igen. Det optimale ville være at robotten kunne scanne dens omgivelser i en bred bue.

## Robot version 2

Version 2 af robotten er opbygget efter at skulle have et centreret omdrejningspunkt, for at farvesensoren kan scanne optimalt. Derfor er de to hjul placeret direkte på midten af robotten.

Et kugleleje er placeret foran, da det er der, der bliver lagt mest vægt på. Bagpå er der placeret en lille stang der skal forhindre robotten i at tippe bagover. Farvesensoren er stadig placeret foran, men på grund af den nye hjulplacering kan den scanne effektivt.

På grund af placeringen af hjulene er denne version utrolig effektiv til at dreje, og der skal derfor ikke bruges nær så meget kraft. Den kan derfor også scanne et meget større område, og kan derfor finde tilbage på linjen, hvis den kører forkert.



## Robot version 3

3. version er en direkte opgradering af version 2. Her er der placeret et kugleleje bagpå i stedet for et støtteben. Der er også placeret en anordning foran, som skal kunne gribbe fat i en flaske. Denne anordning bruger den lille motor.   
Farvesensoren er placeret tættere på jorden, da der var problemer med at aflæse refleksioner optimalt. Den største ændring er den nye gribbe anordning. Denne er lavet til at kunne gribbe fast i en flaske. Anordningen er drevet af den lille motor, som er placeret på toppen af robotten.

# Delbeskrivelse

## EV3 Brick

EV3 Bricken er en programmerbar computer som kan bruges til en lang række forskellige opgaver. EV3 er den 3. generation af Lego Mindstorm brick og er grundlaget for hele opgaven.

Koden til EV3 Bricken er i sproget C og den kan programmeres i en lang række forskellige programmer. Lige fra den basale løsning fra Lego selv til et mere avanceret program som RobotC. Til dette projekt koder vi i RobotC.   
**EV3 Specifikationer**

|  |  |
| --- | --- |
| Operativsystem | Linux |
| Processor | 300 MHz ARM9 |
| Flash Hukommelse | 16 MB |
| RAM | 64 MB |
| Skærmopløsning | 178x128 Sort og hvid |
| Tilslutningsmuligheder | USB 2.0, USB 1.1, Bluetooth |
| Porte | 4 Motorporte (Output) (A,B,C,D) 4 Sensorporte (Input) (1,2,3,4) |
| Strøm | 6 AA Batterier Alkaline eller Lithium Ion |
| Tilslutningskabler (?) | RJ12 – 6P6C (Telefonkabel) |

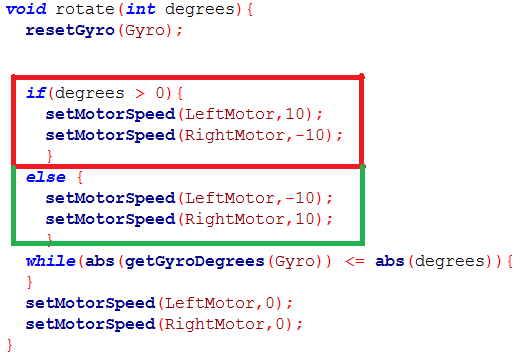
## Gyro

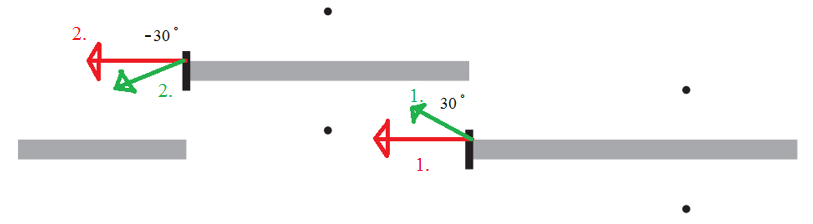
Gyroen fungerer som et kompas. Sensoren måler robottens rotationsbevægelse og kan bruges til at orientere sig.

I test(X), skal robotten skifte linje 2 gange. Her skal gyroen ved kontakt med en sortstreg(stoplinje), skifte til X antal grader, således at robotten styrer mod den næste streg.

