1 Robo-Camp 2019: Lektionsplan og Undervisningsmateriale

I dette kapitel vil den anvendte lektionsplanen samt undervisningsmaterialet fra efterårslejren "Robo-Camp" der blev afholdt i forbindelse med PANaMA projektets undervisningstilbud i form af to science camps (Crypto-Camp 2019, Robo-Camp 2019) der hver blev afholdt to gange i løbet af efteråret 2019, oktober 7. – 11., samt 14. – 18. blive gennem gået (se kapitel IV. Autumn-Camps i forskningsværksted Kiel, for en nærmere beskrivelse af efterårslejrene).

Lektionsplanen er opbygget i flere trin og faser, hvoraf underviseren er fri til at vælge enten at følge det hele, udvælge de dele der passer bedst ind i klassens nuværende niveau, eller at danne egne lektionsplaner ud fra det materiale der stilles til rådighed i forbindelse hermed. Lektionsplanen og gennemgangen heraf, skal derfor heller ikke ses som et endeligt facit på, hvordan et sådanne forløb skal afholdes, men som en inspiration til hvilke tematikker, samt informationer der evt. kan være relevante at inddrage, samt formidle. Der gøres opmærksom på at fordi efterårslejrene ikke var inddelt i klassiske undervisningsmoduler, er tiderne for de enkelte trin og faser i lektionerne, derfor heller ikke skarpt opdelt i moduler af 45min.

Det overordnede tema for lektionsplanen er robotter og automatisering, mens fokus ligger på at skabe en forståelse hos eleverne for, hvad en robot er og ikke er, samt give dem et grundlæggende kendskab til robotteknologi, herunder: Mikrocontrollere, sensorer og aktuatorer. Målet er at sætte eleverne i stand til bagefter, selv at anvende teknologien til udviklingen af egne robotter, samt til løsningen af konkrete problemstillinger.

Det udviklede materiale, hvoraf dele heraf vil blive præsenteret i nærværende lektionsplanen, kan i sin helhed findes på følgende link (bemærk at materialet for nuværende er på engelsk): http://www.teknologiskolen.dk/robocamp2019/

Note: I lektionerne tages der udgangspunkt i mikrocontrollerteknologien Arduino Uno, samt de sensorer og aktuatorer der indgik i forløbet. Dette betyder at de angivne enheder ligeledes tager udgangspunkt heri, og et digitalt signal vil derfor eks. blive beskrevet som værende mellem 0V (LOW) og 5V (HIGH), til trods for at dette er platformsbestemt. F.eks. operer både micro:bit og Calliope Mini, med en spænding på mellem 0V (LOW) og 3.3V (HIGH) når disse er forbundet gennem USB, hvorimod hvis de kører på en 3V batteriforsyning, er det følgende der er gældende, 0V (LOW) og 3V (HIGH).

1.1 Introduktion til robotter – Hvad er en robot? (70 min.)

Kort om indholdet: Lektionen er overvejende teoretisk, men med stor vægt på dialog med eleverne. Omdrejningspunktet er at få etableret en fælles forståelse for hvad en robot er/ikke er, samt hvilke kernekomponenter de består af, disses indbyrdes virke og relation til hverandre.

Læringsudbytte: Eleverne bliver bekendt med hvad mulig definition på en robot kan være, de bliver sat i stand til at identificere og analysere hvilke objekter der kan klassificeres som værende en robot, samt hvorfor. Derudover får de indsigt i hvilke interne kernekomponenter en robot indeholder, hvilken rolle de spiller i robotten, samt samspillet mellem dem. I forlængelse heraf, vil potentielle misforståelser vedr.

robotter, grundet moderne mediers portrættering af disse, ligeledes blive opklaret, samt evt. afmystificeres.

1.1.1 1. Fase – Den indledende samtale (30 min):

Forklar eleverne at førend man kan arbejde med og lave vores regne robotter, må man først vide hvad en robot er. Indled herefter samtalen om hvad en robot er, ved at spørge klassen indtil hvad de forstår ved ordet robot, samt hvad der udgør en sådanne.

Klassen præsenteres nu for en serie af billeder (eks. af humanoide-robotter, ikke-humanoide robotter, elektriske maskiner, samt ikke elektriske redskaber som evt. en cykel), hvortil eleverne for hvert billede spørges ind til hvorvidt objektet på billedet er en robot eller ikke, samt hvilken begrundelse der lægger til grund for svaret.

Herefter kan den officielle definition eller rettere mangel på samme bringes i spil, der findes nemlig ikke nogen endegyldig definition af, hvad en robot er, om end der er en generel konsensus vedr. en række parametre. En mulig definition herpå kan derfor være følgende:

"En programmerbar maskine, hvilken kan fungere autonomt, samt sanse sine omgivelser og tilpasse sine handlinger dertil".

Tag gerne en snak med eleverne om, hvordan man kan nedbryde ovenstående definition til brug for en analyse af, hvad en robot skal kunne, for at opfylde denne definition, dvs. en robot skal kunne:

- Sanse verdenen
- Handle i verdenen
- Tage en beslutning (hvordan der skal handles, i forhold til det sansede)

Med de tre krav på plads, kan samtalen herefter naturligt lede over i en snak om, hvordan robotter kan opfylde ovenstående krav, hvor eleverne spørges ind til nedenstående spørgsmål:

Den kan sanse verdenen:

-Hvordan sanser vi verdenen? (syns, høre, smage, lugte, følesans).

-Hvordan kan en robot sanse verdenen? (den har brug for mindst én sensor, eks. en lys-sensor).

Den kan handle i verdenen:

-Hvordan handler vi i verdenen? (muskler).

-Hvordan kan en robot handle? (den har brug for mindst én aktuator, eks. en DC motor)

Den kan tage en beslutning:

-Hvordan tager vi en beslutning? (hjernen).

-Hvordan kan en robot tage en beslutning? (den har brug for mindst én mikrocontroller – eks. en Arduino).

Herved er de tre kernekomponenter der i vores definition, udgør en robot og som en sådanne skal besidde mindst én af hver af, identificeret: Sensorer, aktuatorer og mikrocontrollere.

Mulig definition af en robot: *"En programmerbar maskine, hvilken kan fungere autonomt, samt sanse sine omgivelser og tilpasse sine handlinger dertil".*

Kernekomponenter: Mikrocontrollere, sensorer og aktuatorer.

1.1.2 2. Fase – Kernekomponenterne, deres virke og samspil (30 min):

Efter at have identificeret de tre kernekomponenter, kan disse herefter gennemgås én ad gangen for at forklare hvad de gør og hvordan de spiller sammen. I og med at mikrocontrolleren styrer sensorerne såvel som aktuatorerne er denne et godt sted at begynde.

1.1.2.1 Hvad er en mikrocontroller?

En mikrocontroller er basalt set, en lille computer. Den har en processer (regnekraft), RAM (hukommelse) og andre nødvendige komponenter, præcist som en computer har. Mikrocontrollere findes i dag, i næsten alle former for elektrisk udstyr: I sodavandsautomaten, fjernsynet, mikrobølgeovnen, vækkeuret osv.

Mikrocontrolleren råder over en række ind og udgange, dens såkaldte I/O pins (Input/Output). Mikrocontrollerens I/O pins, kan programmeres til enten at måle hvilken spænding der er på dem, eller til selv at forsyne andre komponenter hermed.

Ofte vil man anvende mikrocontrolleren til at aflæse hvilken spænding en sensor forsyner en given I/O pin med, hvorefter den vil foretage et valg baseret herpå, for derefter at forsyne en aktuator med en spænding der udløser en tiltænkt handling, i overensstemmelse med dette valg.

Det signal mikrocontrolleren modtager fra en sensor kaldes INPUT, mens det signal mikrocontrolleren sender til en aktuator kaldes OUTPUT. INPUT signaler kan enten aflæses som digitale (LOW (0V) eller HIGH (5V)), eller analoge (fra 0 (0V) til 1023 (5V)), mens OUTPUT signaler kan skrives som digitale (LOW (0V) eller HIGH (5V)) eller PWM (Pulse-width modulation) (fra 0 (0V) til 255 (5V)) hvilket er en simulering af et analogt signal.

Mikrocontroller: En lille computer der indeholder programmerbare I/O pins (Input/Output), eks. en Arduino Uno. Mikrocontrollerens rolle er at afvikle det program, programmøren har overført til den og den er derfor så at sige ”hjernen” i robotten.

Mikrocontrolleren vil ofte blive programmeret til at modtage et analogt/digitalt INPUT signal (0V – 5V), behandle signalet og foretage en beslutning om hvordan der skal handles herpå, hvorefter et digitalt/PWM OUTPUT signal (0V – 5V) forsyner en tilsluttet aktuatorer, der udfører denne handling.

Digitale signaler (input og output**):** Digitale signaler er enten 0V eller 5V.

Analoge signaler (input**):** Analoge signaler har en værdi der befinder sig et sted mellem 0V og 5V og læses af Arduinoen som et tal mellem 0 (0V) og 1023 (5V).

PWM signaler (output**):** PWM signaler (Pulse-width Modulation), er en simulering af analoge signaler og skabes ved at en I/O pin, i et tilpasset interval, svinger mellem skiftevis at være 0V og 5V. Svingningerne foregår så hurtigt, at signalet i praksis kan anvendes som et analogt signal. I Arduinoen skrives et PWM-signal som et tal mellem 0 (0V) og 255 (5V).

1.1.2.2 Hvad er en sensor?

Der kan med fordel begyndes med, at eleverne spørges om, hvilke eksempler på sensorer de kender til i forvejen, hvorefter en eksempelliste fremvises, eks. indeholdende: Kontakter, lyssensorer (LDR - Light

Dependent Resistor), afstandssensorer (ultralydssensor), farvesensorer, bevægelsessensorer, mikrofoner, temperatursensorer, infrarødsensorer (IR-Sensor) osv.

Dialogen kan nu drejes over på, om man kan finde en ting der er fælles for dem alle? Navnlig at de registrerer ændringer i det miljø de befinder sig i.

I samme ombæring kan der kommes ind på om robotter kan have sensorer, der kan opfange ting mennesker ikke kan. Kan mennesker eks. høre ultralyd, se infrarødt lys mm.?

Herefter kan der følge en kort gennem gang af, hvordan sensorer overordnet set anvendes: Nogle sensorer sender konstant et signal tilbage til mikrocontrolleren (eks. LDR), mens andre først skal aktiveres førend de sender et signal tilbage (eks. afstandssensoren). Fælles er dog, at mikrocontrolleren modtager dette signal som et INPUT, der så kan aflæses og fortolkes.

Samtidigt vil langt de fleste sensorer eleverne kommer til at arbejde med, returnere et analogt signal på 0V – 5V, hvilket oversættes af mikrocontrollerens 10-bit ADC (analog-to-digital converter) til et tal på 0-1023 (LDR, Temperatur, Potentiometer, IR-Sensor mfl.), hvorimod eks. afstandssensoren måler tiden der går mellem at et INPUT signal går fra HIGH (5V) til LOW (0V).

Afslut gerne ved at vise billeder af de forskellige sensorer eleverne kommer til at arbejde med i undervisningen, sådan at de allerede nu bliver visuelt bekendt med hvordan disse ser ud.

Sensor: Sensorer registrerer ændringer i det miljø de befinder sig i, dette kan eks. være interaktionen med en kontakt, ændringer i lysniveauet (LDR), lydniveauet (mikrofon), sensorens afstand til det nærmeste objekt (ultralydssensor) osv.

Signal: Mikrocontrolleren modtager sensorens værdi, som et INPUT signal, hvilket enten aflæses analogt eller digitalt.

1.1.2.3 Hvad er en aktuator?

Som ved sensorer, kan eleverne spørges ind til hvilke eksempler de kender til i forvejen, hvorefter en eksempelliste kan fremvises, eks. indeholdende: motorer (DC, servo, stepper), LED'er (Light Emitting Diodes), højtalere, varmeelementer osv.

