Høgskulen på Vestlandet

Bacheloroppgave

ELE350

Predefinert informasjon

Startdato:	08-05-2023 09:00 CEST	**Termin:**	2023 VÅR
Sluttdato:	22-05-2023 14:00 CEST	**Vurderingsform:**	Norsk 6-trinns skala (A-F)
Eksamensform:	Bacheloroppgave		
Flowkode:	203 ELE350 1 O 2023 VÅR		
Intern sensor:	Olav Gerhard Haukenes Nygaard		

Deltaker

Navn:	Alexandra Viktoria Monsen
Kandidatnr.:	268
HVL-id:	591459@hvl.no

Informasjon fra deltaker

Egenerklæring *:	Ja
Inneholder besvarelsen konfidensielt materiale?:	Nei
Jeg bekrefter at jeg har registrert oppgavetittelen på norsk og engelsk i StudentWeb og vet at denne vil stå på vitnemålet mitt *:	Ja

Gruppe

Gruppenavn:	BO23EB-32
Gruppenummer:	37
Andre medlemmer i gruppen:	Bård Løvik

Jeg godkjenner avtalen om publisering av bacheloroppgaven min *

Ja

Er bacheloroppgaven skrevet som del av et større forskningsprosjekt ved HVL? *

Nei

Høgskulen på Vestlandet

BACHELOROPPGAVE:

BO23EB-32 Ciros og robot i et digitalt simuleringssystem

Bård Løvik
Alexandra Viktoria Monsen

22. mai. 2023

Dokumentkontroll

Rapportens tittel: BO23EB-32 Ciros og robot i et digital simuleringssystem	*Dato/Versjon* 16. mai. 2023/0.3
	Rapportnummer: BO23EB-32
Forfatter(e): Bård Løvik Alexandra Viktoria Monsen	*Studieretning:* AUT
	Antall sider m/vedlegg 69
Høgskolens veileder: Olav Gerhard Nygaard	*Gradering:* Åpen
Eventuelle Merknader: Vi tillater at oppgaven kan publiseres.	

Oppdragsgiver: Fagskolen på Vestlandet	*Oppdragsgivers referanse:* Guttorm Lyngvær
Oppdragsgivers kontaktperson(er) (inklusiv kontaktinformasjon): *Bjørnar Seim Telefon: 917 78 989 E-post: bjornar.seim@vlfk.no* *Guttorm Lyngvær Telefon: 406 11 630 E-post: guttorm.lyngver@vlfk.no*	

Revisjon	Dato	Status	Utført av
R0.0	17.04.23	Første utkast	AVM/BL
R0.1	24.04.23	Legger inn teoridel	AVM/BL
R0.1	01.05.23	Test med student	AVM/BL
R0.2	02.05.23	Andre utkast	AVM/BL
R0.3	16.05.23	Endring av oppsett og struktur	AVM/BL
R1.0	22.05.23	Siste utkast	AVM/BL

Forord

Denne rapporten er skrevet som en del av vår bacheloroppgave ved studiet Automatisering med robotikk på Høgskulen på Vestlandet, campus Bergen.

Å jobbe med dette prosjektet har vært en lærerik prosess. Det har oppstått flere utfordringer underveis i prosjektet, men dette har gitt oss muligheten til å anvende tidligere kunnskap fra studie. Samtidig har utfordringene ført til at vi har tilegnet oss mye ny kunnskap som vi vil ta med oss videre i arbeidslivet.

Vi vil gjerne takke våre veiledere Guttorm Lyngvær og Bjørnar Seim fra Fagskolen på Vestlandet. De tok oss godt imot da vi ankom Fagskolen og har vært behjelpelige med tildeling av studieplasser på skolen, samt gitt oss tilgang til læringsplattformer og veiledning ved behov. Vi ønsker også å takke Olav Gerhard Nygaard for god veiledning, positive og konstruktive tilbakemeldinger, samt gode innspill til oppgaven. Vi vil også rette en spesiell takk til studentene i ‘’emne-H robotikk’’ på Fagskolen for testing og tilbakemelding av laboratorieoppgaver.

Sammendrag

Fagskolen på Vestlandet har definert en bacheloroppgave hvor de ønsker økte ressurser i programmeringsverktøyet Ciros. På bakgrunn av dette har vi fått i oppgave å utvikle laboratorieoppgaver for studenter på Fagskolen.

Vi har utformet fire modeller i Ciros med tilhørende oppgavesett, løsningsforslag og modelleringsprosess. Disse oppgavene har økende vanskelighetsgrad. Alle oppgavene benytter Ciros 6.4, robot-type Mitsubishi RV-4FL og programmeringsspråk Mitsubishi Electric Factory Automation Basic IV.

Kravspesifikasjonene for prosjektet er i hovedsak oppnådd. Det har vært utfordrende å oppfylle læringsmålene til Fagskolen fordi nivået på laboratorieoppgavene ikke kunne være for avansert. Dette skyldes at studentene er yrkesfaglærte og har derfor varierende teoretisk bakgrunn. Med tanke på dokumentasjon har vi oppnådd kravspesifikasjonen i stor grad.

Summary

The Tertiary Vocational College in Western Norway has defined a bachelor thesis where they request improved teaching tools for the programming tool Ciros. Due to limited teaching resources, our focus has been to develop laboratory assignments for the students at the college.

We have designed four laboratory assignments with sets of tasks, solutions and modeling processes. The assignments are designed in different levels of complexity. All tasks use Ciros 6.4, Mitsubishi RV-4FL robot type, and Mitsubishi Electric Factory Automation Basic IV programming language.

The requirements for the project have been achieved in general. It has been challenging to meet the learning objectives of the vocational school as the level of laboratory tasks cannot be too advanced. This is because the students are skilled workers and do not have the same theoretical background. With regards to documentation, we have achieved the requirements.

Innholdsfortegnelse

1 Innledning

I dette kapittelet presenterer vi oppdragsgiver og problemstillingen som ble gitt. Det vil også bli gitt en kort presentasjon om hvordan vi vurderer å løse problemstillingen.

1.1 Oppdragsgiver

Oppdragsgiveren for prosjektet er Fagskolen på Vestland. Dette er en av de største fagskolene i landet, med ca. 1700 studenter og ca. 120 ansatte som er plassert rundt på 8 studiesteder i Vestland fylke[1]. Studentene på skolen er yrkesfaglærte som ønsker å utvikle sin kompetanse.

1.2 Problemstilling

Fagskolen ønsker å utvide bruken av Ciros som er et modelleringsprogram for å designe og programmere robotikk. De har behov for økte ressurser til å tilrettelegge programmeringsverktøyet for faglærerne og studentene innen fagfeltet robotikk. Med bakgrunn i dette har vi fått i oppdrag fra Fagskolen å lage eller modifisere eksisterende modeller i Ciros og utvikle et oppgavesett med laboratorieoppgaver rettet mot studenter på fagskolenivå.

I tett dialog med Fagskolen og i henhold til kravspesifikasjonene har vi kommet frem til at læringsmålet for laboratorieoppgavene skal være å introdusere studentene til forskjellige begreper innen robotteknologi. Blant annet robotens frihetsgrader, referanserammer og invers kinematikk.

1.3 Idé for løsningsforslag

Ciros gir mulighet for å introdusere og utvide kompetansen innen programmering og visualisering av robotsystemer. Fagtemaet robot i et digitalt simuleringssytem er av stor interesse. Simulering opp mot fysiske system gir høy presisjon og god analyse av systemet.

I dialog med veileder på Fagskolen er det kommet frem at det er ønskelig med flere laboratorieoppgaver innen robotikk. Fagskolen ønsker å utvide omfanget av laboratorieoppgaver innen robotikk for å fremme læring hos studentene. Det er tenkt at vi skal designe laboratorieoppgaver som studentene på Fagskolen skal løse. Laboratorieoppgavene skal videre deles inn i mindre moduler med varierende vanskelighetsgrad. Dette er for å tilpasse oppgavene til studentenes basiskompetanse og interesse. Laboratorieoppgavene skal også kunne brukes som introduksjon til robotikk og

Ciros for fremtidig undervisningspersonell ved Fagskolen. En stor del av oppgaven er å lage bruksanvisning for modellering av laboratorieoppgavene i Ciros samt løsningsforslag til oppgavesettet.

2 Kravspesifikasjon

Gjennom flere samtaler med Fagskolen ble det utviklet og gitt krav til hvordan problemstillingen skal besvares. Kravene som ble formulert blir videre presentert punktvis, etterfulgt av en utfyllende forklaring.

- **Anvende Ciros**
  - Bruk av Mitsubishi robotarm
  - Programspråk Mitsubishi Electric Factory Automation (Melfa) Basic IV
- **Et produkt som studentene kan ta nytte av:**
  - Det skal være en fungerende modell
- **Læringsmål fra Fagskolen som studentene skal oppnå gjennom laboppgavene:**
  - Økt kunnskap i Ciros 6.4
  - Kunne forstå og lage posisjon lister
  - Forstå og anvende frihetsgrader
  - Forstå og anvende referanserammer
  - Forstå konseptet fremover og invers kinematikk
- **Veiledende dokumentasjon skal:**
  - Være oversiktlig med inndeling til oppgaver
  - Ha tydelige bilder
  - Være på norsk
  - Være mulig å gjennomføre uten tidligere erfaring i Ciros

Laboratorieoppgavene skal bli designet i Ciros. Vi skal lage hver modell i separate filer, hvor hver modell inneholder en Mitsubishi robotarm. De ulike oppgavene skal blant annet ha geometriske figurer tilpasset sitt formål. Ettersom Mitsubishi roboter programmeres i Melfa Basic IV, skal dette bli programmeringsspråket. Hovedmålet for prosjektet er at det skal lages fungerende modeller i Ciros med tilhørende laboratorieoppgaver som studentene kan ta læring av.

Laboratorieoppgavene skal videre dekke de gitte læringsmålene til studentene. Det må derfor utformes oppgaver som er knyttet opp mot alle læringsmålene. Dette innebærer blant annet at studentene skal kunne forstå og anvende frihetsgrader og referanserammer.