Ved kontakt med en sortstreg skal robotten stoppe. Så skal gyroen nulstilles, sådan at den står på 0 grader i forhold til den sorte streg. Herefter skal gyroen vide om der skal rykkes i minus eller plus grader, så robotten kommer i den rigtige retning.

Der nulstilles og bestemmes retning afhængig af plus- eller minus grader:





Ved første missionsstreg (M1) stopper robotten og nulstiller gyroen. Fra denne udgangsposition(rød pil1) skal den rotere 30 grader(grøn pil1), her ved robotten at den skal rotere til højre, da graderne er større end 0(figur(x markeret med rødt).

Anden missionsstreg (M2). Gyroen nulstiller igen og der kommer en ny udgangsposition(rød pil2). Fra denne position skal robotten roterer -30grader(grøn pil2). Det kommer af koden, at når graderne er mindre end 0, roteres der til venstre(figurx markeret med grøn)

## Test 1

I første test var der nogle mindre problemer. Det første problem var at sensoren, som skulle detektere den sortelinje, ignorerede den. Sensoren er derfor sat længere ned mod jorden, hvilket løste problemet. Næste problem var at hjulene roterede for hurtigt, hvilket gjorde at gyroen ikke kunne følge med. Motorkraften ved rotation blev sat ned og det hele virkede.

Ved testene blev der gættet lidt på hvilke grader der skulle bruges. Det startede på 45grader ved første linjeskift, hvilket gjorde at robotten ramte en batteriport. Herefter er graderne sat til 30, hvilket fungerer perfekt.

## Motorer

De to store motorer er stærkere end den lille, men har derimod en lidt større responstid. De er beregnet til at køre og styre robotten. Det smarte med disse motorer er at de har en rotationssensor, som gør dem meget præcise. De kan altså køre samtidig, med så en god en præcision at de nemt kan køre ligeud. Den lille motor er hurtigere og fungerer derfor perfekt som ’’fangearme’’.

|  |  |  |
| --- | --- | --- |
|  | Lille motor | Stor Motor |
| RPM | 240-250 rpm | 160-170 rpm |
| Stå moment | 8 Ncm | 40 Ncm |
| Kører moment | 12 Ncm | 20 Ncm |

# Linjefølger

## ../../../Pictures/Robot/stregsmall.pngLinjen

Figure 1 - Grå linje på hvid baggrund med sort missionmarkør.

Ruten som robotten skal gennemføre, er indikeret med en grå streg på en hvid baggrund. Stregen er 5cm bred og på udvalgte steder afbrudt, af et tværgående sort rektangel. Rektanglet er 10cm x 2 cm og indikerer begyndelsen på en opgave, som robotten skal udføre. Linjen og missions indikatorerne er altså visuelle fingerpeg og kan derfor med fordel aflæses med en fotoelektrisk sensor, så som den inkluderede farve sensor, i et LEGO mindstorms EV3 kit. Farvesensoren bruges til at skelne mellem de tre nuancer, som banen er indikeret med.

## EV3 farvesensor

Farvesensormodulet består af en fotoelektrisk farvesensor og en RGB lysdiode. Modulet har tre forskellige måletilstande:

### Farvemåling

I denne tilstand belyser sensoren emnet, med hvidt lys fra sensorens indbyggede lyskilde. Derpå måles hvilke farver, der reflekteres tilbage til sensoren, som så processerer måleresultaterne og klargør dem til afsendelse over I2C protokollen.

I RobotC kan den målte farve præsenteres i to forskellige formater:

1. Som RGB værdi. I dette format vil hver af de tre farver repræsenteret, få en værdi mellem 0 og 255.
2. Som hue. I dette format bliver farven udtrykt ved en talværdi mellem 0 og 360.

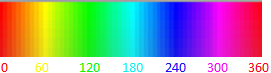


Figure 2 - Farvespektre som hue værdier

### Reflekteret lys fra egen lyskilde

I denne tilstand belyser sensormodulet emnet med et rødt lys. Derpå måles der hvor meget lys, der reflekteres tilbage fra emnet i det givne farvespektrum. Fordelen ved at belyse og måle med en grundfarve frem for f.eks. hvidt lys er, at målingen kan være mere resistent over for forstyrrelser. Den målte værdi er et tal mellem 0 og 100, hvor 0 er mindst refleksion og 100 er høj refleksion.