Herefter kan fællestrækket ved disse identificeres, navnlig at de påvirker det miljø de befinder sig i, mens der kan kommes ind på at eks. infrarøde LED'er og ultralydshøjtalere, kan udføre handlinger som mennesker ikke er i stand til.

I modsætning til sensorer der sender et signal til mikrocontrolleren, hvilket denne modtager som et INPUT, så er det i denne forbindelse mikrocontrolleren der her sender et OUTPUT, hvilket aktuatoren modtager. I forbindelse med de aktuatorer eleverne forventes at arbejde med i forløbet, er signalet der her anvendes som OUTPUT, enten digitalt (LOW (0V) eller HIGH (5V)), eller PWM (Pulse-width Modulation) (fra 0 (0V) til 255 (5V)).

Aktuator: aktuatorer påvirker det miljø de befinder sig i, dette kan eks. være gennem fysisk bevægelige dele såsom motorer (DC, servo, stepper), men også gennem ændringer i lys-niveauet (LED), lyd-niveauet (buzzer, højtaler), temperaturen (varme-elementer) osv.

Signal: Mikrocontrolleren sender kommando til aktuatoren, gennem et OUTPUT signal, hvilket enten er digitalt eller PWM.

1.1.3 Opsummering:

Opsummer nedenstående og gennemgå herefter den samme billedserier fra tidligere, i plenum sammen med eleverne. Diskuter endnu engang, hvorvidt objekterne på billederne er robotter eller ej, opfylder de vores krav til en robot? Kan de tre kernekomponenter identificeres (vær obs. på at mikrocontrolleren ofte er skjult, hvorfor man må nøjes med at formode at den er der)? Opsummeringen kan med fordel gentages flere gange gennem forløbet, for derved at træne eleverne i at identificere og analysere robotter, samt genopfriske samspillet mellem mikrocontrollere, sensorer og aktuatorer. Anvend derfor gerne nye billedserier for hver gang, samt objekter fra elevernes hverdag.

Definition:
"En programmerbar maskine, hvilken kan fungere autonomt, samt tilpasse dennes opførsel på omgivelserne".

Dvs. en robot kan sanse omverdenen, tage en beslutning, samt handle i overensstemmelse dermed (evt. handle derefter):

Kernekomponenter:
En mikrocontroller (tager beslutninger)
En sensor (registrerer ændringer i miljøet)
En aktuator (udfører ændringer af miljøet)

Oversigt:

	Mikrocontrollere:	**Sensorer:**	aktuatorer**:**
Egenskab:	Tager beslutninger. Lytter til sensorer og kommanderer aktuatorer.	Registrerer ændringer i det miljø de befinder sig i.	Påvirker det miljø de befinder sig i.
Signal:	Modtager et INPUT fra sensorer. Sender et OUTPUT til aktuatorer.	Sender et INPUT til mikrocontrolleren, hvilket aflæses enten analogt eller digitalt.	Modtager et OUTPUT fra mikrocontrolleren, hvilket enten er digitalt eller PWM.

1.1.4 3. Fase – Hvad er en robot ikke (10 min):

Den for kurset gældende definition på hvilke krav et objekt skal opfylde, førend vi kan kalde det for en robot er nu blevet defineret, samtidigt med at der også har været en kort gennemgang af, hvordan kernekomponenterne fungerer og spiller sammen i relation til hinanden. Men det er ofte mindst lige så vigtigt at vide, hvad en robot ikke kan, som hvad den kan. Dette er eks. meget relevant når mennesker arbejder sammen med robot, eks. i industrien, for robotten tænker ikke på dig og tager derfor heller ikke hensyn den ikke er programmeret til at tage.

Spørg eleverne i plenum, om de har nogle ideer om hvad en robot ikke er/ikke kan, selvom de ligesom os mennesker, kan sanse verdenen, handle i verdenen, samt tage en beslutning?

Nogle gode emner at komme ind på, er eks. at robotter ikke er i stand til at have emotionelle følelser. Denne dialog kan med fordel startes ved at eleverne først præsenteres for to robotter, en "cute" robot a la. NAO, PARO etc., samt en industriel robot, eks. en svejsemaskine. Herefter kan eleverne spørges ind til, hvilken af de to robotter der er i stand til at have de største empatiske følelser. Ingen af robotter er selvfølgeligt i stand hertil, men eleverne vil igennem deres liv, blive mødt med robotter der er designet til at indikere at de besidder empatiske følelser, ligeledes vil de gennem moderne medier, ofte blive præsenteret for robotter som værende i stand hertil.

Det er samtidigt vigtigt at slå fast overfor eleverne, at robotter ikke er selvbevidste og ikke er intelligente i klassisk forstand – igen, uagtet hvad film og andre medier fremstiller dem som værende. Forklar eleverne at de udelukkende operer på baggrund af et logisk kredsløb, dvs. alle deres beslutninger udelukkende er baseret på matematiske beregninger, hvilket de til gengæld er lynenes hurtige til at udføre. Men af samme årsag, skulle man derfor afsætte tilstrækkeligt med tid dertil, vil man ved at gennemgå alle beregningerne med papir og blyant, altid komme frem til den samme beslutning som robotten.

Af samme årsag er robotter heller ikke hverken reflekterende, intuitive eller kreative. De gør præcist det de er programmeret til, og intet andet. Robotter kan derfor eks. ikke have en dum dag, være træls eller ikke gide gøre det vi siger til dem. Som et eksempel kan man tage en robot der er programmeret til at køre lige ud. Sætter denne robot nu op på et bord, hvad sker der så? Den kører direkte udover kanten uden på noget tidspunkt at have haft en mening herom, eller på nogen måde reflektere over konsekvenserne herved, hverken før eller efter. Den stopper altså ikke intuitivt op, på samme måde som vi mennesker ville gøre det. Dette kan nogle gange være svært for eleverne at forstå og det er derfor vigtigt at slå det fuldstændig fast, at robotter udelukkende gør det de er programmeret til, hverken mere eller mindre.

Robotter er ikke: Intelligente i klassisk forstand, intuitive, reflekterende og selvbevidste eller i stand til at have følelser.

Robotter tager beslutninger baseret på matematik og logik: Hvis man derfor afsætter tilstrækkeligt med tid, vil man med papir og blyant, altid komme frem til den samme beslutning som robotten.

1.2 Introduktion til robotter – Hvorfor er det så svært at lave robotter? (30 min.)

Kort om indholdet: Lektionens omdrejningspunkt er en fysisk aktivitet, hvor elever i grupper af tre, skal interagere sammen og gøre det ud for en robot, ved at simulere dennes tre kernekomponenter: Mikrocontrolleren, sensorerne og aktuatorerne.

Læringsudbytte: Eleverne bliver gennem fysisk aktivitet bekendt med de grundlæggende interne kommunikationskanaler der udgøres af samspillet mellem mikrocontrolleren, sensorerne og aktuatorerne (INPUT og OUTPUT). Samtidigt lærer de om robotters begrænsninger, hvilket sættes i relation til vores intuitive måde at agere på i ukendte miljøer.

Fysiske materialer pr. gruppe (3 elever pr. gruppe): 2stk. stof der kan anvendes som bind for øjnene, samt 3stk. ens farvede plastikkrus eller lign. objekter.

1.2.1 1. Fase – Introduktion, Regler, Opsætning, Udfordringer og Mål (12 min):

Introducer eleverne til hvorfor det er så svært at lave robotter, selvom de indtil videre langt hen ad vejen, kan sammenlignes med mennesker (hjernen:mikrocontrolleren, sanser:sensorer, muskler: aktuatorer)?

En af de grundlæggende forklaringer er selvfølgeligt relaterbar til den afsluttende snak i sidste lektion om, hvad robotter ikke er. Tænk blot på hvor ofte man vi foretager os eks. en intuitiv handling, blot ved at gå rundt i skovbunden.

Spørg evt. eleverne hvilken robot de tror er det nemmeste at lave: En robot der kan udføre 1000vis af ting i et kontrolleret miljø, eller en robot der kan udføre 1 ting i et ukontrolleret miljø? Forklar herefter at det nemmeste er det kontrollerede miljø, fordi alle variabler der indgår derved er kendt på forhånd, hvorfor robottens manglende intelligens og evne til at reflektere, samt tænke intuitivt ikke længere problematisk i samme grad.

Samtidigt er vores kommunikation mellem hjernen, sanserne og musklerne, forfinet gennem millioner af års naturlig udvikling, hvorfor det nu er så indlejret i vores system at vi ikke længere, aktivt skal tænke over hvordan inputtet fra vores sanser skal fortolkes, eller hvordan outputtet til vores muskler skal formuleres.

Efter introduktionen til problemstillingen, introduceres eleverne til aktivitetens regler, opsætning samt udfordringer og mål, se nedenstående:

Problemstilling: Som levende væsner er vores hjerne, sanser og muskler gennem millioner af års naturlig evolution, blevet fintunet til at arbejde sammen, dette gør sig ikke gældende for robotter. Samtidigt er robotter ikke intelligente i klassisk forstand eller i stand til at reflektere og foretage intuitive handlinger. Derfor er det også utroligt svært at lave robotter, fordi deres mangler gør, at de ikke automatisk kan tilpasse sig ukendte variabler.

Regler:

- Tre elever udgør tilsammen én robot (mikrocontroller, sensor og aktuator):
- Mikrocontroller (har bind for øjnene):
    - Kan tage beslutninger.
    - Kan spørge sensoren efter information og lytte til svaret.
    - Kan give aktuatoren kommandoer.
- Sensoren:
    - Kan sanse verdenen og svare mikrocontrolleren, når denne bliver spurgt.
- Aktuatoren (har bind for øjnene):
    - Kan lytte til kommandoer fra mikrocontrolleren, samt udføre dem.

Opsætning:

- Udpeg et gruppebord for hver gruppe og placer dem foran dette.
- Efter at mikrocontrolleren og aktuatoren har fået bind for øjnene, fordeles de tre plastikkrus i ”robottens” nærmiljø, gerne i forskellige højde niveauer (under en stol, på nabobordet og lign.).

Udfordringer og mål:

- Lokaliser de tre plastikkrus.
- Indsaml de tre plastikkrus.
- Stabl de tre plastiskkrus på det udpegede bord.

1.2.2 2. Fase – Aktiviteten (18 min):

Inddel eleverne i grupper af tre elever pr. gruppe, hvorefter aktiviteten kan påbegyndes. Herefter skal eleverne nu i virke af den robot de udgør, forsøge at arbejde sammen om at få løst udfordringen og nå i mål med at stable plastikkrussene på deres respektive borde. Når målet er nået, roteres der og eleverne i gruppen tildeles en ny rolle i robotten, dette gentages tre gange indtil alle tre elever, har forsøgt sig med alle tre roller (mikrocontroller, sensor, aktuator).

1.3 Introduktion til Arduino - Lær Arduino systemet at kende (6 timer og 30 minutter):

Kort om indholdet: Lektionen blander teori med praktiske og konkrete hands-on øvelser, vedr. forståelsen for og anvendelsen af en række udvalgte sensorer og aktuatorer, samt grundlæggende programmerings principper. De programmeringsprincipper, sensorer og aktuatorer der heri gennemgås, indgår alle i lektionsplanens afsluttende temaprojekt.

Læringsudbytte: Eleverne opnår en basal forståelse for elektricitet, mikrocontroller teknologi (Arduino Uno), programmeringsprincipper, samt sensorer og aktuatorer. Derudover bliver de sat i stand til selv at kunne arbejde med og konstruere simple input/output systemer.