Veiledende dokumentasjon skal være oversiktlig og ha tydelige bilder. Målet er at laboratorieoppgavene skal være mulig for studenter å gjennomføre uten tidligere erfaring i Ciros. Dokumentasjon knyttet til Ciros er per dags dato i hovedsak på spansk og tysk. Å lage oversiktlig dokumentasjon på norsk vil derfor være til nytte både for studenter og fagpersonell ved Fagskolen. I tillegg er det blitt vurdert at det kan være nyttig å lage et løsningsforslag i form av en video.

3 Analyse av problemet

I dette kapittelet skal vi se på generell robot teori og knytte dette opp mot studentenes fagnivå. Det blir også presentert valgt løsning i henhold til teoretisk vurdering og kravspesifikasjonene.

3.1 Robot teori

Innenfor temaet robot teori skal vi se nærmere på læringsmålene for studentene på Fagskolen. Dette innebærer generell kunnskap om referanserammer, kinematikk og frihetsgrader. Vi skal også se på bruken av digital tvilling og robot programmeringsmetodikk.

En sentral del av valg av løsning har vært å definere både teoretisk og praktisk fagnivå på laboratorieoppgavene. Vi har valgt å beskrive nødvendig teoretisk grunnlag for studentene og laboratorieoppgavene separat.

3.1.1 Referanserammer

I robotikk refererer en referanseramme til koordinatsystemet[2] som brukes til å beskrive roboten og dens miljø. En referanseramme er en definert posisjon og orientering i rommet som brukes til å beskrive bevegelsen og posisjon til roboten i forhold til omgivelsene. Det er ofte nødvendig å bytte fra ett koordinatsystem til et annet for å kontrollere bevegelsen til roboten. Dette kan bety konvertering fra det interne koordinatsystemet til det globale.

I tillegg til å bli brukt til å beskrive robotens bevegelse, kan referanserammer også brukes til å definere målene som roboten kan oppnå. Du kan for eksempel definere en referanseramme som representerer ønsket plassering og retning som roboten vil nå, og deretter kontrollere bevegelsen til roboten for å nå dette målet.

Referanserammer er derfor en viktig del av robotikk, og er avgjørende for nøyaktig og effektiv programmering og kontroll av roboter.

3.1.2 Invers kinematikk

En annen viktig del av robotikk er invers kinematikk. Invers kinematikk lar roboten beregne de nødvendige leddbevegelsene for å nå ønsket posisjon og retning av endeanordningen. For å forstå invers kinematikk er det viktig å først forstå fremover kinematikk. Fremover

kinematikk beskriver forholdet mellom leddvinklene og posisjonen til den endelige effektforsterkeren til roboten. Invers kinematikk er en invers operasjon hvor formålet er å finne de nødvendige leddvinklene som bringer endeeffektoren til ønsket posisjon og orientering. Dette kan være en utfordrende oppgave, spesielt for komplekse roboter med flere frihetsgrader. Det finnes ulike teknikker for å løse det inverse kinematikk problemet, inkludert analytiske metoder, numeriske metoder og hybrid metoder[3].

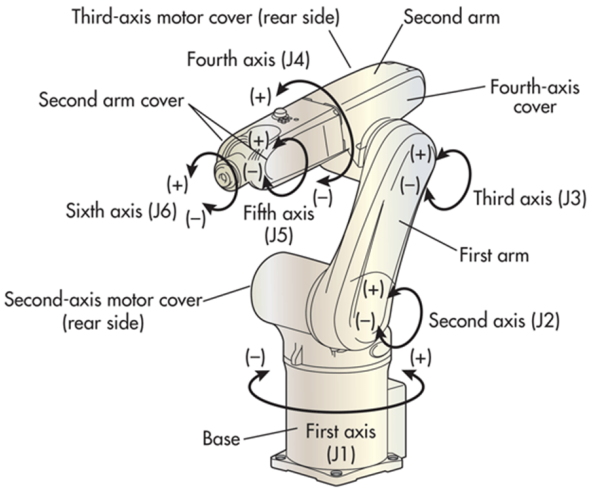
Analysemetoder er basert på en matematisk analyse av geometrien til roboten og kan gi eksakte løsninger i visse tilfeller. Numeriske metoder, som Newton Raphson - algoritmen bruker iterative beregningsmetoder for å finne løsninger. Hybride metoder kombinerer analytiske og numeriske teknikker for å oppnå raskere og enda mer nøyaktige resultater.

En annen faktor som spiller en viktig rolle i robotikk er optimalisering. Optimalisering brukes til å finne den mest effektive bevegelsen eller banen for roboten, for eksempel å minimere energiforbruket eller forkorte bevegelsestiden.

3.1.3 Frihetsgrader

Frihetsgrader viser til hvor mye uavhengig bevegelsesfrihet en robot har[4]. Dette konseptet er veldig viktig innen robotikk fordi det kan påvirke en robots evne til å bevege seg og utføre oppgaver. Generelt sett kan en robot med høyere frihetsgrad ha større bevegelsesfrihet og evne til å utføre mer komplekse oppgaver enn en robot med lavere frihetsgrad. For eksempel kan en industrirobot med seks frihetsgrader bevege seg mer fleksibelt og gjøre mer nøyaktige bevegelser enn en robot med bare tre frihetsgrader.

En enkel robot kan ha to frihetsgrader, for eksempel en mobil robot som kan bevege seg fremover og bakover og svinge til venstre eller høyre. Mens en avansert robot kan ha flere frihetsgrader, for eksempel en robotarm som kan bevege seg i flere retninger og justere gripeevnen i flere vinkler.



Figur 1: Frihetsgrader til 6 axet robot [5]

Frihetsgrader kan regnes ut ved følgende formel[6]:

$$DOF = \lambda(N - J - 1) + \sum_{i=1}^{J} Fi$$

N = Totalt linker

J = Antall joints som som er knyttet til N

Fi = Frihetsgrader i ledd "i"

λ = 3 dersom det er plan bevegelse. Dette vil si oppgitt i (x, y, ϕ). Dersom det er romlig bevegelse settes $\lambda = 6$. Da er bevegelsen oppgitt i $(x, y, z: [R])$.

Robotens frihetsgrader varierer avhengig av robotens design og formål. For eksempel kan en sveiserobot ha færre frihetsgrader enn en industriobot som utfører mer komplekse oppgaver. Riktig antall frihetsgrader er derfor viktig for å designe en robot som er egnet for det spesifikke formålet den skal brukes til.

3.1.4 Vurdering av laboratorieoppgavenes faglige nivå

I tett dialog med oppdragsgiver har vi vurdert nødvendig fagnivå når det gjelder robot bevegelser. For å sikre riktig fagnivå på laboratorieoppgavene er det nødvendig at oppgavene testes av studenter.

Når det gjelder referanserammer skal studentene både forstå konseptet og kunne anvende dette i Ciros. Referanserammer brukes ofte når man har flere arbeidsområder i Ciros. Studentene må derfor kunne klare å skille mellom ulike koordinatsystem. Dette skal de gjøre ved å benytte seg av illustrasjonstegninger gitt i oppgavene. Illustrasjonstegningene er gitt med et referansepunkt til de globale koordinatene og ved hjelp av dette skal de regne seg frem til ønsket posisjon.

Når det gjelder kinematikk skal studentene kunne forstå konseptet fremover kinematikk og invers kinemtaikk. De skal ikke utføre utregninger, men forstå hva som skjer når robotarmen benytter seg av de ulike formene for kinematikk. Studentene får tildelt noe teori på deres fagnivå som skal gjennomføres før de utfører første laboratorieoppgave. Videre i den første laboratorieoppgaven skal studentene anvende teach-in for å forståelse for hvordan kinematikken fungerer. Hovedfokuset i den første laboratorieoppgaven er på invers kinematikk ettersom det er dette som kan anses som mest utfordrende. Vi utelukker likevel ikke fremover kinematikk ettersom det er viktig at studentene også forstår det mest grunnleggende. De skal også svare på noen kontrollspørsmål oppgavesettet for å sikre at de har forstått konseptet kinematikk.

Når det gjelder frihetsgrader har vi valgt å fokusere på den fysiske roboten som har 6 frihetsgrader. Dette er for å unngå å gjøre oppgavesettet for komplisert. Vi tar også høyde for at den fysiske roboten vi har blitt kjent med og studentene anvender som digital tvilling har 6 frihetsgrader. Laboratorieoppgavene studentene får tildelt har i tillegg noe forenklet teori knyttet til frihetsgrader. Dette er teori med hovedfokus på forståelse av hvordan ulike ledd kan bevege seg og rotere. De skal også få forståelse for begrensninger innenfor robotens frihetsgrader.

3.1.5 Digital tvilling

Digital tvilling er en virtuell eller digital versjon av en fysisk enhet, produkt eller prosess[7]. Digital tvilling er et begrep som er av økende bruk i forskjellige industrier. Digital tvilling er som regel programmert online, men kan også kjøres i offline programmering.

En digital tvilling bruker data fra sensorer, maskinlæring og andre teknologier for å samle informasjon om den fysiske enheten i sanntid. Denne informasjonen brukes deretter til å

lage en nøyaktig virtuell modell av enheten som kan brukes til å teste forskjellige scenarioer og optimalisere ytelsen.

Tvilling-teknologi har potensial til å revolusjonere måten vi designer, produserer og vedlikeholder fysiske enheter og systemer. Dette kan også bidra til å redusere kostnader, øke produktiviteten og forbedre sikkerheten i industrien.

Teknologien brukes i forskjellige bransjer, for eksempel i produksjonsbransjen. Tvilling-teknologi gir produsenten mulighet til å simulere forskjellige produksjonslinjer og finne den mest effektive kostnadseffektive metoden for å produsere en enhet eller et produkt. Teknologien brukes også til å optimalisere energiproduksjon og energidistribusjon. En digital tvilling av en kraftstasjon kan brukes til å optimalisere driften og redusere energitap. Innenfor transportbransjen brukes tvilling-teknologi til å optimalisere drift og vedlikehold av transportinfrastruktur. En digital tvilling av en bro kan for eksempel brukes til å overvåke belastning og forutsi når broen trenger vedlikehold eller reparasjon. Dette er bare noen eksempler på hvordan teknologien kan benyttes i ulike bransjer. Digital tvilling teknologi kan ha stor innvirkning på industrien og samfunnet som helhet, og vi vil trolig se enda mer av denne teknologien i fremtiden.