### Reflekteret lys fra omgivelserne

Den sidste tilstand måler sensormodulet, hvor meget af det omkringværende lys, der tilbagekastes fra emnet. Det vil sige at sensoren ikke leverer noget lys, fra sin egen lyskilde og er 100% afhængig af omkringværende lyskilder.

### Valg af måle metode

For at udvælge den bedst mulige målemetode, til at løse opgaven, er farvesensoren blevet testet på to forskellige overflader, med forskellige farver og nuancer. Den bedste metode er den, hvor de målte data klarest viser forskel på banens markeringer:

Gulvet i gangarealet udenfor grupperum B306, har påtegnet en blå figur meget lig med den endelige bane, robotten skal køre på. Gulvet er gradieret sort og grå. Farverne på gulvet er meget anderledes, end de på den endelige bane og kontrasten mellem baggrund og figuren, der skal følges er ringe. Det sidstnævnte er særdeles interessant, da det sætter farvesensoren på hård prøve. Hvis den vil være i stand, til at skelne mellem kontrasterne vil den sandsynligvis have gode odds, for at skelne mellem den skarpere kontrast på den rigtige bane.



Figure 3 - Gulvet uden for grupperum B306

Den opstillede testbane lavet udskrevet på A4 papirark. Denne banes farver og udformning er meget tæt på den endelige bane. Med andre ord er testbanen det tætteste på virkeligheden der kan kommes.



Figure 4 - Testbane af A4 papir

### Test af Målingsttilstande

Farvemålingstilstanden er først testet på gulvet i gangarealet og derefter på testbanen. Sensoren holdes vinkelret på overfladen, i en afstand af 5mm fra overfladen. Afstanden er valgt på baggrund, af den afstand LEGO foreslår i EV3-user guide[[1]](#footnote-1) og fordi det er den mindste afstand, der passer med monteringshuller i LEGO klodserne. Farven udlæses som hue værdi.

De andre målemetoder testes med samme fremgangsmåde, dog udlæses værdierne, som mængden af reflekteret lys på en skala fra 0 til 100.

#### Måleresultater

|  |  |  |  |
| --- | --- | --- | --- |
|  | Farverefleksion (hue) | Refleksion fra eget lys (0-100) | Refleksion fra omgivelser (0-100) |
| Gangareal Grå overflade | 60 | 20 | 1 |
| Gangareal Blå streg | 190 | 10 | 1 |
| Testbane Hvid overflade | 62 | 85 | 7 |
| Testbane Grå streg | 64 | 48 | 4 |

Figure 5 – Grafisk oversig over forskellige måledata

#### Konklusion på resultater fra test af farvesensor

Formålet af de foregående test, er at finde den sensor indstilling, som bedst egner sig til, at differentiere mellem banens hvide baggrund og de grå streger. Det vil sige at de målinger, som giver den største numeriske forskel mellem en streg og en baggrundsfarve, vil blive valgt som den bedste.

Farverefleksionen viser sig at være meget god, til at se forskel på gangarealets blå og grå farver. Dog er der meget ringe forskel på testbanens hvide og grå.

Refleksion fra eget lys giver en markant mindre forskel, på gangarealets blå og grå farver, men på testbanen ser resultatet meget bedre ud end farverefleksion.

Til sidst vises resultaterne for refleksion fra omgivelserne. Disse resultater viser sig at have så lille en forskel mellem linje og baggrund, at det ikke er nemt at differentiere mellem dem.

Baseret på resultaterne i foregående afsnit, besluttes det at bruge refleksion fra eget lys til den fremtidige konstruktion af robotten. Denne måle metode giver den bedste differentiering, mellem den grå streg og hvide bane. Desuden er metoden meget resistent over for udefrakommende lys, så længe lyset ikke er rødt.