Fysiske materialer pr. gruppe (2 elever pr. gruppe): Lektion indbefatter et større udvalg af forskellige materialer og komponenter, den fulde liste kan findes på hjemmesiden der linkes til i toppen af afsnittet. Det samme kan de udviklede cheat sheets, hvoraf dele er inkluderet i dette afsnit, samt eksempler på programmer til de forskellige sensorer og aktuatorer, såvel som løsningsforslag til de forskellige opgaver. De udviklede cheat sheets, kan med fordel printes ud og uddeles allerede i første fase af denne lektion.

Note: Efter hver fase er der givet et par eksempler på opgaver, der lader eleverne eksperimentere med de gennemgåede opstillinger. Det anbefales derfor, at hver fase påbegyndes ved at de anvendte materialer deri udleveres til eleverne, sådan at de løbende kan kopiere disse.

1.3.1 0. Fase – Introduktion til det videre forløb (10 min):

Introducer eleverne til det afsluttende temaprojekt, for derved at sætte de kommende lektioner i en konkret kontekst, med det formål at fjerne en del af det abstrakte og ukonkrete element heraf. De følgende faser kan ligeledes med fordel – hvor det er muligt – præsenteres i relation til temaprojektet. Dette således at eleverne eks. bliver bekendt med at transporteringsrobotten anvender IR-sensorer, samt hvorfor og hvor de er placeret, førend de går i gang med at lære om disses anvendelse i relation til mikrocontrolleren.

1.3.2 1. Fase – Introduktion til elektricitet og aktuatorer (LED) (20 min):

Introducer eller genopfrisk basalt kendskab til elektricitet og hvordan strøm flyder i elektriske kredsløb, forklar kort hvad strøm, spænding og modstand er. De nærmere udregninger kan udelades.

Tegn et diagram for en strømkilde (5V), der forsyner en LED (Rød – 5mm), ink. formodstand 150Ω og gennemgå dette for eleverne.

Introducer dernæst breadboardet og hvordan dets huller hænger internt sammen, vis gerne illustrationer heraf. Konstruer kredsløbet på breadboardet, mens Arduinoen kan agere strømforsyning hertil, ved at anvende dennes 5V og GND pins.

Ved anvendelsen af en aktuator (LED), er ét af de tre krav til en robot, hermed opfyldt.

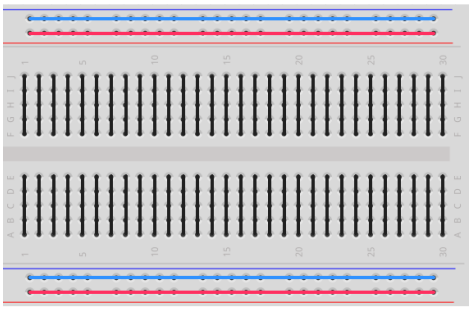


Fig. 1: Oversigt over breadboardets interne forbindelser.

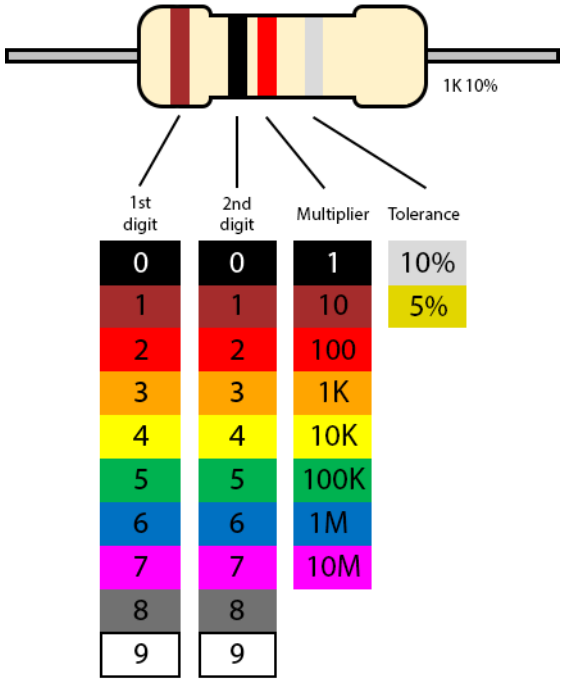


Fig. 2: Oversigt over hvordan farvekoderne på modstande aflæses.

1.3.3 2. Fase – Introduktion til Arduinoen (45 min):

Introducer Arduinoen og gennemgå overordnet dens funktionalitet, hvad er pins, hvad er deres funktion mm. samt introducer dem til Arduino programmeringsmiljøet.

Anvend nu Arduinoen, til at kontrollere LED'en fra tidl., fremfor at den blot agerer strømforsyning. Dette kan med fordel gøres på en projekter, så eleverne kan følge med på deres egne computere og programmere programmet samtidigt med underviseren. Start med at introducere hvad variabler er og forklar herefter hvordan programstrukturen fungerer, sæt dernæst den tilkoblede pin til OUTPUT i setup, mens den i loop sættes til HIGH. Gennemgå igen forskellen på INPUT og OUTPUT, hvorfor LED'en sættes til OUTPUT, samt forskellen på LOW (0V) og HIGH (5V). Det kan også være en god ide, i samme ombæring kort at komme ind på de mest anvendte datatyper (int, String, boolean), i forbindelse med at den tilkoble pin, gemmes som en int.

Mulig analogi på variabler: En god analogi kan være at sammenligne dem med de flyttekasser mange har stående ude i garagen, hvorpå der er skrevet en label og en ting er kommet deri. Nu kan den ønskede kasse altid findes frem igen og det i lagte tilgås, ligesom det kan udskiftes med noget andet. Forklar at variabler derfor er en metode til at gemme ting i mikrocontrollerens hukommelse, så de senere kan findes frem igen.

Når alle elever har fået LED'en til at lyse, kan denne sættes til at blinke ved at lade den tilkoblede pin veksle mellem HIGH og LOW, men uden at der er indsat et delay. Eleverne vil hurtigt opdage at de ikke kan se nogen forskel på før og nu. Dette er en fin indgangsvinkel til at lade dem opleve hvor hurtigt en processor kan udføre beregninger og handlinger baseret derpå. Introducer delay funktionen, men spørg først eleverne hvor denne skal indsættes. I mange tilfælde vil svaret blive, at det er mellem pin'en sættes til HIGH og LOW. Hvis dette er tilfældet, leder det videre til endnu en god øvelse for eleverne, for som før vil de igen ikke se nogen nævneværdig forskel som effekt heraf. En elev kan med fordel bedes om at blinke med lyset i klasseværelset i intervaller af ca. 2 sekunder, samt forklare hvad denne gør, forud for hver handling: Tænd, Vent, Sluk, Vent, Tænd osv. Bed derefter eleverne gennemgå koden i programmet: Tænd, Vent, Sluk, Tænd osv. Her vil de med stor sandsynlighed hurtigt opdage at ved anvendelsen af et loop, er der ingen ventetid mellem overgangen fra den sidste linje, til den første linje, hvorfor der ligeledes skal indsættes et delay efter at pin'en sætte til LOW.

Opgaver til eleverne:

- Eksperimenter med blink hastigheden. Hvor ligger overgangen i millisekunder, fra at øjet kan registrere at denne blinker og til at det ligner at den er tændt konstant.
- Eksperimenter med at få to LED'er til at blinke enten samtidigt, eller skiftevis.

Vis hvordan der ved at justere på størrelsesforholdene på de indsatte delays, kan simuleres hvordan LED'en eks. ser ud som om den tændt på halv styrke. Dette leder over til en samtale med eleverne om forskellen på digitale, analoge og PWM (Pulse-width Modulation) signaler.

Gennemgå forskellene på digitale og analoge signaler og forklar samtidigt eleverne at Arduino'en kun kan læse analoge signaler, men ikke generere dem. Arduinoen anvender i stedet et simuleret analogt signal, kaldet PWM. Forklar dem at et PWM-signal basalt set det samme, som det de selv konstruere lige før, gennem manipulationen af størrelsesforholdet på de anvendte delays (se Figur 3).

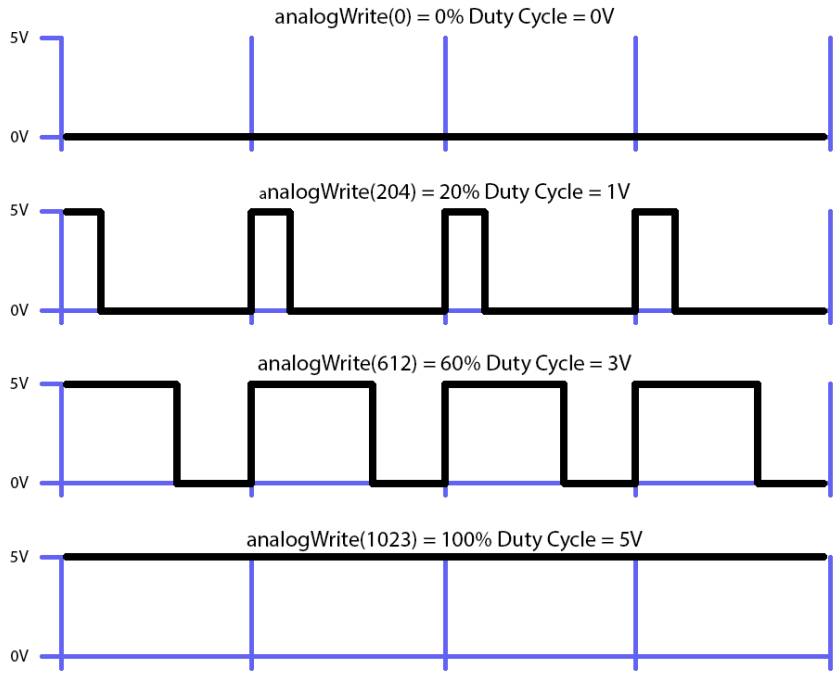


Fig. 3: Illustration af hvordan et PWM-signal, simulerer et analogt-signal.

Arduino Pins:

```
analogRead:    A0-A5
digitalRead:   A0-A5, 2-13

analogWrite:   3, 5-6, 9-11
digitalWrite:  A0-A5, 2-13

Used by the R3 Motorshied:
A0-A1, 3, 8-9, 11-13
```

Datatypes:

```
int:      -32768 to 32767
          LOW or HIGH
          A0-A5 (analog pins)
          2-13 (digital pins)

bool:     true or false
          0 or 1
          LOW or HIGH

String: "text text"

void:     no data
```

Program Structure:

```
//variables can be initialized
//here

//runs one time only
void setup(){
  //setup pins and enable serial
  //here
}

//runs repeatedly
void loop() {
  //write code for the program
  //here
}
```

Fig. 4: Oversigt over Arduinoens pins, samt de mest brugte datatyper og Arduino programstrukturen.

Variable (with a value):

```
//datatype name = value
int varName = 13;

//datatype name = value
bool varName = true;

//datatype name = value
String varName = "Hello!";
```

Delay:

```
void loop() {
  //pause for 1000 milliseconds
  delay(1000);
}
```

Fig. 5: Oversigt over hvordan variabler og delays kan anvendes.

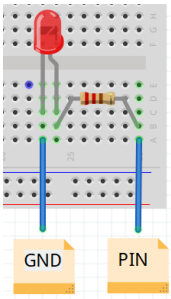
LED: 5mm Red LED (resistor size: 150Ω)

```
int LEDPin = 2;                 //the used pin

void setup() {
  pinMode(LEDPin, OUTPUT);      //set pin as OUTPUT
}
```

```
void loop() {
  //turn the LED off (LOW) or on (HIGH) *
  digitalWrite(LEDPin, HIGH);

  //turn it gradually from off (0) to on (255) **
  analogWrite(LEDPin, 180);
}
```



GND: Short leg
PIN: Long leg

*PIN: 2, 4-7, 10, A2-A5
**PIN: 5, 6, 10

Fig. 6: Eksempel på hvordan en LED kan anvendes.

1.3.4 3. Fase – Introduktion til sensorer (kontakt) (25 min):

Introducer kontakten og gennemgå for eleverne hvordan den virker og kan ses som en knækket ledning, der når den trykkes sammen, skaber forbindelse igennem sig.