Gjennom dialog med faglærer på fagskolen så vi at den eksisterende løsningen for det fysiske systemet ikke sammenfaller med den ovenfor beskrevne teorien vedrørende digital tvilling. Vi har derfor valgt å fokusere på fagskolens behov for å kunne simulere robot bevegelse i Ciros.

3.1.6 Robot Programmeringsmetodikk

Teaching av roboten er et sentralt begrep innenfor robotikk og blir brukt for å bestemme robotens bevegelser. Det finnes flere måter å gjøre dette på, blant annet teach metoden, ledd koordinater, globale koordinater, verktøy koordinater og arbeidsstykke koordinater[8]. Disse metodene blir presentert under, samt ulike metoder for programmeringsmetodikk.

Teach metoden er mest populær teknikken og anses som standardmetoden for programmering av en robot. I teach metoden blir det brukt en bærbar programmeringsenhet som er interfacet mellom operatøren og roboten. Roboten blir

manuelt styrt til ønsket posisjon for å lage programmet. Det er mulig å ta i bruk flere koordinatsystemer for å gjøre programmeringen enklere.

Ledd koordinater brukes når man styrer robot leddene uavhengig av hverandre i ønsket retning. Dette krever flere bevegelser for hvert ledd for å oppnå ønsket posisjon og orientering av verktøyet.

Globale koordinater styrer verktøyets senterpunkt langs X-, Y-, eller Z-aksene til robotens globale aksesystem. Ved hjelp av dette koordinatsystemet kan du også utføre rotasjoner av verktøyet rundt de samme aksene. Robotens globale koordinatsystem blir vanligvis definert ved robotens base.

Verktøy koordinater fungerer på en lignende måte som det globale koordinatsystemet, men i koordinatsystemet er robotens akser festet til verktøyets senterpunkt (Tool Center Point (TCP)) og beveger seg med det. Dette systemet hjelper roboten til å bevege seg i ulike vinkler. Dette oppnås ved å rotere aksene til ønsket vinkel, for så å starte en rettlinjet bevegelse langs den samme aksen.

Arbeidsstykke koordinater blir brukt ved å definere koordinatsystemer som et punkt i rommet innenfor robotens arbeidsområde. Et eksempel hvor dette er nyttig kan være oppgaver der roboten arbeider mellom forskjellige arbeidsstykker og verktøy som kan bevege seg.

Offline-programmering krever en datamaskin med egen programvare for å simulere arbeidsområdet. En av fordelene med denne metoden er at det blir færre stopp i produksjonen. Denne metoden innebærer grundig simulering og ‘’debugging’’, dette gjør det mulig å oppnå en nøyaktig programmering. Denne måten å programmere på passer best til mer komplekse simuleringer hvor presisjon og nøyaktighet anses som viktig.

Online-programmering innebærer å ta roboten ut av produksjon og sette den i programmeringsmodus. I programmeringsmodus er det mulig å endre eller oppdatere funksjonene til roboten. Dette gjør en som oftest ved bruk av en bærbar programmeringsenhet og enkle metoder som å lede robotarmen fysisk fra punkt til punkt. Denne måten å programmere på er lettere å lære enn offline-programmering, men

fungerer i motsetning best for enkle simuleringer. Metoden vil også kunne gå på bekostning av presisjonen til roboten.

Makroprogrammering er en teknikk der programmering instruksjoner eller programkoder genereres som makroer[9]. Makroer er kode-fragmenter som kan gjenbrukes og tilpasses for å automatisere oppgaver. Makroer kan brukes i mange programmeringsspråk, blant annet Java, Python, C++. Makroprogrammering kan være nyttig dersom en skal utføre en repetitiv oppgave eller skrive en kode som er tidkrevende å gjennomføre manuelt. Ved hjelp av makroer kan man automatisere oppgaver som å formatere programkode, legge til importerte biblioteker samt definere funksjoner og variabler. Det er i tillegg mulig å generere komplekse algoritmer og strukturer. Denne typen programmering kan ha ulemper som medfører at koden blir mer komplekst, mindre lesbar og vanskeligere å vedlikeholde. Det er risiko for feil dersom makroer brukes feil eller er avhengig av endringer i hovedkoden. Det er derfor viktig å bruke makroer varsomt og kun når når er nødvendig for effektivitet og produksjon.

Basert på kravspesifikasjonene som ble bestemt sammen med veileder fra Fagskolen er det blitt satt at studentene skal programmere i Ciros 6.4 med programmeringsspråket Melfa Basic IV og benytte en Mitsubishi robotarm. De skal benytte seg av alle de nevnte programmeringsmetodikkene med unntak fra teach-methode og makroporgrammering. Teach-methode er en metode som brukes opp mot et fysisk system med en bærbar programmeringsenhet. Vi mener denne metoden ikke er aktuell fordi studentene skal jobbe opp mot en simulert modell. Vi vurderte å implementere makroprogrammering, men innså fort at en slik programmeringsmetodikk ville medført for høyt fagnivå i forhold til studentenes faglige utgangspunkt.

3.2 Valg av løsning

En sentral del av valg av løsning har vært å definere både teoretisk og praktisk fagnivå på laboratorieoppgaver. Når det gjelder den grunnleggende teorien for robot bevegelse, eksempelvis referanserammer, invers kinematikk og frihetsgrader, så er forståelse av dette viktig for studentens gjennomføring av de praktiske øvelsene. Vi har valgt å beskrive nødvendig teoretisk grunnlag for studentene og de praktiske oppgavene separat.

Når det kommer til digital tvilling, brukes ikke Ciros direkte opp mot roboten. Det brukes et hjelpeprogram som heter Robotics Technology (RT) ToolBox 3. I dette programmet må hver enkelt posisjon bli lagret på nytt via fysisk teach-in. Dette kan ta flere timer å få nøyaktig og blir derfor tidkrevende for både studenter og ansatte. Ettersom studentene jobber mest hjemmefra, ble det utfordrende å få til fysisk simulering. Vi valgte derfor å ha fokus på digital simulering i oppgavene.

Vi valgte å produsere fire oppgavesett hvor det er en økning i vanskelighetsgraden fra laboratorieoppgave 1. til laboratorieoppgave 4. I det fjerde oppgavesettet kan oppgaven løses på to måter. Den ene er en ferdig versjon hvor studentene skal kjøre programmet og forstå hva som skjer, før de deretter svarer på noen spørsmål for å sikre forståelse. Den andre måten vil være inkludere at studentene skal få programmet til å kjøre som ønsket basert på oppgavesettet.

Vi valgte et progressivt løsningsalternativ ettersom vi selv har erfaring med progressive laboratorieoppgaver. Dette valget medfører også en større garanti for å treffe studentene sitt fagnivå. Valget med to versjoner for laboratorieoppgave fire vil sikre at alle studentene kan klare å gjennomføre laboratorieoppgaven.

Til hver laboratorieoppgave vil det være vedlagt et veiledende dokument for gjennomføring og modellering samt et løsningsforslag. Det veiledende dokumentet skal videre inneholde beskrivende tekst og bilder og være enkelt og oversiktlig slik at studentene og fagpersonell kan forstå det. Dette skal vi løse ved å ta skjermbilder underveis i prosessen, og få studenter og andre til å teste ut veiledningen. Dette er viktig for å kunne sikre at dokumentasjonen er informativ nok til å løse oppgavene. Vi skal i tillegg legge til en video til hver laboratorieoppgave som viser hvordan oppgaven kan løses.

Med tanke på dokumentasjon har vi valgt å legge vekt på at denne skal være enkel og oversiktlig. Dokumentasjonen skal inneholde illustrerende bilder og forklarende tekst. Dette er fordi vi først og fremst ønsker at studentene skal klare å gjennomføre oppgaven, men vi ønsker også at oppgavene skal kunne fremme motivasjon og læring. Vi tar i tillegg høyde for ulike språknivå i engelsk og velger derfor å ha all dokumentasjon på norsk.

Valg av tematikk for laboratorieoppgavene er gitt i oversikten under:

1 Tema: Enkel robot bevegelse med teach-in
2 Tema: Enkel flytting av bokser over hinder
3 Tema: Repetisjon av bevegelse inklusiv bruk av fargesensor
4 Tema: Programmering av komplett arbeidsprosedyre inklusiv sortering av bokser

3.3 Vurderinger i forhold til verktøy og HW/SW komponenter

Fagskolen tildelte oss noen satte rammer for hvordan vi skulle løse dette prosjektet. De ønsket at vi skulle benytte oss av programvaren Ciros 6.4 for å introdusere studentene til faget robotikk. Ved bruk av denne programvaren må studentene også lære seg programmeringsspråket Melfa Basic IV som brukes opp mot Mitsubishi robotarmer.

Vi hadde ingen erfaring med disse verktøyene fra tidligere. Dette medførte at vi i starten av prosjektet brukte mye tid til å sette oss inn i bruken av programvaren. Her fikk vi noe støtte fra Fagskolen i form av en kort introduksjon til Ciros og tilgang til e-læringsmoduler fra Festo.

På Høgskulen på Vestlandet (HVL) ble vi introdusert for andre roboter og programvarer. Et av disse programvarene var Robot Operating System (ROS). ROS er et sett med programvare og verktøy for å bygge robotprogrammer. ROS kan anses som et alternativ til Ciros.

Ciros og ROS har de samme målene for programmering av roboter. Det skal kunne simuleres og kobles opp mot fysisk robot. ROS er mer komplekst enn Ciros, og inneholder noe mer tilgjengelig funksjoner og programkoder. Setter vi programmene opp mot hverandre kan de defineres som gode alternativer, men bruk av Ciros vil muligens gi et bedre læringsutbytte for studenter med begrenset erfaring innen programvareprogrammering.

Vi ser oss nødt til å tilpasse oss Fagskolen sitt utstyr. Dette begrunnes med tilgjengelig utstyr som er på Fagskolen, samt hva som er opplæringsplattformen.