# Planlægning af kørsel

Den overordnede strategi er at få robotten hele vejen gennem banen uden kritiske fejl. Der er derfor lagt mest vægt på at få en pålidelig robot til de mere simple opgaver. Derfor er de mere teknisk krævende opgaver valgt fra, da det er der hvor der er størst riskio for fejl. Vippen er den første opgave som er valgt fra, da der simpelthen er for stor risiki for at robotten kan falde ned og vælte. Og da vippen er så tidligt på banen, vil det kunne ødelægge chancen for at gennemføre resten. Derefter er målskiven blevet valgt fra, da der er for stor chance for at robotten kan kører forkert. Ented ved at sidde fast I indersiden af cirklen, eller vælge den forkerte vej rundt.

For at danne et bedre overblik over banen henvises der til kortet i billaget. Her er hver port, missionsstreg og flaske nummereret så det er nemmere at overskue.[[2]](#footnote-2)

Strategien for at komme igennen banen bliver fordelt imellem de forskellige missionsstreger (M1-M12.)

|  |  |
| --- | --- |
| M1-M2 | Robotten skal derfor finde M1,dreje 30 grader til højre og derefter kører fremad indtil den lokaliserer næste linje. Herfra skal den følge linjen frem til M2. |
| M2-M3 | Når M2 er identifiseret drejer robotten 30 grader til venstre og kører fremad indtil den rammer den originale linje. Herfra følger den linjen hen til M3. |
| M3-M4 |  |
| M4-M6 | Her skal robotten følge svinget rundt og ende ved M6. Vippen (M5) er blevet fravalgt da den udgør for stor en risiko. |
| M6-M7 | Når M6 er identificeret skal roboten dreje 20 til venstre og herefter ignorerer de første to grå kanter den kører hen over. Herefter drejer den 10 grader til højre for bedre at kunne finde den tredje kant. Ved den tredje grå kant skal robotten igen begynde at følge linjen og kører frem til M7. |
| M7-M9 | Her springes M8 over på grund af tidsbegrænsninger med henhold til kodning. Robotten kører derfor ligeud til M9. |
| M9-M10 | Når M9 er identificeret, drejer robotten 45 grader til højre. Herefter bruges enkoderne i motorerne til at måle en afstand. Når den ønskede afstand er målt drejer robotten 45 grader til venstre og finder tilbage til linjen. Herefter tager den svinget rundt og når til M10. |
| M10-M11 | Samme fremgangsmåde som i M9-M10, bortset fra at der her er et delay fra M10, da det ikke er muligt at dreje med det samme. |
| M11-M12 | Samme fremgangsmåde som i M9-M10, med undtagelse af at robotten her starter med at dreje til højre i stedet for venstre. |
| M12 | Når M12 er identificeret, kører robotten en given afstand som svarer til midten af pladen. Herefter stopper den og banen er gennemført. |

## Fejlhåndtering

Der er lavet en enkelt procedure til fejlhåndtering. Når robotten ikke kan lokalitere den ønskede farveforskel, begynder den at dreje rundt om sig selv i samme retning den havde før den mistede fokus. Når den så finder den ønskede farveforskel fortsætte den af linjen.

# Program

Programmet består af to tasks hvor den ene er main tasken og den anden er musik tasken. Derudover er det lavet adskillige hjælpefunktioner der primært kaldes fra main task.

## Main task

I main tasken ligger den basale navigations kode, der gør robotten i stand til at følge en linje og detektere sorte missionsmarkører. Hele navigations koden ligger i et uendeligt loop, der kaldes efter musik tasken er startet og farvesensoren er blevet kalibreret. Når navigationskoden begynder forventer programmet at robotten er placeret max 7,5 cm til højre for linjen.

Selve navigationsloopet er opdelt i 5 faser:

1. Initialisering af refleksionsmåling og nulstilling af timer1
2. Gå ind i loop der kører så længe refleksionsværdien er højere end skilleværdien. I loopet sættes de to hovedmotorer til at køre fremad med asynkron hastighed, hvor den højre motor kører hurtigst. Dette får robotten til at køre fremad samtidigt med at den drejer til venstre. Den egentlige effekt af loopet er at robotten vil køre fremad imens den langsomt søger ind mod stregen. Hver gang loopet kører læses der også op farvesensoren om refleksionen ændrer sig. Det sidste der bliver gjort inden loopet gentages er at kalde checkIfLost() funktionen. Når loopet afsluttes på grund af for lav refleksionsværdi nulstilles timer1 før næste loop påbegynddes.