Lav et kredsløb med den på breadboardet og tilslut det til Arduinoen, lad samtidigt eleverne kopiere dette. Gennemgå kort hvorfor der det er nødvendigt at anvende en modstand i forbindelse hermed, for ikke at brænde kredsløbet af. Forklar at fordi kontakter er sensorer, skal de i setup delen af Arduino koden, angives som INPUT. Anvendelsen af INPUT, samt en modstand ifm. kontakten, giver en naturlig introduktion til spændingsdelere og konstruktionen af egne sensorer (mere om det i de følgende afsnit). Til senere forløb er det værd at bemærke at den anvendte modstand ikke er strengt nødvendig, fordi der i stedet kan anvende Arduinoens indbyggede pull-up modstande, hvilke aktiveres ved at udskifte ”INPUT” med ”INPUT_PULLUP”.

Lav et nyt program, hvori kontaktens værdi aflæses. Udelad i første omgang alt seriel kommunikation (Serial.begin(9600); og Serial.println(switchValue)), sådan at der kun er analogRead(switchPin). Tag herefter en snak med eleverne om, hvordan man kan få indsigt i hvad mikrocontrolleren ”tænker”, for selvom der er indsat en analogRead kommando, kan de pt. ikke se værdien noget sted. Det er ikke utænkeligt at en eller flere af dem, evt. fra tidl. undervisning eller selvstændige projekter hjemmefra, er bekendt med print-statements.

Introducer herefter serial monitoren, samt hvordan seriel kommunikation aktiveres og print statements anvendes, til at få printet variablers værdier, samt egne tekstbeskeder til monitoren. Forklar hvordan print statements udgør et vigtigt redskab ifm. debugging, fordi dette er den mest umiddelbare metode til at få indsigt i hvilke dele af programmet der udføres, samt hvilke værdier der opereres med.

Ved anvendelsen af en sensor (kontakt), er to af kravene til en robot, hermed opfyldt. Introducer derfor og gennemgå hvordan betingelser kan anvendes til at lade mikrocontrolleren tænde/slukke for LED’en, alt efter om kontakten er trykket ned eller ej, hvorved det tredje og sidste krav er blevet opfyldt.

Alt efter typen af switch, men den godt stå og svinge et par gange mellem at være oppe og nede, når denne skifter stadie. Det kan derfor ofte være en god ide at indsætte et ganske kort delay, på nogle få millisekunder.

Opgaver til eleverne:

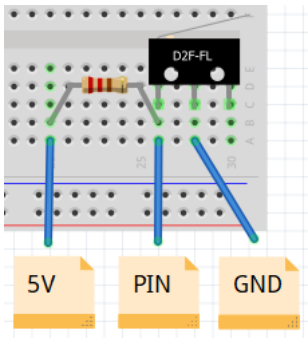
- Eksperimenter med hvilken størrelse et delay potentielt skal have, førend svingningerne filtreres fra, så kontakten ikke registreres som værende trykket ned flere gange end den reelt er.

Switch: Resistor size: 1KΩ

```
int switchPin = A5;           //the used pin
int switchValue;              //to store the value

void setup() {
  pinMode(switchPin, INPUT);  //set pin as INPUT
  Serial.begin(9600);         //enable serial
}

void loop() {
  //read and store the value, then prints it
  switchValue = digitalRead(switchPin);
  Serial.println(switchValue);
}
```



*PIN: 2, 4-7, 10, A2-A5

Fig. 7: Eksempel på hvordan en kontakt kan anvendes.

Conditional (Structure):

```
void loop() {
  if (x == y) {
    //if yes, run code
  }
  else if (x < 4 && y < 4) {
    //if yes, run this instead
  }
  else {
    //else, run this instead
  }
}
```

*there can only be one "if"
*there can be multiple "else if"
*there can only be one "else"

Conditional (Operators):

```
==   equal
!=   not equal
>    larger
<    smaller
>=   larger or equal
<=   smaller or equal

&&   and
||   or
```

Fig. 8: Oversigt over hvordan betingelser kan anvendes.

1.3.5 4. Fase – Forklaring af spændingsdelere og Sensorer (LDR) (65 min):

Spændingsdelere danner grundlaget for mange sensorer og det er ligeledes en nem måde at konstruere vores egne på. Introducer derfor spændingsdelere og hvordan forholdene mellem modstandene spiller sammen.

Mulig analogi på variabler: En god analogi til formålet, er at spændingen kan ses som et fad med fem kager (en kage for hver de 5 volt). De to modstande kan derfor sammenlignes med to personer. Tilføj kravet at alle kagerne skal være spist, efter at fadet har passeret den sidste person. Det er vigtigt at understrege at det derfor ikke afgørende hvor sultne personerne er, for alle fem kager skal spises

uanset hvad. Forklar at det der derimod er afgørende, er forholdet mellem hvor sultne de er. Hvis de er lige sultne i forhold til hinanden, deles kagerne ligeligt mellem dem. Mens hvis den første person er meget sulten i forhold til den anden, vil han derfor spise flest kager, mens den anden ikke spiser ret mange og omvendt.

Efter at fadet med kager har passeret den første person, kan man nu se hvor mange kager der er tilbage på fadet, dette er vores indikator på forholdet mellem de to personernes sult, fordi man ved at den anden person uanset hvad, skal spise de resterende.

Efter at have introduceret og forklaret hvordan spændingsdelere fungerer, kan nedenstående øvelser herefter gennemgås og laves (kontakten tages blot ud af kredsløbet og erstattes herefter af en modstand – derefter er opstillingen den samme som fra tidl.). Repetere gerne at analoge værdier af Arduinoen læses som et tal mellem 0 (0V) og 1023 (5V).

Modstand 1:	Modstand 2:	Aflæst værdi (Arduino):	Værdi i volt:
1KΩ	1KΩ		
560Ω	560Ω		
820Ω	180Ω		
180Ω	820Ω		
1KΩ	Uendelig (kontakt oppe)		
1KΩ	Ingen (kontakt nede)		

Med øvelserne overstået, kan formlen for spændingsdelere gennemgås, hvorefter yderligere opstillinger kan beregnes først og derefter efterprøves i praksis.

$$U1 = U * \frac{R1}{R1 + R2}$$

Forklar eleverne, at selvom mange sensorer er baseret på spændingsdelere, så er de ovenstående der lige har været konstrueret, ikke sensorer. Dette fordi de er statiske af natur og altså ikke registrerer ændringer i det miljø de befinder sig i. Hvis der i stedet for statiske modstande, havde været anvendt dynamiske der ændrer deres interne modstand i takt med ændringer i de omgivelser de befinder sig, havde dette opfyldt kravene til en sensor. Introducer derfor LDR'en (Light Dependent Resistor) og forklar at dennes interne modstand netop er dynamisk varierende alt efter hvor meget lys der falder på den.

Indsæt LDR'en i spændingsdeleren og aflæs herefter og print værdien af spændingsdeleren til monitoren. Fordi at spændingsfaldet over den enkelte modstand i spændingsdeleren ikke længere er konstant, repræsenterer den aflæste værdi nu de ændringer i miljøet hvor sensoren befinder sig (lys-niveauet).

Opgaver til eleverne:

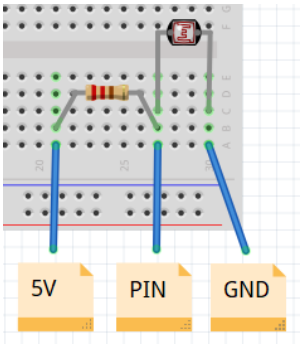
- Konstruer et system lignende en gadelampe, så der tændes for en LED når det er mørkt og slukkes igen når det er lyst.
- Konstruer et system inspireret af smarthomes, hvor der gennem anvendelsen af betingelser, gradvist skruer op/ned for lysstyrken på LED'en, i takt med at lyset ændrer sig, så man opnår et konstant lys-niveau inde i stuen.

LDR (Light Dependent Resistor / Photoresistor): Resistor size: 1KΩ

```
int LDRPin = A5;                    //the used pin
int LDRValue;                       //to store the
value

void setup() {
  pinMode(LDRPin, INPUT);           //set pin as INPUT
  Serial.begin(9600);               //enable serial
}

void loop() {
  //read and store the value, then prints it
  LDRValue = analogRead(LDRPin);
  Serial.println(LDRValue);
}
```



*PIN: A2-A5

Fig 9: Eksempel på hvordan en LDR kan anvendes.

1.3.6 5. Fase – Udvidelse med yderligere sensorer (Potentiometer, IR-Sensor) (60 min):

Introducer eleverne til henholdsvis potentiometeret og forklar at ligesom lyssensoren (LDR), bygger denne også på en spændingsdelerer, hvorfor komponenterne frit kan skiftes rundt, uden at der behøves laves ændringer i koden. Forklar at potentiometeret har en glidende overgang i forholdet mellem de to modstandes værdier, baseret på hvor glideren interagerer med den cirkulære modstand.

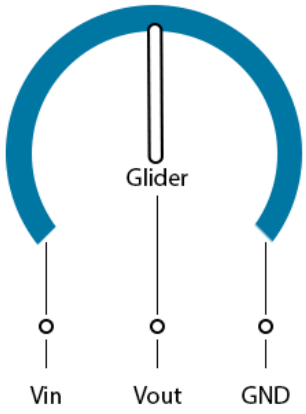


Fig. 10: Illustration af hvordan potentiometeret er opbygget og fungerer.

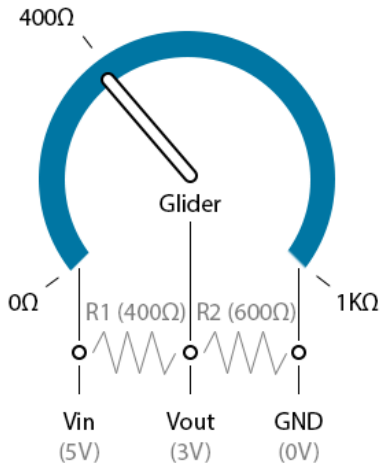


Fig. 11: Illustration af et 1K potentiometer i brug.

Introducer IR-sensoren og gennemgå hvordan den er opbygget af en IR-LED, samt en IR Fotodiode. IR-sensoren fungerer ved at IR-LED'en udsender et infrarødt lys (dette er usynligt for det menneskelige øje, men kan ofte ses med en kameraet i en mobiltelefon) der herefter i varierende grad reflekteres tilbage til IR Fotodioden afhængig af overfladen. Det er mængden af reflekteret lys der aflæses af sensoren. Eleverne vil derfor også finde, at hvis de holder den væk fra overfalden, så læser den "sort" (eller rettere, den registrerer en mangel på refleksion).

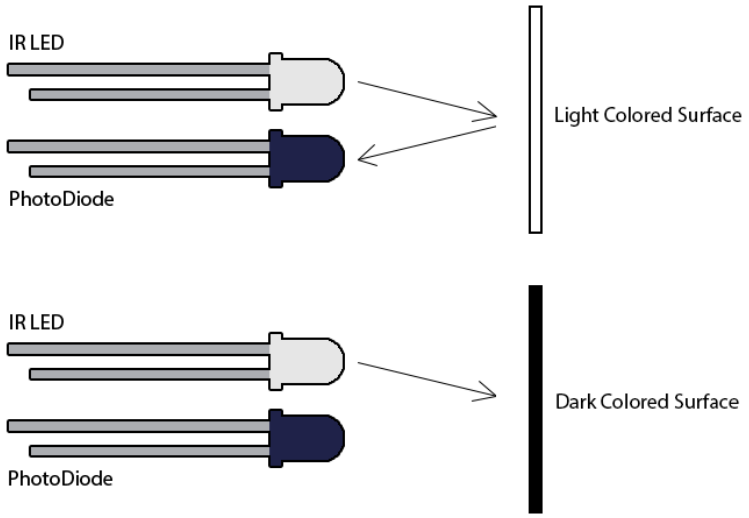


Fig. 12: Illustration af hvordan IR-sensoren fungerer.