Andre program som er brukt er følgende:

- Figma
- Paint 3D
- Windows videoredigering
- Excel

5 Realisering av valgt løsning

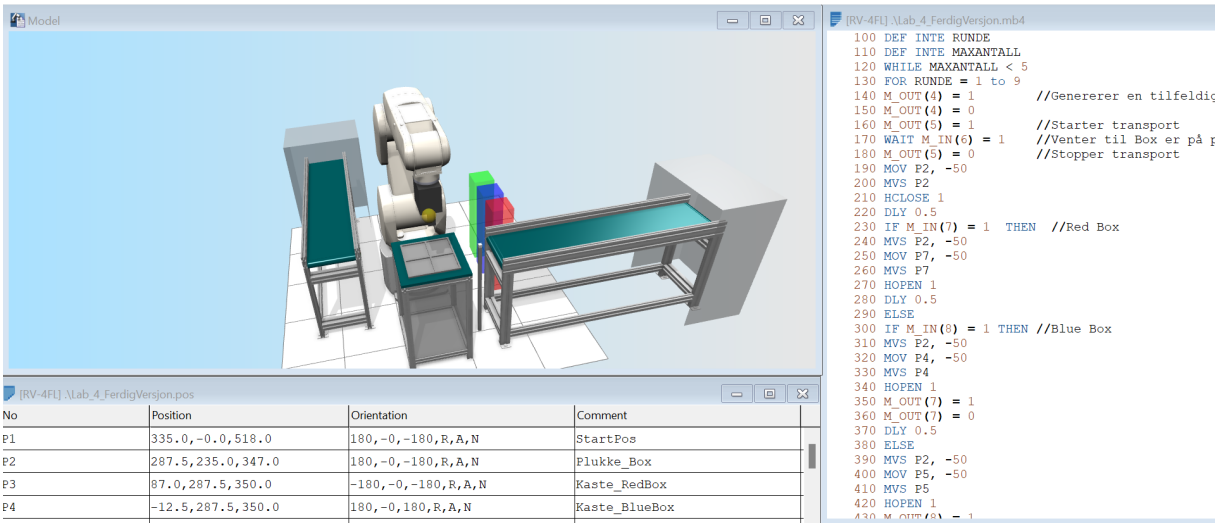
I dette kapittelet gjør vi rede for hvordan modeller i Ciros er blitt utviklet. Vi ser også på hvordan strukturen av både oppgavene og løsningsforslagene i oppgavesettet er formulert. Til slutt presenteres det en kort introduksjon over tematikken til de produserte laboratorieoppgavene.

Alle laboratorieoppgavene tar som nevnt utgangspunkt i Ciros 6.4. Studentene må ha denne programvaren installert på datamaskinen sin, samt et hjelpeverktøy Ivanti. Ivanti er en Virtual Private Network (VPN)-tjeneste som tildeler lisenser for bruk av Ciros på en sikker måte. Laboratorieoppgavene er selve kjernen i vårt bachelorprosjekt, og mye av tiden er brukt på å designe og formulere disse på en god og oversiktlig måte.

5.1 Utvikling av 3D modeller

Utvikling av modeller i Ciros krevde en god del kunnskap vi ikke hadde på forhånd. Vi måtte derfor tilegne oss kunnskap for å nå kravene til Fagskolen. Denne kunnskapen fant vi gjennom ferdige moduler i Ciros og e-læring fra Festo[10]. Her lærte vi oss grunnleggende ferdigheter innen modellering i Ciros og programmering i Melfa Basic IV.

Vi har utviklet egne modeller i Ciros, og selv om noen av dem er inspirert av eksisterende modeller, har vi tilpasset dem til vårt eget formål. Vi har også utarbeidet vedlegg som beskriver hele prosessen i detalj. Disse vedleggene er hovedsakelig rettet mot fagpersonell ved Fagskolen, slik at de kan gjenskape modellene selv uten tidligere erfaring i Ciros.



Figur 2: Utvikling av 3D modell i Ciros 6.4

Da vi startet å modellere i Ciros brukte vi e-læringsmoduler fra Festo. Den første modulen <<CIROS V6 - First Steps>> lærte oss mulighetene Ciros gir opp mot 3D modellering av fabrikk robotarmer, bruk av Ciros som et undervisningsverktøy, generelle funksjoner og kommandoer, samt grunnleggende kunnskap rundt programmering og modellering i Ciros. Den andre modulen <<CIROS - Basics of 3D Simulation>> tok for seg modellering av templater, modellering av transportbånd og sensorer i samhandling med roboten og mer avansert programmering, funksjoner og kommandoer.

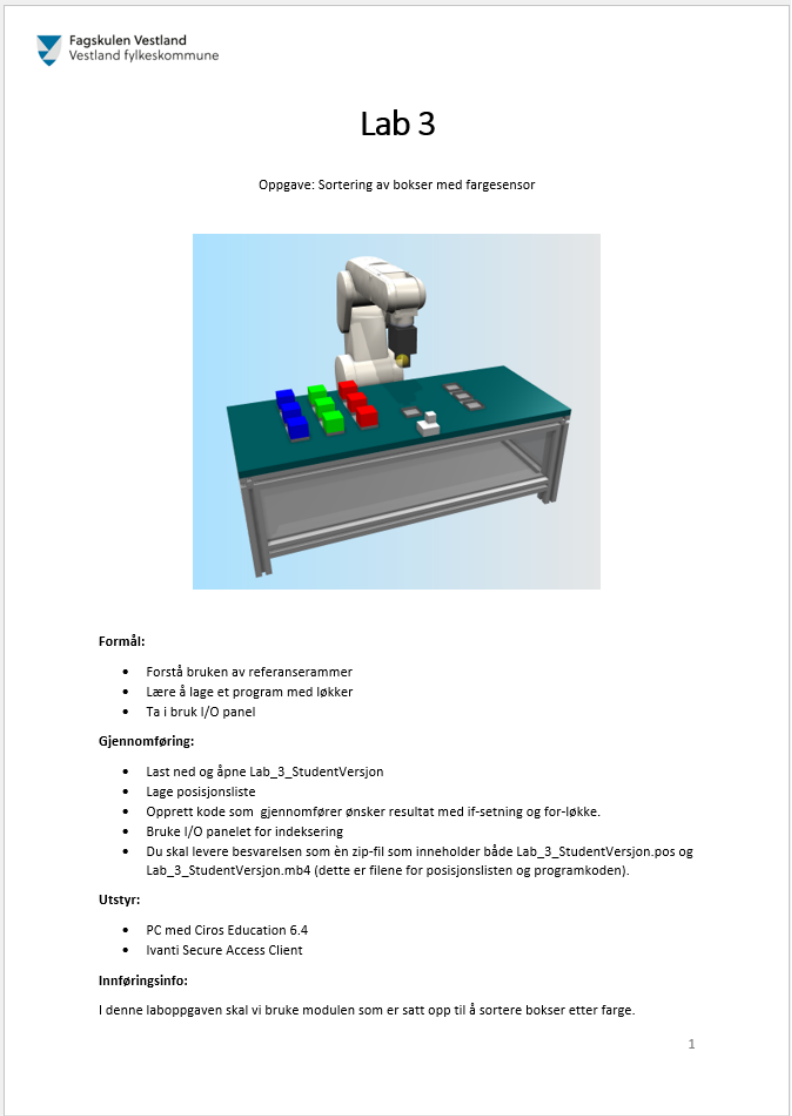
Det var tidkrevende å sette seg inn i 3D modelleringen. Da vi mottok lisensen gikk det mye tid på å fullføre læringsmodulene. Etter disse var gjennomført brukte vi en del tid hvor vi testet ulike modellerings-design for å se hvilke muligheter det var, samt få ideer til 3D modeller som kunne benyttes i laboratorieoppgavene.

Når det gjelder utviklingen av 3D modeller, var modelleringen mest tidkrevende. Dette skyldes at det tok tid å finne frem til ulike objekter for å deretter tilpasse dem til sitt formål. Det var viktig at objektene hadde riktige dimensjoner og posisjon i henhold til roboten.

Hver 3D modell har en prosjektfil som inneholder programkode og posisjonsliste. Det tok mindre tid å programmere roboten ettersom Melfa Basic IV er et grunnleggende robot programmeringsspråk. Det er noe mer komplisert ved bruk av ulike sensorer, men dersom man forstår syntaksen som skal anvendes og har koblet riktig opp i IO-panelet, er det ikke tidkrevende. Posisjonslisten kan opprettes effektivt dersom man benytter grip point eller regner ut matematisk avstander.

5.2 Struktur på oppgavesettene

Vi utformet oppgavesettene etter modelleringen for å sikre at det var fysisk mulig å designe modellen slik vi ønsket. Alle oppgavene er på norsk og formulert på en enkel og oversiktlig måte. Forsiden inneholder oppgavens navn, Fagskolens logo og bilde av modellen. Formål og innføringsinformasjon er inkludert på forsiden for å gi studentene innsikt i hva de skal oppnå og hva som kreves av dem. Ved gjennomføring får studentene kortfattet informasjon om hvordan de skal løse oppgaven og hva som er hovedmålet. I utstyrskravene trenger studentene kun en PC med programvaren Ciros Education 6.4 og hjelpeprogrammet Ivanti Secure Access Client for sikker overføring av lisens. Eksempel på formatering av forsiden er gitt under.



Fagskulen Vestland
Vestland fylkeskommune

Lab 3

Oppgave: Sortering av bokser med fargesensor

Formål:

- Forstå bruken av referanserammer
- Lære å lage et program med løkker
- Ta i bruk I/O panel

Gjennomføring:

- Last ned og åpne Lab_3_StudentVersjon
- Lage posisjonsliste
- Opprett kode som gjennomfører ønsker resultat med if-setning og for-løkke.
- Bruke I/O panelet for indeksering
- Du skal levere besvarelsen som èn zip-fil som inneholder både Lab_3_StudentVersjon.pos og Lab_3_StudentVersjon.mb4 (dette er filene for posisjonslisten og programkoden).

Utstyr:

- PC med Ciros Education 6.4
- Ivanti Secure Access Client

Innføringsinfo:

I denne laboppgaven skal vi bruke modulen som er satt opp til å sortere bokser etter farge.

1

Figur 3: Eksempel på formatering av forside på laboratorieoppgave

5.2.1 Formulering av oppgaver

Målet med formuleringen av laboratorieprøvene har vært å introdusere oppgavene på en oversiktlig og forståelig måte. For å oppnå dette målet er det presentert flere mindre oppgaver som til slutt medfører en komplett besvarelse. Vi har inkludert bilder og veileding i oppgaveteksten for å hjelpe studentene til å visualisere hvordan resultatet skal se ut og finne frem til ulike ting. Noen oppgavesett inneholder kontrollspørsmål for å sikre at studentene har forstått oppgaven i henhold til pensum. Alt i alt har vi laget oppgavesettene på en måte som gir studentene all nødvendig informasjon for å fullføre oppgaven og lære konseptet på en effektiv og motiverende måte.