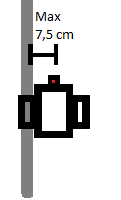


Figure 1 - Illustration af anbefalede robotplacering.

1. Dette loop køres så længe robotten er cirka midt på linjen. Det vil sige at så længe reflektionsværdien er ± 10 % af skilleværdien kører robotten lige fremad. Her bruges checkIfLost funktionen ikke. Når dette loop brydes bliver timer1 igen nulstillet før det næste loop begynder.
2. Når reflektionen er mindre end skilleværdien, men højere end stopLine værdien vil robotten antage at sensoren er over den grå streg. I det tilfælde vil den køre fremad mens den drejer langsomt til højre. Ellers fungere denne fase ligesom fase 2.
3. Den sidste fase er at tjekke efter om robottens farvesensor er over en sort missionsmarkering. Hvis dette er tilfældet vil robotten lægge en til stopLineCounts og derefter bruge denne globale variabel til at vurdere hvilken mission robotten er nået til.

### Musik task

Musik tasken består af et uendeligt loop der står for at der spilles musik så snart musik tasken er startet. Musik tasken vil dynamisk ændre musik nummer alt efter robottens status. Hvis robotten er i en fejltilstand, spilles der en mere dyster lyd, hvorimod der spilles en glad melodi mens robotten kører som den skal.

### Farvekalibrering – manualCallibColor()

For at sikre at robotten er I stand til at navigere I forskellige lysforhold skal farvesensoren kalibreres ved hver opstart. Når kalibreringsfunktionen kaldes, bliver der skrevet simple hjælpe instruktioner EV3 skærmen. Først skal robotten placeres så farvesensoren kan se den hvide baggrunds overflade, den målte refleksionsværdi vises på skærmen. Når værdien er stabil, trykkes der på enter knappen og programmet vil måle gennemsnitsværdien af overfladen over 10 samples, ved at kalde avgReflectedLight() funktionen. Når værdien er gemt, gentages det samme for den grå streg og de sorte missionsmarkører.

Når alle tre nuancer er blevet opmålt udregner programmet gennemsnitsværdien af den lyse værdi og den grå værdi. Dette gemmes som en skilleværdi der bestemmer om farvesensoren er over stregen eller ej.

Værdien for den sorte missionsmarkør gemmes i en global variabel. Denne vil senere blive brugt til at bedømme hvornår sensoren er over en missionsmarkør.

### CheckIfLost(float lostTimer, bool direction)

Denne funktion kontrollerer om robotten har set enten stregen eller baggrunden inden for de seneste 1500 ms. Hvis det ikke er tilfældet vil robotten stoppe med at føre fremad og begynde en stationær søgning efter stregen eller baggrunden. I søgningstilstand vil robotten dreje rundt om sig selv på stedet og fortsætte i den omdrejningsretning den kørte i før de 1500 ms er udløbet.

## Mission1 (M1)

Efter inspektion af banen, findes det fordelagtigt at få robotten til at dreje 30 grader for at ramme den brudte streg og ligeledes 30 grader for at komme tilbage på linjen.



Når robottens farvesensor opfanger, at den mængde lys der bliver reflekteret er mindre end eller lig med vores variabel stopLine, vil mission1 blive initialiseret.

## Mission6 (M6)

Opgave med flere brudte streger

Denne opgave minder meget, om den første opgave (M1) med en enkelt brudt streg. Det vil sige at den samme missionskode kan bruges igen.

Mission6 vil blive kaldet af missionsvælgeren. Mission6 kalder mission1 to gange i træk:

1. Første gang bliver mission1(-30, 200); hvor -30 er det antal grader robotten skal dreje om in egen akse, mod uret. 200 er et delay som får robotten til at køre forbi stregen. Dernæst kører robotten frem indtil den finder linje nummer to.
2. Når linje to er fundet, kaldes mission1(0, 200). Her er robotten allerede drejet i den ønskede orientering og den skal blot fortsætte indtil den tredje linje er fundet.

1. Side 12 LEGO Mindstorms EV3 User guide: [↑](#footnote-ref-1)
2. Nummereret kort bilag side (Fyld resten ud) [↑](#footnote-ref-2)