Opgaver til eleverne (potentiometer):

- Konstruer et system til lysdæmpning, der gradvist tænder for LED'en, med en fem-trins overgang, når potentiometeret drejes fra venstre mod højre.
- Konstruer et system til lysdæmpning, der gradvist tænder for LED'en, med en flydende overgang, når potentiometeret drejes fra venstre mod højre.

Opgaver til eleverne (IR-sensor):

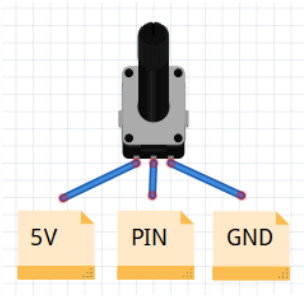
- Eksperimenter med, på hvilken afstand IR-sensoren er bedst til at opfange forskellen mellem hvidt og sort papir.
- Genbrug koden fra potentiometeret og konstruer et system, hvor en LED tænder, hvis IR-sensoren læser en lav værdi (massivt genskin fra hvidt papir), mens den slukker hvis der læses en høj værdi (minimalt genskin fra sort papir).
- Genbrug koden fra potentiometeret og konstruer et system, hvor en LED tænder mere og mere, i fem intervaller, baseret på mængden af genskin på IR-sensoren.

Potentiometer:

```
int PotPin = A5;                    //the used pin
int PotValue;                       //to store the
value

void setup() {
  pinMode(PotPin, INPUT);           //set pin as INPUT
  Serial.begin(9600);               //enable serial
}

void loop() {
  //read and store the value, then prints it
  PotValue = analogRead(PotPin);
  Serial.println(PotValue);
}
```



*PIN: A2-A5

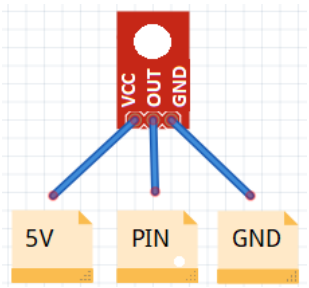
Fig 13: Eksempel på hvordan Potentiometeret kan aflæses.

IR-Sensor: QRE1113 (Analog)

```
int IRPin = A5;                 //the used pin
int IRValue;                    //to store the
value

void setup() {
  pinMode(IRPin, INPUT);        //set pin as INPUT
  Serial.begin(9600);           //enable serial
}

void loop() {
  //read and store the value, then prints it
  IRValue = analogRead(IRPin);
  Serial.println(IRValue);
}
```



*PIN: A2-A5

Fig 14: Eksempel på hvordan IR-Sensoren kan aflæses.

1.3.7 6. Fase – Aktuatorer (Servo- og DC-Motor) (60 min):

Introducer eleverne til servomotoren og derigennem også til biblioteker og forklar dem at dette er en samling af forprogrammeret kode, hvilket i dette tilfælde skal anvendes til at kontrollere servo-motoren. Lad eleverne indstille servomotoren til en given position og lad dem herefter prøve indstille den til at bevæge sig fra side til side. Her er det vigtigt at komme ind på anvendelsen af delays, foruden hvilke servo motoren ikke har tid til at gå til den angivne position, førend den får signalet om den næste position og vice versa.

Opgaver til eleverne:

- Konstruer et system der kan anvendes i et eksoskelet, med det formål at forstærke det menneskelige albueled, så lagerarbejdere ikke slider kroppen gennem tunge løft, ved at lade servomotorens position, følge potentiometerets.
- Konstruer et system til automatisk åbning af vinduer, ved at lave servo motoren veksle mellem at stå i en position af 0 og 90°, når der trykkes på en kontakt.

Introducer eleverne til motorshieldet, samt DC-motoren. I samme ombæring kan motorshieldets indbyggede H-broer (den har to, én til hver motor) forklares ganske kort. På Figur 15 kan der ses en forsimplet udgave af kredsløbet for en H-bro, det vigtige at bide mærke i, er de fire transistorer P1, P2 og N1, N2, samt de to forbindelser dertil F1 og F2. De er sat op således at når der sættes spænding på F1, så åbnes P1, mens N1 lukkes og vice versa når der ingen spænding er på F1. På samme måde afhænger P2 og N2 af spændingen på F2. Ved derfor at sætte 5V til F1 og 0V til F2, vil strømmen løbe den ene vej gennem motoren, mens den vil løbe den anden vej hvis der blev sat 0V til F1 og 5V til F2. Satte man derimod 5V eller 0V til både F1 og F2, vil strømmen ikke kunne løbe nogen af vejene. Forklar at grunden til at motorshieldet og dets H-broer anvendes til at styre DC-motorerne, er at disse kan lede en stærkere strøm gennem motorerne end Arduinoen selv kan forsyne dem med. Den strøm motorshieldet leder gennem DC motorerne, kommer direkte fra ”Vin” (voltage in – dvs. den tilsluttede batteripakke (9V)).

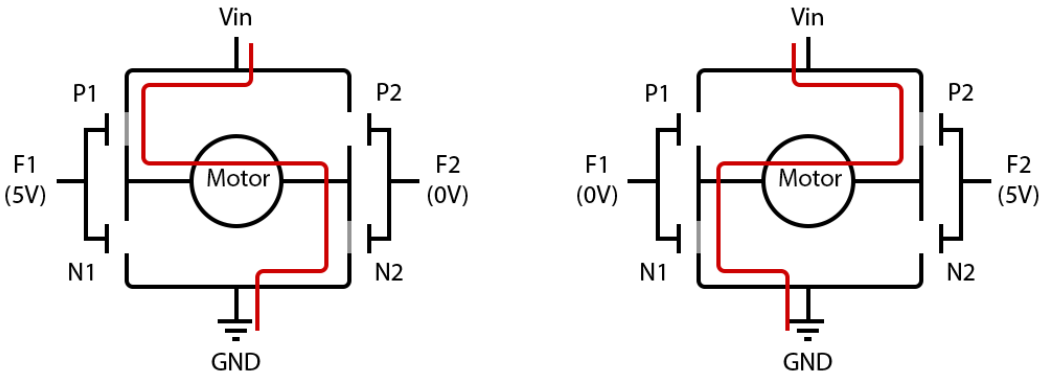


Fig. 15: Forsimplet illustration af kredsløbet i en H-bro.

Opgaver til eleverne:

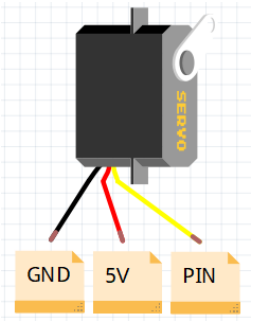
- Eksperimenter med at få DC-motoren til at køre skiftevis med og mod uret i intervaller af 5 sekunder, samt forskellige hastigheder.
- Konstruer et system beregnet til ventilation, hvor DC-motoren gradvist ændrer sin hastighed, i takt med at et potentiometer drejes fra venstre mod højre og vice versa.

Servo-Motor:

```
#include <Servo.h>          //include library
Servo servoMotor;           //create new servo
object
int servoPin = 2;           //the used pin

void setup() {
  //attach the used pin to the servo object
  servoMotor.attach(servoPin);
}

void loop() {
  //turn the servo to a position (value: 0-179)
  servoMotor.write(90);
}
```



*PIN: 2, 4-7, 10, A2-A5

Fig. 16: Eksempel på hvordan servomotoren kan sættes i en given position.

DC-Motor:

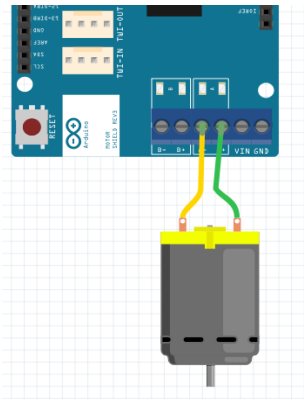
```
int motorASpeed = 3;                //PWM A
int motorADirection = 12;           //DIR A

void setup() {
  //set all motor pins to PUTPUT
  pinMode(motorASpeed, OUTPUT);
  pinMode(motorADirection, OUTPUT);
}

void loop() {
```

```
  //DIRECTION: Turn clockwise = HIGH,
  //turn counter-clockwise = LOW
  digitalWrite(motorADirection, HIGH);

  //SPEED: no speed = 0, full speed = 255
  analogWrite(motorASpeed, 150);
}
```



*for the pins used for motor B, see the motor-shield

Fig. 17: Eksempel på hvordan DC-motoren kan styres.

1.3.8 7. Fase – Andre typer af sensorer (Afstandssensor, Farvesensor) (60 min):

Introducer afstandssensoren til eleverne og gennemgå hvordan den kan anvendes til måle afstand fra sensoren til det nærmeste objekt. Gennemgå hvordan den er opbygget over en højtaler og en mikrofon, hvor højtaleren sender et ping ud (ping'et kan ikke høres af mennesker, fordi frekvensen er for høj), der når dette rammer en overflade, reflekteres tilbage til mikrofonen. Tiden det tager mellem at ping'et sendes ud og kommer tilbage, kan herefter bruges til at udregne distancen mellem sensoren og det ramte objekt. Til det formål anvendes metoden pulseIn(), hvilken registrerer tiden det tager fra at en pin enten går fra HIGH til LOW, eller LOW til HIGH, tiden måles i mikrosekunder (0.000001s). Følgende formel kan derfor anvendes, til at udregne afstanden i centimer.

$$Pulse\ In\ (måles\ i\ mikrosekunder, 0.000001s)$$

$$Lydens\ hastighed = \frac{343m}{s} = \frac{34300cm}{s} = \frac{0.034cm}{ms}$$

$$Afstand\ til\ objektet\ i\ cm: Pulse\ In\ (ms) * \frac{0.034cm}{1ms} / 2\ (frem\ og\ tilbage)$$

Opgaver til eleverne:

- Eksperimenter med at måle forskellige afstande inde i klasseværelse og udskrive disse til skærmen. Find den mindste og længste afstand som sensoren virker på.
- Konstruer et alarmsystem til en industrirobot, der advarer medarbejderne om at de er for tæt på maskinerne, ved at lade en LED lyse op, når afstandssensoren registrerer at der er en person indenfor 2 meters afstand.

Introducer farvesensoren og forklar kort hvordan denne virker. Den er opbygget således at den gennem et filter, måler den reflekterede værdi af henholdsvis Rød, Grøn og Blå (RGB) lys, én farve ad gangen. Ud fra RGB-værdierne, kan man skabe alle farver, men det er vigtigt at vide, at de alle tre tilsammen giver hvid, mens fraværet af alle tre resulterer i sort. Ideelt set, vil værdierne opføre sig på Figur 18, i virkeligheden er dette dog ikke tilfældet. Når sensoren anvendes, registrerer den først en farve og

nedskriver hvilke RGB-værdier sensoren returnerer, hvorefter disse værdier kan anvendes til at definere en given farve (i det udviklede bibliotek, er det kun farverne rød, grøn, blå og gul, der kan defineres – se figur 20). Desuden skal man grundet at RGB-værdierne repræsenterer mængden af reflekteret lys, være opmærksom på vekslende lysforhold. Er eks. farven gul blevet defineret i høj solskin, ud fra et sæt af læste RGB-værdierne, vil disse altså ikke nødvendigvis være de samme når det er blevet aften. For at komme rundt om dette, er det derfor ofte en god ide at afskærme det man læser, for derved i større grad at holde lysforholdene statiske.