5.2.2 Utarbeiding av løsningsforslag

Vi produserte løsningsforslagene parallelt med oppgavesettet for å sikre en tett sammenheng mellom dem. Utgangspunktet var at studentene skulle kunne løse oppgavesettet på egen hånd, men vi har også tatt høyde for at noen kan ha vanskeligheter med dette. Derfor er løsningsforslagene utformet på en måte som ikke bare viser svarene, men også hvordan man kan komme frem til dem.

Den ferdige versjonen er en ferdigstilt modell som inneholder den endelige løsningen og den kan være til nytte hvis studenten sliter med å løse oppgaven ved hjelp av løsningsforslaget. De kan bruke den ferdige modellen til å prøve å svare på eventuelle kontrollspørsmål og fortsatt forstå konseptet.

5.3 Oversikt over oppgavesett

Dette delkapittelet gir en oversikt over de enkelte oppgavesettene som er utarbeidet.

5.3.1 Tema: Enkel robot bevegelse med teach-in

Formålet med denne oppgaven er å gjøre studenten kjent med fagbegrepet kinematikk. De vil også gjøre seg kjent med en enkel metode for programmering av roboten.

Oppgaven går ut på flytte roboten manuelt ved hjelp av 3D modellen og registrere de enkelte posisjonene og orienteringene. Studenten skal basert på dette utvikle en kort programkode for å bevege roboten til lagrede posisjoner.

5.3.2 Tema: Enkel flytting av bokser med endring av posisjon lister

I denne oppgaven er formålet at studentene skal forstå og bruke posisjonslister. De skal kunne anvende de posisjonene som allerede er gitt, og modifisere disse til riktige posisjoner. De skal også opprette helt nye posisjoner ved å se sammenhengen mellom de ulike posisjonene.

Oppgaven går ut på å flytte bokser over en vegg. Veggen er inkludert i oppgaven ettersom studentene skal forstå at de ikke kan flytte boksene direkte, men må anvende en hjelpe posisjon. Dette vil også gi studentene en større forståelse av begrensningene for robotarmen.

5.3.3 Tema: Repetisjon av bevegelse inklusiv bruk av fargesensor

I denne oppgaven skal studentene ta i bruk metoder som de har brukt i de tidligere oppgavene. Blant annet fylle inn posisjonsliste ved bruk av flere referansepunkter og bruke for- og if-setninger når de skal programmere.

Oppgaven går ut på å sortere bokser på farge. Studentene skal ved bruk av en fargesensor programmere en kode som kan sortere etter ønsket farge. De må da ta i bruk I/O panelet i Ciros for å identifisere index nummeret til ønsket farge.

5.3.4 Tema: Programmering av komplett arbeidsprosedyre inklusiv sortering av bokser

I denne oppgaven skal studentene ta i bruk tidligere erfaringer for å simulere en tiltenkt arbeidsprosedyre. Her kreves det at studentene har god forståelse for logisk programmering i Melfa Basic IV.

I denne modellen ønsker vi å simulere en enkel arbeidsprosess som skal automatiseres. Det blir produsert bokser av tilfeldige farger inn på et transportbånd. Studenten skal programmere en programkode som sorterer bokser med ønsket farge på en palle. For så å sende de videre til neste transportbånd. Bokser med uønsket farge skal kastes i simulert søppelspann med samme farge.

Denne laboratorieoppgaven blir også presentert som en tolk- og forklar-oppgave. Det vil si at studentene blir tildelt en ferdig versjon av programmet. De skal kommentere kort hva de forskjellige avsnittene i koden utfører, samt hva som skjer når ulike inn- og ut-signaler blir satt til høy og lav.

6 Testing

Kapittelet gjennomgår ulike tester gjennom prosjektet. Blant annet testing av modeller, oppgavesett og løsningsforslag av faglærer og til slutt av studenter som hadde robotikk faget parallelt med prosjektet.

6.1 Testing under produksjon

Det ble tidlig oppdaget at Ciros sin feilsøking funksjon er begrenset og lite informativ. Vi så oss derfor nødt til å gjennomføre hyppig testing under produksjon for å detektere feil fortløpende. På denne måten unngikk en feilsøking av store programmer. Det bidro også til oppdagelse av eventuelle feil som studentene kan møte på.

6.2 Test av oppgavesett og løsningsforslag

I starten av prosjektet satte vi opp en fremdriftsplan. I fremdriftsplanen satt vi frister for når vi skulle teste programmet opp mot faglærer og studenter på Fagskolen. Dette ble gjort for å sikre at nivået på oppgavene passer fagkunnskapene til studentene og for at oppgavesettet og løsningsforslaget skulle være forståelig og motiverende å fullføre.

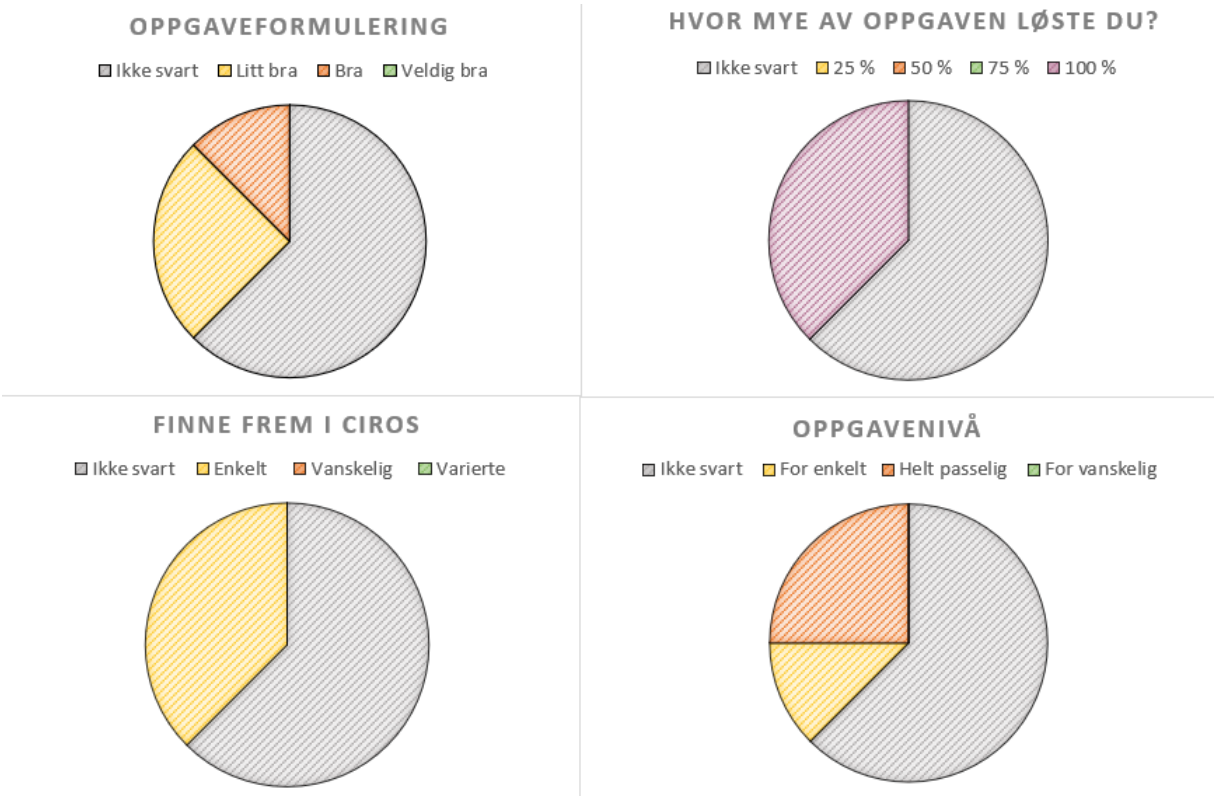
6.2.1 Test med faglærer

Etter at første utkast av laboratorieoppgave 1 og 2 var ferdig fikk vi veilederen vår fra Fagskolen til å gjennomføre disse. Tilbakemeldingene var generelt positive, men vanskelighetsgraden var generelt litt for høy for studenter på Fagskolen. Ellers ønsket han også at oppgavesettene gjerne kunne inneholde mer visuell forklaring i form av figurer og bilder.

Basert på tilbakemeldingene valgte vi å endre laboratorieoppgave 1 og 2 til laboratorieoppgave 2 og 3. Tilbakemeldingene angående det visuelle fikk vi også endret på og oppgavene ble generelt mer oversiktlig og forståelig etter endringene.

6.2.2 Test med student

Vi opprettet en spørreundersøkelse til Lab 2. Studentene fikk da tildelt en link hvor de skulle besvare noen avkrysningsspørsmål og noen åpne spørsmål. Diagrammene under viser følgende resultat av avkrysning:



Figur 4: Diagram for avkyssningsspørmål fra studentundersøkelse

Svar på andre spørsmål var som følger:

Dette er noe forkortet, med de viktigste tilbakemeldingene. Full undersøkelse kan du finne her: [Studentundersøkelse]

Spørsmål	Student 1	Student 2	Student 3
Hva synes du var bra med oppgaven?	"Bra layout på oppgavetekst. Passe stor oppgave."	"Helt grei innføring av hvordan aksesystemet virker."	"Tipsene, bilder, veileding til å finne frem posisjon-liste og programmeringsvindu et."
Hva synes du var negativt i oppgaven?	Ukjent syntaks, kunne vært mer informasjon om dette. Langt og repeterende program. Tips om linjenummering	Oppgaven starter med nedlastning av "LAB_1". Flere hint om hvordan programmet bør være.	Oppgavetekst litt rotete og ustrukturert. Bedre skille mellom oppgavetekst og tips. Litt mye med 9 bokser. Ble mye copy paste. Kanskje bruke mindre klosser over flere hinder?

Andre tilbakemeldinger?	Gjennomfører sier vi skal laste ned Lab_1, riktige skal vel være Lab_2.	Litt vel enkel oppgave. Lett å finne posisjoner når mange allerede lå inne. Bygge videre på koden med flere funksjoner som en bonus oppgave.	Feil i oppgavetekst. Står Lab_1 og ikke Lab_2. Kanskje ha med hvilken funksjon som er tenkt i oppgaven, så kan studenten lese seg opp på syntaksen/virkemåten .