Opgaver til eleverne:

- Eksperimentere med at måle og definere farverne Rød, Grøn, Blå, Gul, Hvid og Sort. Hvilke farvers RGB-værdier adskiller sig mest fra hinanden og er der nogle der ligger tættere end andre?
- Eksperimenter med hvilken forskel i værdi farverne har, alt efter hvilke lysforhold der er gældende.
- Konstruer et system, der anvender betingelser til at udskrive de registrerede farver til skærmen.

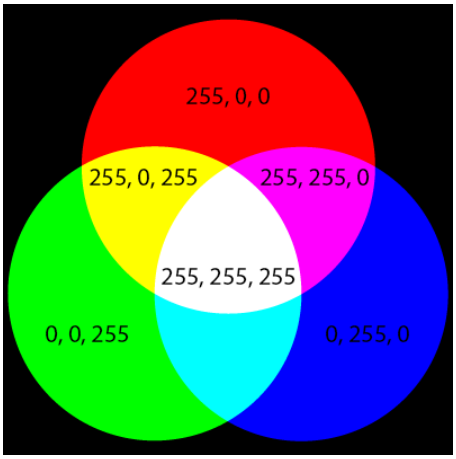


Fig. 18: Oversigt over de Ideelle RGB-værdier.

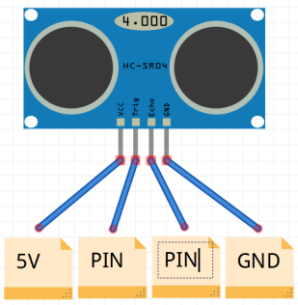
Ultra-Sound Sensor: HC-RS04

```
int trigPin = 2;              //the used trig pin
int echoPin = 4;              //the used echo pin
int distance;                 //to store the value

void setup() {
  pinMode(trigPin, OUTPUT); //sets pin as OUTPUT
  pinMode(echoPin, INPUT);  //sets pin as INPUT
  Serial.begin(9600);       //enables serial
}

void loop() {
  //store the returned value from the function
  distance = getDistance();

  //prints the stored value
```



*PIN: 2, 4-7, 10, A2-A5

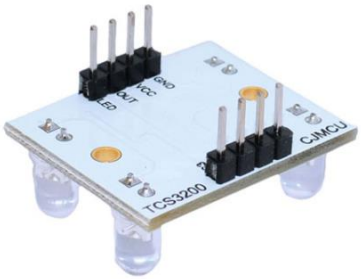
```
  Serial.println(distance);
}

//function - returns the distance
int getDistance() {
  //sends out a trigger sound
  digitalWrite(trigPin, LOW);
  delayMicroseconds(10);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  //returns the received echo in centimeter
  return pulseIn(echoPin, HIGH)* 0.034/2;
}
```

Fig. 19: Eksempel på hvordan afstandssensoren kan anvendes.

Color-Sensor: TCS3200




```
//includes the library
#include <Color.h>

//the used pins
int S0 = 2;
int S1 = 4;
int S2 = 5;
int S3 = 6;
int OUT = 7;

//creates a new color-sensor object
Color color(S0, S1, S2, S3, OUT);

//used to store the color-value
String colorValue = "";

void setup() {
  setColors();
  Serial.begin(9600);  //enables serial
}

void loop() {
  //prints the R(ed), G(reen), B(lue) and W(hite) values from the
  //sensor.
  //write down these values for each color you want to identify
  //(red, green, blue, yellow) and insert them into the
  //setColors function.
  Serial.println(color.getRGBValues());

  //gets the color and prints it
  colorValue = color.getColor();
  Serial.println(colorValue);
```

```
}

//Defines the Colors
void setColors() {
  //color.setRed(R, G, B, W);
  color.setRed(40, 104, 22, 81);
  color.setBlue(46, 36, 14, 56);
  color.setGreen(89, 56, 16, 33);
  color.setYellow(29, 36, 12, 55);
}
```

```
READ ME NOTE: In order to enable the inclusion of the Color library, copy
paste the entire "Color Library" folder, into the "Documents -> Arduino ->
Libraries" folder.
```

Fig. 20: Eksempel på hvordan farvesensoren kan anvendes.

1.3.9 8. Fase – Introduktion til yderligere programmeringsprincipper (Funktioner og While-Løkker) (45 min):

I forbindelse anvendelsen af afstandsasensoren og farvesensoren, blev der til begge anvendt en funktion. Introducer derfor eleverne for funktioner og forklar hvordan de fungerer ved at indkapsle et stykke kode, der herefter kan kaldes igen og igen. Dette gør det både lettere at anvende (ændringer i koden skal kun foretages ét sted, fremfor mange steder), samt lettere at læse (funktionen kan indkapsle mange linjers kode, under et enkelt og meningsgivende navn).

Opgaver til eleverne:

- Lav en funktion kaldet "printHello", der når den kaldes printer "Hello" til skærmen.
- Lav en funktion kaldet "sayHelloTo" hvilken tager en String som parameter, der når den kaldes printer "Hello ", samt parameteret.
- Lav en funktion kaldet doubleThisNumber, der tager en int som parameter, der når den kaldes gange værdien fra parameteret med to, hvorefter den nye værdi returneres. Den returnerede værdi skal gemmes i en variabel og printes.

Introducer while-løkker og forklar hvordan de gentager et stykke kode igen og igen, så længe den tilknyttede betingelse er sand.

Opgaver til eleverne:

- Lav en funktion indeholdende en while-løkke der kører 10 gange, hvori counteren printes.
- Lav en funktion med en while-løkke der printer "Happy Birthday to You" med intervaller af 1 sekund, trykkes en kontakt ned skal løkken afbrydes.

Function (return no value):

```
void loop() {
  //runs the code inside-
  //myFunction
  myFunction();
}
```

Function (return a value):

```
void loop() {
  //stores the returned value-
  //from myFunction
  int returnedValue;
  returnedValue = myFunction();
}
```

```
//datatype name() {}
void myFunction() {
  //some code to run
}
```

```
//datatype name() {}
int myFunction() {
  //some code to run

  //return a value
  int value = 100;
  return value;
}
```

Fig 21: Oversigt hvordan funktioner kan anvendes.

While loop:

```
void loop() {
  int x = 0;

  //while (argument is correct)
  while(x < 10) {
    x = x + 1;
  }
}
```

Break:

```
void loop() {
  //while (argument - is correct)
  while(true) {

    if(something) {
      //break out of the loop
      break;
    }

  }
}
```

Fig. 22: Oversigt over hvordan while-loops kan anvendes.

1.4 Temaprojekt – Automatisering (12 x 45 min.)

Kort om indhold: Projektet omhandler automatiseringen af en kaffebønnecentral, hvortil der skal udvikles to typer af robotter. En robot der sorterer bønnerne efter kvalitet (farvesortering), samt en robot der transporter dem rundt på lageret (linjefølgning). Eleverne vælger hvilken en af de to robotter de vil arbejde med, hvorefter det er tiltænkt at de arbejder relativt selvstændigt med projektet i mindre grupper (det anbefales at eleverne er to elever pr. gruppe).

Læringsudbytte: Eleverne lærer at anvende det lærte fra tidl. lektioner i en praktisk kontekst, til at løse konkrete problemstillinger inspireret fra virkelighedens industrielle arbejdspladser. Samtidigt giver dette dem en dybere indsigt i og forståelse for udviklingen af komplekse automatiserede systemer, samt disses begrænsninger.

Materialer: I det udviklede materiale er LEGO blevet anvendt som platform for de udviklede robotter. For at lette integreringen af elektronik i LEGO-platformen, er der derfor samtidigt blevet udviklet en række 3D modeller der er kompatible hermed. Manualer hertil, samt 3D modeller er ligesom tidl. beskrevet materiale, gjort tilgængelig på hjemmesiden. Det samme gør sig gældende for en komplet oversigt over de i projektet anvendte komponenter, løsningsforslag, samt forslag til videre udvikling af robotterne mm. Det er vigtigt at notere sig, at robotterne kan konstrueres på mange forskellige måder og af mange forskellige materialer, de udviklede LEGO-manualer er derfor kun et bud på en løsning hertil.

1.4.1 Projekt oplæg:

Introducer eleverne til følgende projekt, hvilket tager sit udgangspunkt i en kaffebønne central, hvor bønnerne først sorteres efter kvalitet, hvorefter de fordeles rundt på det tilknyttede lager. På centralen har man tidligere håndsorteret bønnerne, hvorefter de ved håndkraft er blevet fordelt på lageret. Dette indebærer hårdt fysisk arbejde, samtidigt med at det tager langtid. Centralens ønske er derfor, at disse to processer skal automatiseres, hvorfor der skal bygges og programmeres to typer af robotter, der hver kan varetage en af disse funktioner.

> **Kaffebønnesortering (**farvesortering**)**: Bønnerne sorteres efter kvalitet, hvilken afgøres af deres farve, i forbindelse med projektet udgøres bønnerne af 3D printede klodser (rød, grøn, blå og gul). Robotten skal derfor kunne modtage en bønne, registrere hvilken farvekategori denne tilhører, rotere fire tilknyttede kasser rundt, således at den korresponderende kasse står ud for transportbåndet, for derefter at aflevere bønnen heri og rotere kasserne tilbage til udgangspunktet igen.
>
> **Transportering (**linjefølgning**)**: De sorterede bønner pakkes i kasser, der i forbindelse med projektet udgøres af 3D printede kasser (rød, grøn, blå og gul), farven indikerer hvilket af fire korresponderende områder på lageret, de skal afleveres på. Lageret er blevet udstyret med sorte streger i gulvet, hvilke robotterne skal navigere efter. Derudover skal robotterne for at undgå ulykker, stoppe op hvis der er et objekt foran dem der spærrer vejen, dette kan være menneskelige medarbejdere eller andre robotter.

Centralen er allerede indrettet med et område hvorpå sorteringen foregår og derudover har medarbejder lige færdiggjort arbejdet, med at optegne de streger robotterne skal navigere efter, se Figur 23.

Når eleverne har valgt hvilken af de to robotter de vil arbejde med, kan de følge den tilhørende fremgangsmåde fra de efterfølgende to afsnit.

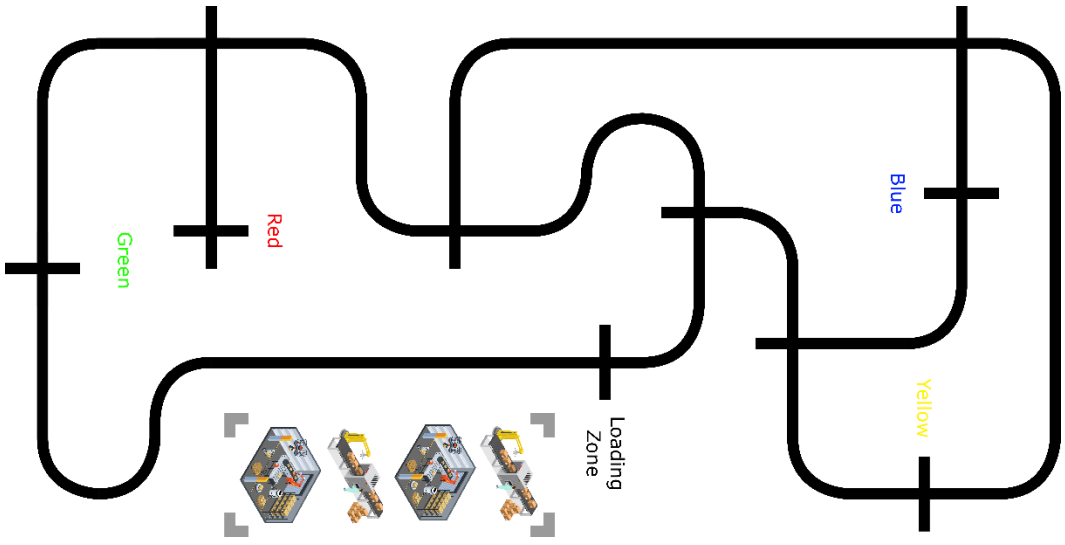


Fig. 23: Oversigt over lagerets lastezone og transporteringsruter.