Tabell 1: Forkortet tilbakemeldinger fra studentundersøkelse

Slik det fremstår basert på de studentene som har gjennomført laboratorieoppgaven og svart på undersøkelsen fremstår laboratorieoppgave 2 som en oppgave med passelig vanskelighetsgrad. Basert på avkrysningsspørsmålene får vi positive tilbakemeldinger, noe som gjenspeiler målsettingen våres. Basert på de åpne spørsmålene kunne vi hatt mer informasjon for syntaks og linjenummerering. Oppgaveteksten kan modifiseres for å ha bedre skille mellom hva de skal gjøre i oppgaven og tips. Det går igjen for studentene at det står “Lab_1” ved gjennomføring-informasjon. Dette er en feil som har blitt oversett da vi endret første utkastet av laboratorieoppgave 1 til å bli laboratorieoppgave 2.

I samtale med faglærer til studentene har det kommet frem at det er flere studenter i klassen som ikke har klart å løse oppgaven og derfor heller ikke svart på undersøkelsen. Som man ser utgjør dette over 50% av diagrammene. Undersøkelsen er dermed ikke komplett grunnet for lite data. Basert på manglende tilbakemelding fra en stor gruppe av studentene og de tilbakemeldingene vi har fått, velger vi å konkludere med at oppgaven er for avansert for studentene på dette fagnivået.

7 Diskusjon

I dette kapittelet blir arbeidet mot fremdriftsplanen diskutert, i tillegg ser vi på ulike utfordringer som dukket opp underveis.

7.1 Framdrift

Gant-planen viser et visuelt bilde over fremdriftsplanen for arbeidsoppgaver og frister. Denne ble utformet i startfasen av prosjektet. Hensikten var å gi oss en oversikt over hvordan vi skulle fordele tiden vår i et større prosjekt. Store deler av denne Gant-planen har vært veiledende og vi har alltid jobbet opp mot å fullføre arbeidskrav og viktige hendelser til satt tidsfrist.

#	Aktivitet	Resultat	Planl. Arbeid [timer]	Ansvarlig	w1 2/1 6/1	w2 9/1 13/1	w3 16/1 20/1	w4 23/1 27/1	w5 30/1 3/2	w6 6/2 10/2	w7 13/2 17/2	w8 20/2 24/2	w9 27/2 3/3	w10 6/3 10/3	w11 13/3 17/3	w12 20/3 24/3	w13 27/3 31/3	w14 3/4 7/4	w15 10/4 14/4	w16 17/4 21/4	w17 24/4 28/4	w18 1/5 5/5	w19 8/5 12/5	w20 15/5 19/5	w21 22/5 26/5	w22 29/5 2/6	w23 5/6 9/6	w24 12/6 16/6	
				År	2023																								
				Uke																									
				Man																									
				Fre																									
7	Eksamensperiode januar		0	BL/AVM							Begreset jobbing																		
8	Forstudie arbeid	Ferdig rapport	90	BL/AVM		~75-100 t pr stud																							
9	Metode undervisning	Vært på undervisningen	10	institutt		2t	2t	2t		2t				2t															
10	Forstudie innlevering	Levert forstudie	0	BL/AVM																									absolutt frist
11	Modellering og software i CIROS	Fungernde modeller/sw	100	BL/AVM						30t	10t	30	30																
12	Evaluering av produsert innhold. Testing av produkt/simulator	Fremlagt produsert innhold	1	BI/AVM + fagskole																									
13	Forbedre og avansere produsert innhold. Forberedelse til midtveispresentasjon	Bedre laboppgaver	120	BL/AVM										30	30	30	30	Påske											
14	Test av veiledene dokumentasjon	Tilbakemelding	1	BL/AVM + student																									
	Mitdveis presentasjon	Midtveis presentasjon	2	BL/AVM																									
15	Forbedring etter midtveispresentasjon		60	BL/AVM															30	30									
16	Oppgaveskriving og plakatdesign	Ferdig rapport og plakat	105	BL/AVM																	30	25	25	25					
18	Refleksjonsnotat	Innlevert refleksjonsnotat	1	BL/AVM																									
19	Ordinær eksamen		0	BL/AVM																									
20	Konte eksamen		0	BL/AVM																									
21	Bachelor oppgave innlevering	Levert bachelor	0	BL/AVM																									absolutt frist
22	Bachelor oppgave presentasjon	Presentert Bachelor	25	BL/AVM																					25				avtales med sensorer
23	EXPO23		0	BL/AVM																									alle må stille
24	Avslutningsfest		0	BL/AVM																									

	Forkortelser
	BL = Bård Løvik
	AVM = Alexandra Viktoria Monsen
◆	Rød boks= Viktige hendelser

Figur 5: Fremdriftsplan

Vi har lagt i henhold til fremdriftsplanen med unntak av testing av laboratorieoppgaver knyttet til studentene. Dette ble forsinket grunnet sen informasjon om når studentene skulle ta faget Robotikk. Det er blitt testet en laboratorieoppgave mot studentene, mens hovedmålet var å få testet alle laboratorieoppgavene og deretter forbedre disse.

I henhold til B.3 Risikoliste forekom sykdom, samt programvare problematikk. Sykdom har medført selvstudie og begrenset med jobbing. Programvare problematikken har i hovedsak gått ut på at det kun er én lisens på studioversjonen i Ciros. Programvinduer og

posisjonslister kan lagres i Ciros Education og gjorde det derfor mulig å få gjort noe uten studio-lisens. Til tross for sykdom og programvare problematikk har dette ikke påvirket vår opprinnelige fremdriftsplan.

7.2 Utfordringer underveis

I dette kapittelet beskrives ulike problemstillinger underveis i prosjektet.

7.2.1 Samsvar mellom tittel og problemstilling

Opprinnelig var navnet på vår bacheloroppgave “Ciros og robot i et digitalt tvillingoppsett". Ettersom vi fokuserer på simulerte modeller, og ikke digital tvilling, vurderes denne tittelen på oppgaven som misvisende. Vi mener “Ciros og robot i et digitalt simuleringssystem” er en mer eksakt beskrivelse i henhold til hva oppgaven våres omhandler.

7.2.2 Modellering i Ciros

Vi måtte starte med å lære oss programvaren Ciros. Dette var essensielt i prosjektet vårt. E-læringen vi brukte har spilt en viktig for at vi skulle lære oss dette. Det hadde vært ønskelig med flere av disse modellene og gjerne at de eksisterende modellene var bedre forklart, med bilder og forklarende tekst.

Det har vært mye prøving og feiling ved utforming av modellene til laboratorieoppgavene. Store deler av dette skyldes at det er for lite tilgjengelig informasjon, dermed har dette også vært tidkrevende. På den andre siden har dette fremmet læringen vår og medført at laboratorieoppgavene er unike med noe inspirasjon fra læringmodulene.

7.2.3 Testing og vurdering

De oppgavene som er laget har blitt vurdert som vanskelige av faglærere. I ettertid har vi tenkt at vi burde fått et eksempel på laboratorieoppgaver som studentene har hatt tidligere år for å treffe vanskelighetsgraden bedre. Basert på mangel på tilbakemeldinger fra studentene er oppgaven for vanskelig til å være en laboratorieoppgave 2.

Basert på tilbakemelding fra veileder på Fagskolen er vi klar over at oppgavene trolig er for vanskelig for de fleste studentene. Vi valgte derfor å endre dette så langt det lot seg gjøre. Dersom vi hadde hatt noen eksempel på tidligere oppgaver studentene hadde fått tildelt, ville dette trolig bidratt til mer tilsvarende nivå på laboratorieoppgavene fra starten av.

7.3 Videre arbeid

Basert på arbeidet med oppgaven har vi fått forståelse for at programvaren Ciros er et godt verktøy for å kombinere med hjemmeundervisning i tillegg til undervisning på fagskolen til det fysiske systemet. For videre arbeid med oppgavesettene har vi følgende forslag.

1. Utforme en introduksjonsvideo for oppsett av Ciros. Dette vil gjøre det enklere og mer effektivt for studentene og fagpersonell å installere og ta i bruk Ciros 6.4.
2. Lage digitale løsningsforslag for studentene i henhold til eksisterende og nye laboratorieoppgaver.
3. Produsere enklere introduksjonsoppgaver. Dette innebærer lavere fagnivå, og mer innførende teoretisk kunnskap.
4. Bruk av Ciros som en digital tvilling. Dette innebærer å finne løsninger på hvordan du kan koble modellene i Ciros opp mot et fysisk system.

Som en generell forbedring kan det legges bedre til rette for fjernopplæring og å opprette et ‘’Q&A’’ dokument hvor man kan finne svar på utfordringer i opplæringssituasjonen.

8 Konklusjon

Prosjektoppgaven har blitt gjennomført i tett dialog med Fagskolen og har videre fokusert på utvikling og testing av fire laboratorieoppgaver i Ciros som skal kunne gjennomføres av studenter både hjemme og på skolen. En av laboratorieoppgavene har blitt testet på studenter der vi har evaluert tilbakemeldingene fra studentene. I tillegg brukte vi mye tid på å tilegne oss ferdigheter i robotmoduleringsverktøyet Ciros i startfasen av prosjektet, da dette var ferdigheter vi ikke hadde på forhånd.

Robotmoduleringsverktøyet Ciros er en kompleks programpakke som krever god forståelse av 3D animering og robotteori. Programpakken er likevel velegnet for å lære robotprogrammering og å teste ut egenutviklede robotkoder.

Fire laboratorieoppgaver er utviklet og beskrevet. Disse omhandler enkel robotbevegelse, sortering og bruk av robotsensor. Det ble gjennomført en vurdering av fagnivå, både når det gjelder robotteori og praktisk programmeringserfaring. I laboratorieoppgavene er det også inkludert robotteori slik som invers kinematikk og referanserammer samt metoder for robotprogrammering. Det er i tillegg laget en introduksjonsvideo til en av laboratorieoppgavene.