I videoerne nedenfor, kan der ses eksempler fra virkelighedens verden, på robotter der er designet til automatisk sortering af kaffebønner, samt transportering.

Kaffebønnesortering: https://www.youtube.com/watch?v=i0nmIl-bNLI

Transportering: https://www.youtube.com/watch?v=WIlS3vNSuQ4

1.4.2 Kaffebønnesortering:

Robotten består af to hoveddele, hvilke kombineres til én. Den første del er transportbåndet, hvorpå kaffebønnerne først placeres, herefter fragtes de hen under en farvesensor hvorefter deres farve registreres (rød, grøn, blå eller gul), dernæst er det transportbåndets opgave at fragte dem det sidste stykke hen til og ned i en korresponderende kasse. Den anden del er den mekanisme der roterer de korresponderende kasser, således at når bønnerne når enden af transportbåndet, så falder de ned i den korrekte kasse. Se Figur 24 nedenfor.

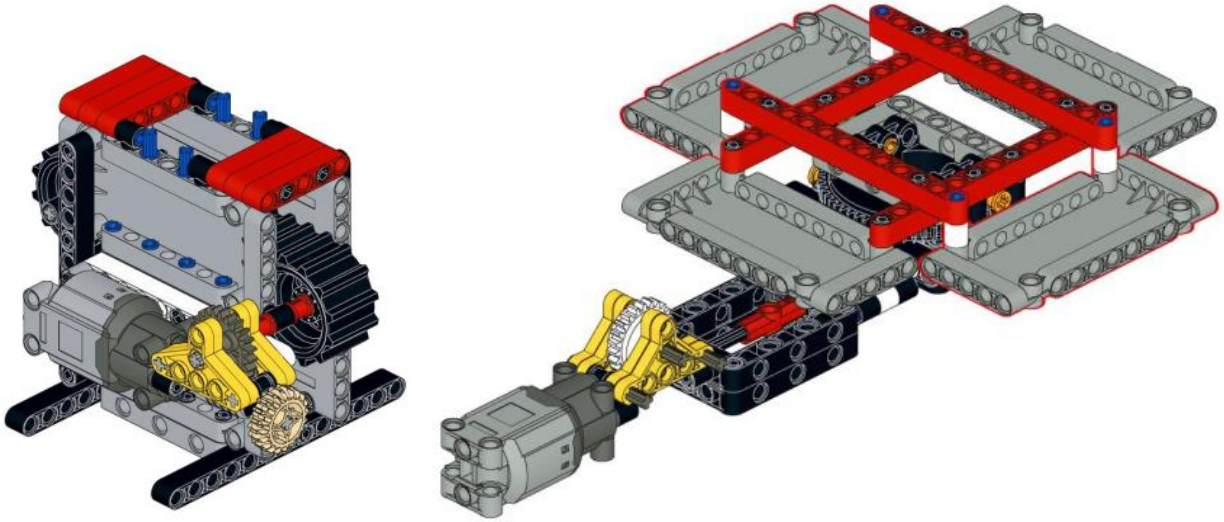


Fig. 24: LEGO-modellerne for transportbåndet og kasserne.

1.4.2.1 De Indledende Skridt:

1. Konstruer robotten:

Det første der skal have gøres, er at konstruere selve robotten, så der er noget at arbejde ud fra.

Opgave:

- Følg LEGO manualen ”Bean Sorter – Belt” og byg transportbåndet.
- Følg LEGO manualen ”Bean Sorter – Rotator” og byg rotationsmekanismen til kasserne.
- Følg LEGO manualen ”Bean Sorter – Combine” og byg de to dele sammen.
- Tildel nu kasserne hver deres individuelle farve, evt. med en tusch og vælg hvilken der som standard skal vende ind mod transportbåndet.

Materialer:

- 2stk. LEGO Power Functions Motor (Large).

2. Integrer Arduino platformen:

Nu skal Arduinoen tilføjes, hvilken er den mikrocontroller der skal stå for at kontrollere vores robot, samt det dertilhørende motorshield, hvilket er en udvidelse til Arduinoen som man anvender til at styre motorerne med, samt et breadboard.

Opgave:
- Følg manualen "Bean Sorter – Integrate the Arduino platform" og tilføj Arduinoen, motor shieldet, breadboardet, samt batteripakken.

Materialer:
- Arduino Uno.
- Motor shield. R3
- Breadboard (halv størrelse).
- Batteripakke + Ledning.
- Arduino modul (3D printet).
- Breadboard modul (3D printet).
- Batteripakke modul (3D printet).
- 2stk. han/han ledninger.

3. Test motorerne:
Det er nu tid til at få tilsluttet motorerne og samtidigt gennemføre en række mindre test, for at sikre os at de virker efter hensigten.

Opgave:
- Forbind motorerne til motor shieldet.
- Få transportbåndet til at køre fremad i 2 sekunder, med fuld hastighed.
- Få transportbåndet til at køre baglæns i 2 sekunder, med halv hastighed.
- Få kasserne til at rotere med uret i 2 sekunder, med fuld hastighed.
- Få kasserne til at rotere mod uret i 2 sekunder, med halv hastighed.

Cheat Sheets:
- Effectors – DC Motor.
- Code – pinMode.
- Code – digitalWrite.
- Code – analogWrite.
- Code – delay.

1.4.2.2 Transportbåndet:
Det anbefales at oprette en ny programfil til denne del, men gem gerne den tidl. fra De Indledende Skridt.

1. Anvend funktioner til at styre transportbåndet:
Funktioner er en god måde at indkapsle et stykke kode på, så det kan bruges mange gange, uden at man skal skrive koden hver eneste gang. Samtidigt gør det koden lettere at læse for mennesker.

Opgave:
- Lav en funktion "startBelt", der starter transportbåndet.

- Lav en funktion ”stopBelt”, der stopper transportbåndet.
- Lav en funktion ”deliverBean”, der afleverer bønnen (starter transportbåndet og lader det køre i 5 sekunder, hvorefter det stoppes igen).
- Test at alle tre funktioner virker.

Cheat Sheets:
- Code – Functions.

2. Integrer farvesensoren:
Førend transportbåndet kan anvendes, skal farvesensoren først integreres heri, samt at det skal testes om den virker efter hensigten.

Opgave:
- Følg manualen ”Bean Sorter – Integrate Color-Sensor” og integrer farvesensoren.
- Placer en ”bønne” på transport båndet under farvesensoren og print til skærmen, hvilken farve det er. Gør dette for alle farverne (rød, grøn, blå og gul).

Materialer:
- Farvesensor (TCS3200).
- Farvesensor modul (3D printet).
- 7stk. han/hun ledninger.

Cheat Sheets:
- Sensors – Color-Sensor (TCS3200).
- Code – Variables.
- Code – Serial Print.

3. Reager når farver læses:
Transportbåndet kan nu kontrolleres, samt anvende farvesensoren til at registrere farver, men indtil videre gør den det samme hele tiden. Det er derfor på tide at oprette en struktur for programmet, der anvender betingelser til at lade programmet foretage forskellige handlinger, alt efter hvilken farve farvesensoren registrerer (rød, grøn, blå eller gul).

Opgave:
- Start transportbåndet.
- Hvis farvesensoren læser ”red”:
  - Print ”red” til skærmen.
  - Stop transportbåndet og aflever ”bønnen”.
- Test at dette virker, når det gør, lav da en ”if else” for hver af de andre farver (grøn, blå og gul), der indeholder den samme funktionalitet som den for den røde farve.

Cheat Sheets:
- Code – Conditionals.

1.4.2.3 Kasserne:
Det anbefales at oprette en ny programfil til denne del, men gem den tidl. fra Transportbåndet.

1. Anvend funktioner til at rotere kasserne:
Anvend som ved transportbåndet, endnu engang funktioner til at indkapsle koden for motorstyring, sådan at den kan genbruges igen og igen.

Opgave:
- Lav en funktion ”startRotation”, der sætter kasserne i gang med at rotere mod uret.
- Lav en funktion ”stopRotation”, der stopper rotationen af kasserne.
- Lav en funktion ”rotatePosition” der tager en int som parameter (kald parameteret ”position”), på nær at printe parameteret til skærmen, efterlades denne funktion tom.
- Test at funktionerne virker.

Cheat Sheets:
- Code – Functions (with parameters).

2. Integrer switchen:
Indtil nu kasserne kun kunnet rotere frem og tilbage, baseret på tid, men fordi tiden det tager at rotere en omgang varierer alt efter hvor meget strøm der er tilbage på batterierne, hvor meget snavs der er i kommet ind i gearingen mm. er dette ikke nogen god løsning. Integrere derfor switch i konstruktionen, sådan at denne kan bruges til at registrere hver gang kasserne har roteret én plads.

Opgave:
- Følg manualen ”Bean Sorter – Integrate Switch” og integrer switchen.
- Test at switchen virker, ved at printe dens tilstand til skærmen.

Materialer:
- Switch.
- Switch modul (3D printet).
- 1KΩ modstand.
- 3stk. han/han ledninger.

Cheat Sheets:
- Sensors – Switch.

3. Kontroller hvor mange gange kasserne roterer:
Fordi at switchen automatisk trykkes ned hvergang en kasse kommer forbi den, kan denne bruge dette til at kontrollere hvor mange positionerne kasserne har flyttet sig, ved at tælle antallet af gange switchen trykkes ned. På den måde kan man altid sikre sig, at kasserne flytter sig præcist det antal positionerne der ønskes. Til det formål skal funktionen ”rotatePosition” færdiggøres, således at den kan kaldes med det antal af rotationer der ønskes.

Opgave:
- Inde i funktionen ”rotatePosition”:
    - Start rotationen af kasserne.
    - Lav en ny tæller variabel kaldet ”i” og giv den værdien 0.
    - Lav et while loop med betingelsen (i < position).

        - Hvis knappen er trykket ned:
            - Inkrementer ”i” med 1.
            - Vent ca. 200ms (afhængig af rotations hastigheden).
    - Stop rotationen af kasserne.
- Test at funktionen virker, ved at kalde den med forskellige parametre (1, 2, 3 osv).
- Tæl nu hvor mange gange hver af kasserne skal rotere en position, førend de står foran transportbåndet, samt hvor mange gange de skal rotere førend de er tilbage ved deres udgangspunkt igen. Skriv disse tal ned på et stykke papir.

Kombiner Transportbåndet og Kasserne:

Det anbefales at oprette en ny programfil til denne del, men gem gerne den tidl. fra De Indledende Skridt.

1. Kombiner det hele:

Det er nu på tide at kombinere transportbåndet med kasserne, så robotten er fuldt automatiseret. Heldigvis er næsten alt arbejdet der kræves hertil allerede lavet, gennem vores anvendelse af funktioner.

Opgave:

- Start transportbåndet.
- Hvis farvesensoren læser ”red”:
    - Print ”red” til skærmen.
    - Stop transportbåndet.
    - Roter kasserne sådan at den røde kasse står ud for transportbåndet.
    - Aflever ”bønnen”.
    - Roter kasserne sådan at den røde kasse igen står på dens udgangspunkt.
- Test at dette virker og udvid derefter programmet til også at omfatte de øvrige farver (grøn, blå og gul).

1.4.2.4 Ideer til Videreudvikling:

Kom gerne selv på nogle ideer til potentielle udvidelser, det kan eks. være at få forskellige LED’er til at lyse for at indikere hvilke handlingerne robotten pt. er ved at foretage sig. Det kan også være at bygge en holder der ved hjælp af en servo-motor, selv kan lægge nye bønner på transportbåndet, én ad gangen.

1.4.3 Transportering:

Robotten hviler på to forhjul, hvilke fungerer som dens drivkraft, samt et kuglehjul. Se Figur 25 nedenfor.