Ved utvikling av de to første laboratorieoppgavene ble nivået på disse vurdert til å være for vanskelig som startoppgaver av faglærer. Det ble derfor laget en ny laboratorieoppgave som har lav vanskelighetsgrad der det kun gjøres en liten endring på eksisterende posisjoner og programmeringskode. I tillegg ble en av laboratorieoppgavene testet på studenter. Dette ble gjort ved at de skulle gjennomføre laboratorieoppgavene etterfulgt av å svare på en spørreundersøkelse. Under arbeidet med testingen av laboratorieoppgavene var det utfordrende å få tilstrekkelig med svar fra alle studentene på spørreundersøkelsen ettersom flere av studentene hadde problemer med å gjennomføre laboppgaven.

Når det gjelder videre arbeid med laboratorieoppgavene er det viktig at fagnivå stemmer overens med fagskolens pensum. Det kan med fordel også utvikles en forbedret introduksjons informasjon om Ciros for Fagskolens studenter som gjennomføres i forkant av laboratorieoppgavene. Det kan også være gunstig å utvikle videobasert løsningsforslag for alle laboratorieoppgavene. Vi anser det som viktig at dette gjøres både for de tre resterende laboratorieoppgavene og eventuelt fremtidige laboratorieoppgaver.

Referanser

[1] "Om Fagskulen Vestland," Fagskulen, tilgjengelig: https://www.fagskulen.no/om-oss/om-fagskulen-vestland/om-fagskulen/, 24. april 2023.

[2] "Robot Coordinate Frames and Points," SolisPLC, tilgjengelig: https://www.solisplc.com/tutorials/robot-coordinate-frames-and-points, 24. april 2023.

[3] "Inverse Kinematics," Motion Planning and Robotics Lab, University of Illinois at Urbana-Champaign, tilgjengelig: https://motion.cs.illinois.edu/RoboticSystems/InverseKinematics.html, 24. april 2023, kapittel 6.

[4] "Robot," Nasjonalbiblioteket, tilgjengelig: https://www.nb.no/items/23fae581cfabda948d6e7ffd447be2ea?page=173&searchText=robot, 24. April 2023, kapittel 4.4.

[5] "The Business Model for Robot Food," Precoil, tilgjengelig: https://medium.com/precoil/the-business-model-for-robot-food-136d461fbfb0, 10. april 2023.

[6] S. Gupta, "Degrees of Freedom of a Robot," Mecharithm, tilgjengelig: https://www.mecharithm.com/degrees-of-freedom-of-a-robot/, 10. april 2023

[7] "Digital Twin for Parcel Sortation," BEUMER Group, tilgjengelig: https://www.beumergroup.com/knowledge/cep/digital-twin-parcel-sortation/, 25. april 2023.

[8] "Methods of Defining Positions in Space," TheTeche, tilgjengelig: https://theteche.com/methods-of-defining-positions-in-space/, 25.april 2023.

[9] "Subprograms, Macros, and Parametric Programming for CNC Machining," Thomasnet, tilgjengelig: https://www.thomasnet.com/articles/custom-manufacturing-fabricating/subprograms-macros-and-parametric-programming-for-cnc-machining/, 25.april 2023.

[10] "Festo," Festo, tilgjengelig: https://lx.festo.com/en/home , 1.februar 2023.

Appendiks A Forkortelser og ordforklaringer

DOF	Degrees of freedom (Frihetsgrader)
HVL	Høgskulen på Vestlandet
MELFA	Mitsubishi Electric Factory Automation
ROS	Robot Operating system
RT	Robotics Technology
TCP	Tool Center Point
VPN	Virtual Private Network

Appendiks B Prosjektledelse og styring

B.1 Prosjektorganisasjon

Prosjektleder: Bård Løvik og Alexandra Viktoria Monsen

Ansvarsområdet er blitt delt ut slik at arbeidsmengden er jevnt fordelt. Vi har samarbeidet der det har vist seg hensiktsmessig.

Inndeling:

- Bård Løvik: Produksjon og utforming av oppgavesett og rapportskriving.
- Alexandra Viktoria Monsen: Produksjon og utforming av oppgavesett, test av oppgavesett, digitalt løsningsforslag og rapportskriving.

B.2 Fremdriftsplan

Fremdriftsplanen vår baserer seg på Gant-skjemaet hvor det vises hvor lang tid det er planlagt å bruke på de forskjellige arbeidsoppgavene. Skjemaet er blitt fulgt i stor grad, men falt litt bakpå etter påske.

Videre blir det presentert hvordan vi har fordelt arbeidstimer per person og studiested.

MANDAG	TIRSDAG	ONSDAG	TORSDAG	FREDAG	TOTALT
8	8	5	5	5	31

Tabell 2: Timefordeling

MANDAG	TIRSDAG	ONSDAG	TORSDAG	FREDAG
FAGSKOLEN	FAGSKOLEN	HVL	SELVSTUDIE	HVL/SELVSTUDIE

Tabell 3: Studiested

B.3 Risikoliste

RISIKO	SANNSYNLIGHET	ALVORLIGHET	KONSEKVENS	TILTAK
sykdom/skade	Høy	Middels	Hindrer fremdrift i henhold til plan	Selvstudie
Konflikt med veileder/oppdragsgiver	Lav	Høy	Dårligere resultat enn ønsket	Kommunikasjon
Konflikt i gruppe	Lav	Middels	Dårligere samarbeid/fremdrift	Kommunikasjon
Software problematikk	Middels	Middel-Høy	Stopp i produksjon/fremdrift	Lisens: Legge en plan for lisensbruk Modellering: Feilsøking og research

Tabell 4: Risikoliste

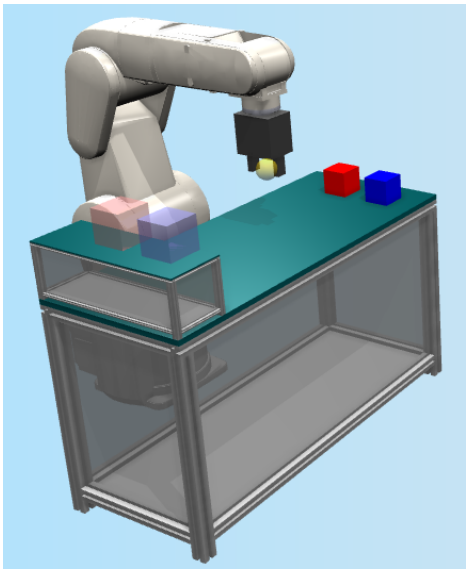
Appendiks C Laboratorieoppgaver

Bilder og figurer i oppgavesettene er i hovedsak skjermbilder fra Ciros 6.4. Tegninger er utformet med tegneprogram.

C.1 Lab 1 - Innføring i robot bevegelser

LAB 1

OPPGAVE: INNFØRING TIL ROBOT BEVEGELSE



Formål:

- Forstå og bruke posisjonslister
- Se sammenheng mellom teach-in og posisjonsliste
- Forstå og anvende frihetsgrader

Gjennomføring:

- Last ned og åpne Lab1_oppgaveversjon
- Finne posisjonen til plassering av rød boks
- Du skal levere Lab1_OppgaveVersjon.pos og Lab1_OppgaveVersjon.mb4 som zip fil. Dette tilsvarer posisjonslisten din, og programmeringsvinduet.
- Du skal levere svar på spørsmål som PDF.

Utstyr:

- PC med Ciros Education 6.4
- Ivanti Secure Access Client

Innføringsinfo:

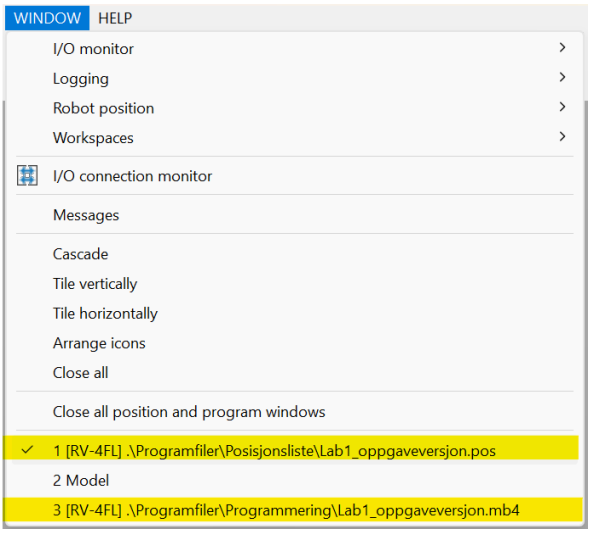
I denne lab oppgaven er bevegelsen for blå boks gitt. Denne er gitt steg for steg frem til posisjonen hvor den skal plasseres. Basert på hvordan denne er gitt i posisjonslisten, skal du kunne få en anelse om hvordan å manere robotarmen slik at den flytter den røde boksen til sin plassering.

TIPS: Lab 1 → Dokumentasjon → Teori. Her kan du finne teori som gjør oppgaven lettere å løse, samt å svare på spørsmålene i slutten av oppgaveheftet.

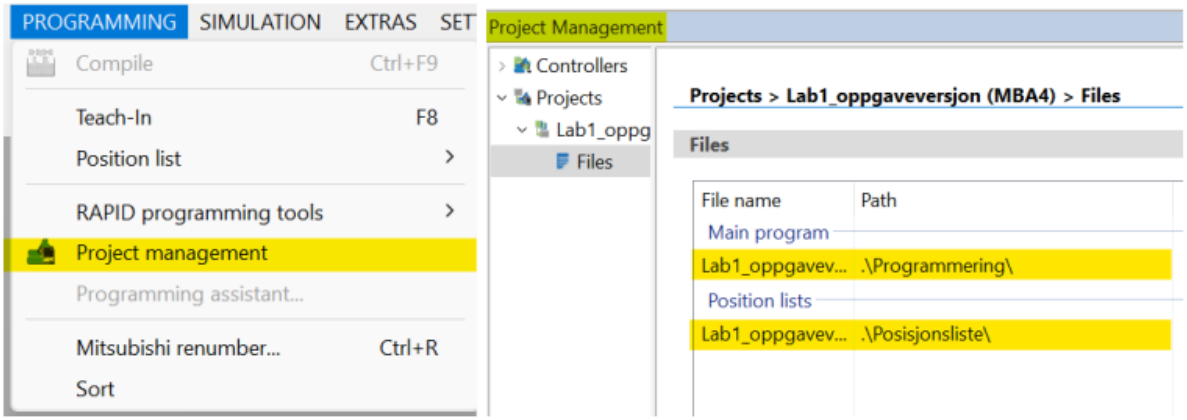
1: Åpne filen «Lab1_oppgaveversjon».