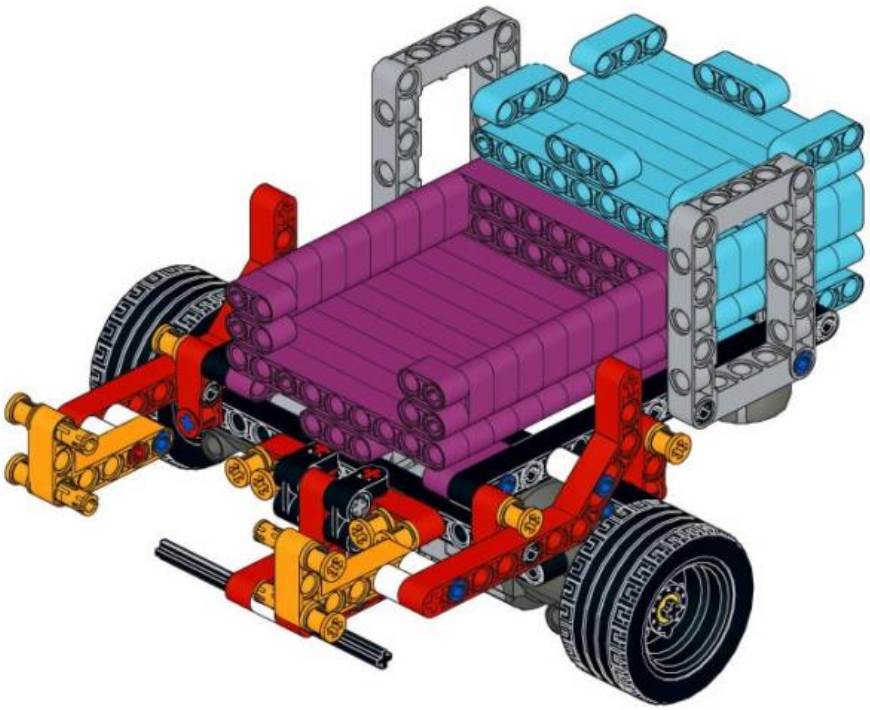


Fig. 25: LEGO-modellen til transportrobotten.

1.4.3.1 De Indledende Skridt:

1. Konstruer robotten:

Det første der skal gøres, er at konstruere selve robotten, så der er noget at arbejde ud fra.

Opgave:

- Følg LEGO manualen ”Transporter – Transporter” og byg transporteren.

Materialer:

- LEGO + 2stk. LEGO Power Functions Motor (L).

2. Integrer Arduino platformen:

Nu skal Arduinoen tilføjes, hvilket er den mikrocontroller der skal stå for at kontrollere vores robot, samt det dertilhørende motorshield, hvilket er en udvidelse til Arduinoen som man anvender til at styre motorerne med, samt et breadboard.

Opgave:

- Følg manualen ”Transporter – Integrate the Arduino platform” og tilføj Arduinoen, motor shieldet, breadboardet, samt batteripakken.

Materialer:

- Arduino Uno.
- Motor shield. R3

- Breadboard (halv størrelse).
- Batteripakke + Ledning.
- Arduino modul (3D printet).
- Breadboard modul (3D printet).
- Batteripakke modul (3D printet).
- 2stk. han/han ledninger.

3. Test motorerne:
Det er nu tid til at få tilsluttet motorerne og samtidigt gennemføre en række mindre test, for at sikre os at de virker efter hensigten.

Opgave:
- Forbind motorerne til motor shieldet.
- Få transporteren til at køre fremad i 2 sekunder, med fuld hastighed.
- Få transporteren til at køre baglæns i 2 sekunder, med halv hastighed.
- Få transporteren til at dreje til venstre i 2 sekunder, med fuld hastighed.
- Få transporteren til at dreje til højre i 2 sekunder, med halv hastighed.

Cheat Sheets:
- Effectors – DC Motor.
- Code – pinMode.
- Code – digitalWrite.
- Code – analogWrite.
- Code – delay.

1.4.3.2 Naviger efter Linjerne:
Det anbefales at oprette en ny programfil til denne del, men gem gerne den tidl. fra De Indledende Skridt.

1. Anvend funktioner til at styre transporteren:
Funktioner er en god måde at indkapsle et stykke kode på, så det kan bruges mange gange, uden at man skal skrive koden hver eneste gang. Samtidigt gør det koden lettere at læse for mennesker.

Opgave:
- Lav en funktion ”driveForward”, der kører robotten fremad.
- Lav en funktion ”crossLine”, der kører robotten ca. 5cm frem og stopper den igen.
- Lav en funktion ”turnLeft”, der drejer robotten en kvart omgang mod venstre.
- Lav en funktion ”turnRight”, der drejer robotten en kvart omgang mod højre.
- Lav en funktion ”uTurn”, der drejer robotten en halv omgang, retningen er ligegyldig.
- Lav en funktion ”stopRobot”, der stopper robotten i 1 minut.
- Test at alle seks funktioner virker.

Cheat Sheets:
- Code – Functions.

2. Integrer IR-sensorerne:

Førend transporteren kan følge linjerne i gulvet, skal dens to IR-sensorer først integreres, samt at det skal testes om de virker efter hensigten.

Opgave:
- Følg manualen "Transporter – Integrate IR-Sensors" og integrer IR-sensorerne.
- Læs først den venstre IR-sensors værdi og print denne til skærmen, noter dernæst værdierne for hvid og sort. Gentag herefter dette for den højre IR-sensor.

Materialer:
- IR-sensor (QRE1113, Analog).
- IR-sensor modul (3D printet).
- 6stk. han/han ledninger.

Cheat Sheets:
- Sensors – IR-sensor (QRE1113, Analog).
- Code – Variables.
- Code – analogRead.
- Code – Serial Print.

3. Følg linjen:
Det er nu på tide at anvende IR-sensorerne, til at kontrollere robotten så den kan følge en linje. Hertil skal der for hver IR-sensor, udregnes den værdi der ligger lige mellem værdien for hvid og værdien for sort. Denne værdi kaldes for tærsklen. For at følge en linje, skal motorerne køre hvis deres respektive IR-sensor ser hvidt, ellers skal de stoppe. På den måde vil robotten automatisk rette op, hvis den er ved at køre skævt og derved krydse ind over linjen, på samme måde vil den også automatisk stoppe når den kommer til et kryds, fordi begge sensorer nu ser sort. Begynd gerne med at simulere dette, ved at styre robotten med hånden.

Opgave:
- Få venstre motor til at køre hvis venstre IR-sensor læser mindre end tærsklen, ellers skal den stoppe.
- Få højre motor til at køre hvis højre IR-sensor læser mindre end tærsklen, ellers skal den stoppe.
- Test at robotten nu kan følge linjen, samt at den stopper når den når til et kryds.

Cheat Sheets:
- Code – Conditionals.

4. Følg linjen, som en funktion:
Indtil videre stopper robottenautomatisk når den når et kryds, men den starter ligeledes også af sig selv igen, hvis man manuelt skubber den forbi krydset, eller sætter den over på en ny linje. Hvis man derfor gerne vil bruge dreje kommandoerne gemt i funktionerne "turnLeft" og "turnRight", til at navigere i et kryds, skaber dette et problem. Prøv gerne at kalde disse fra koden – nu står robotten blot og drejer rundt og følger slet ikke linjen mere. Dette skyldes at de linjer kode der får den til at følge linjen, er udført på et splitsekund, hvorefter dreje kommandoen kaldes igen. Der skal derfor laves en selvstændig funktion der får robotten til at følge linjen, indtil den når et kryds, hvorefter funktionen afsluttes.

Opgave:

- Lav en funktion ”followLine”.
    - Anvend et while loop med betingelsen (true).
        - Få venstre motor til at køre hvis venstre IR-sensor læser mindre end tærsklen, ellers skal den stoppe.
        - Få højre motor til at køre hvis højre IR-sensor læser mindre end tærsklen, ellers skal den stoppe.
        - Hvis begge IR-sensorer læser mindre end tærsklen, stop begge motorer og afbryd while loopet.
- Test om funktionen virker.
- Test nu om den virker sammen med dreje funktionerne, samt funktionen for at køre lige ud i et kryds, ved eksempelvis at kalde funktionerne nedenfor og derefter observere om robotten kører den forventede rute. Husk at når der er gået et minut er funktionen ”stopRobot” færdig, også starter sekvensen forfra.
    - followLine.
    - turnLeft.
    - followLine.
    - crossLine.
    - followLine.
    - turnRight.
    - followLine.
    - uTurn.
    - stopRobot.

Cheat Sheets:

- Code – While loop (with break).

1.4.3.3 Naviger efter Omgivelserne:

Det anbefales at oprette gemme en kopi af programmet, for derefter at fortsætte arbejdet i kopien.

1. Integrer afstandssensoren:

Robotten er nu i stand til at navigere efter linjerne på lagerets gulv, men hvad hvis der er en anden robot foran den, eller at en medarbejder har overset at den kommer lige imod dem. For at undgå de problemer og skader der kan opstå herved, kan der integreres en afstandssensor, der lader robotten måle afstanden hen til det nærmeste objekt. Derved kan den bagefter programmeres til at stoppe, skulle et objekt være for tæt på den.

Opgave:

- Følg manualen ”Transporter – Ultrasound-Sensor” og integrer afstandssensoren.
- Print afstanden til det nærmeste objekt, på skærmen.

Materialer:

- Afstandssensor (HC-RS04).
- Afstandssensor modul (3D printet).
- 4stk. han/han ledninger.

Cheat Sheets:

- Sensors – Ultrasonic-Sensor (HC-RS04).

2. Stop for objekter der blokerer for robotten:
Opgrader funktion ”followLine”, sådan at robotten stopper hvis afstanden til et objekt foran, er under 20cm.

Opgave:

- Opgrader funktionen ”followLine”, sådan at:
  - Det første der sker i while loopet, er at læse afstanden til det nærmeste objekt.
  - Opgrader betingelserne for at motorerne må køre, sådan at betingelsen nu lyder:
    - Hvis venstre IR-sensor er under tærsklen og afstanden er mere end 20cm, kør venstre motor, ellers stop den.
    - Hvis højre IR-sensor er under tærsklen og afstanden er mere end 20cm, kør højre motor, ellers stop den.

1.4.3.4 Aflever Kassen med Bønner:

Det anbefales at oprette gemme en kopi af programmet, for derefter at fortsætte arbejdet i kopien.

1. Integrer servo-motoren:
Medarbejderne kan nu sætte en kasse på robotten og programmere den til at følge en rute, men den kan endnu ikke selv aflevere kassen når den er nået frem. Integrer derfor en servo-motor, der kan anvendes til at tippe kassen af ladet, hvorefter robotten kan køre tilbage efter en ny kasse.

Opgave:

- Følg manualen ”Transporter – Servo-Motor” og integrer servo-motoren.
- Test at servo-motoren virker efter hensigten, ved at sætte den i forskellige positioner.

Materialer:

- Servo motor.
- 6stk. han/han ledninger.

Cheat Sheets:

- Effectors – Servo-Motor.

2. Aflever kassen:
Anvendt en funktion til at lade robotten aflevere en kasse, når den når frem til det korrekte område.

Opgave:

- Lav en funktion ”deliverBox”, der ved hjælp af servo-motoren tipper ladet så kassen falder af, hvorefter servo-motoren kører tilbage til sit udgangspunkt.
- Test at funktionen virker, samt at robotten nu kan køre en rute, hvor den ved det korrekte sted, nu kan aflevere kassen den fragter.

1.4.3.5 Ideer til Videreudvikling:

Kom gerne selv på nogle ideer til potentielle udvidelser, det kan eks. være at få forskellige LED'er til at lyse for at indikere hvilke handlingerne robotten pt. er ved at foretage sig. Det kan også være at anvende en LDR (Light Dependent Resistor) til at tjekke for om robotten har en kasse på ladet.