2: Finn posisjonslisten. Se over posisjonene og prøv å se en sammenheng mellom bevegelsen.

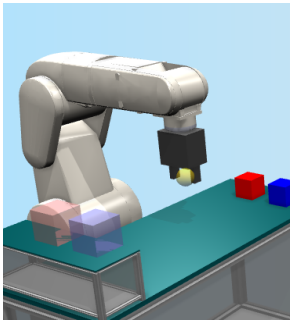
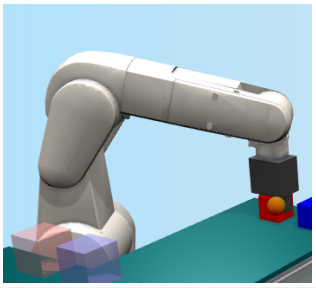
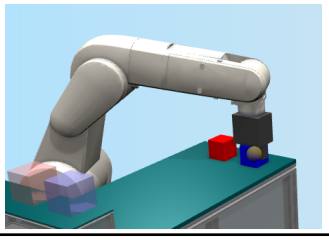
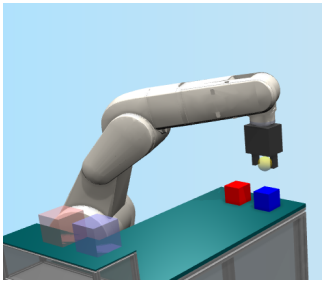
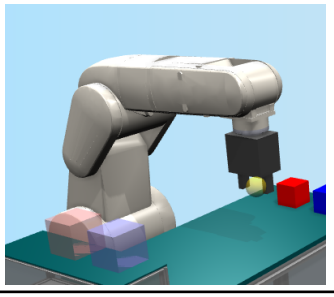
Du skal kunne finne posisjonslisten ved å klikke på «Window». Her er posisjonslisen og programmeringsfilen markert.

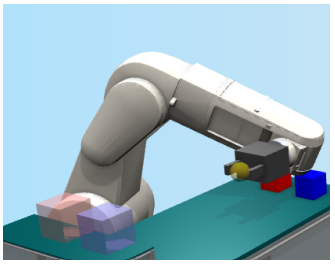
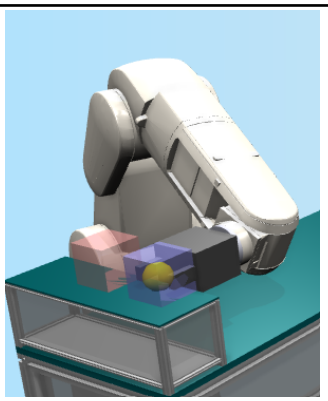


Dersom disse ikke ligger der, kan du finne dem på følgene måte:

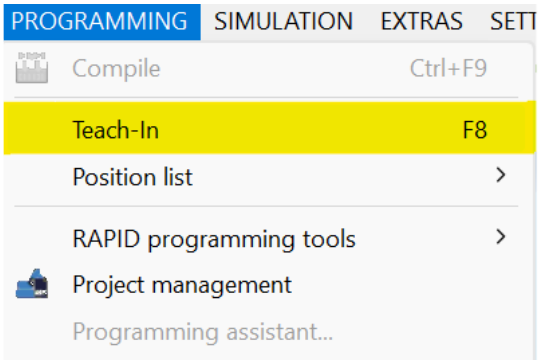


3: Kjør programmet for å se hvordan robotarmen beveger seg steg for seg når den flytter den blå boksen. De viktigste posisjonene vil her være «POB», «P2», «P3», «P4» og «PB». Følgene posisjoner skal gjøre følgende:

Punkt	Hva skjer?	Bilde fra pos:
P1	Hoved posisjonen til roboten	
POR	Plukker opp den røde boksen	
POB	Plukker opp den blå boksen	
P2	Robotarmen beveges i Z aksen. Tilsvarer nå samme Z-verdi som plasseringspos isjonen PB.	
P3	Endring i Y akse basert på P2. Setter roboten i lik posisjon som P1, men parallelt med blå boks.	

P4	Lik som P3 posisjonen, men her roterer vi roboten for å få ny vinkel på griperen.	
PB	Plasseringen til den blå boksen. Basert på posisjon P4, med endring i Z akse.	

4: Gå til Teach-in. Det gjør du på følgende måte:



Svar på følgende spørsmål for å sikre at du har forstått alt.

Spørsmål 1:
Når du kjører koden for den blå boksen, og sammenligner med koden du laget for den røde boksen, hvilken er mest effektiv? Hvorfor?

Spørsmål 2:
Hvilken form for kinematikk brukes i oppgaven?

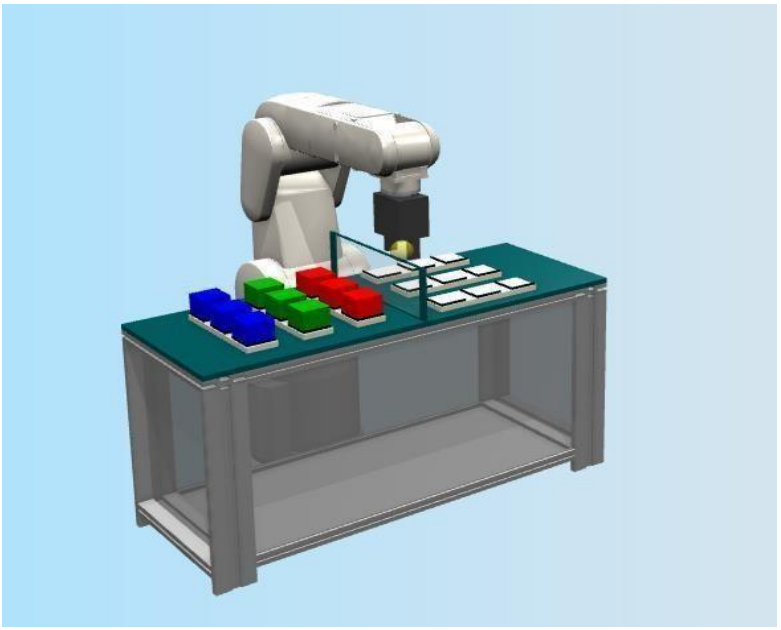
Spørsmål 3:
Når du anvender teach-in for å nå plasseringsposisjonen, er det noen posisjoner roboten ikke kan flytte eller rotere seg til. Hva er årsaken til dette?

Spørsmål 4:
Hvor mange frihetsgrader har denne roboten?

C.2 Lab 2 - Bokser over hinder

LAB 2

OPPGAVE: FLYTTE BOKSER OVER VEGG



Formål:

- Forstå og bruke posisjonslister
- Programmere program basert på posisjonslisten

Gjennomføring:

- Last ned og åpne Lab2_oppgaveversjon
- Lage posisjonsliste
- Skrive program
- Du skal levere Lab2_OppgaveVersjon.pos og Lab2_OppgaveVersjon.mb4 som ZIP fil. Dette tilsvarer posisjonslisten din, og programmeringsvinduet.

Utstyr:

- PC med Ciros Education 6.4
- Ivanti Secure Access Client

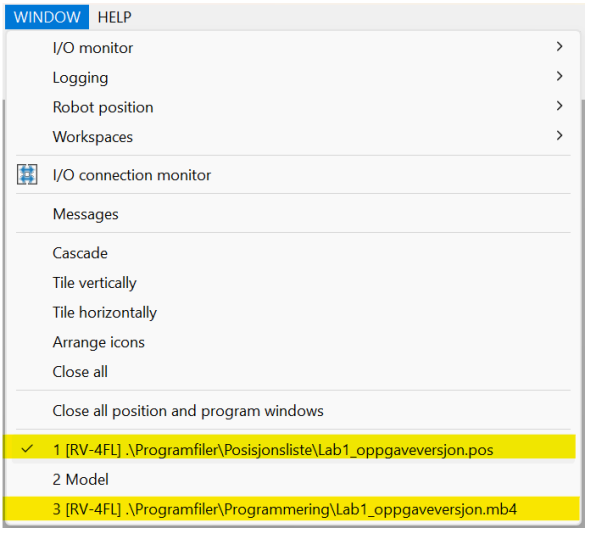
Innføringsinfo:

I denne laboppgaven skal vi bruke modulen som er satt opp til å flytte bokser over vegg. Dette er gjentakende.

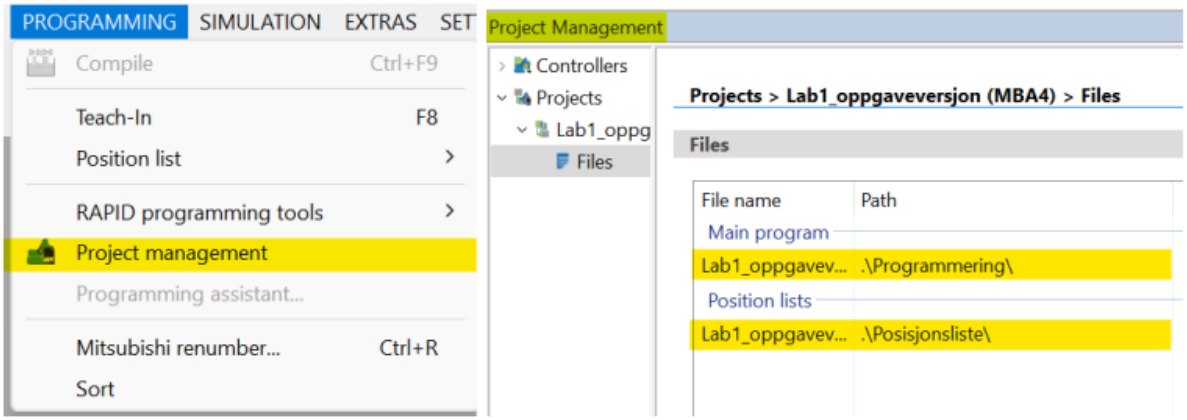
1: Åpne filen «Lab2_oppgaveversjon»

2: Finn posisjonslisten. Start posisjon for hver farge er lagt inn, samt en plasseringsposisjon for en boks. Finn og opprett manglene posisjoner for boksene.

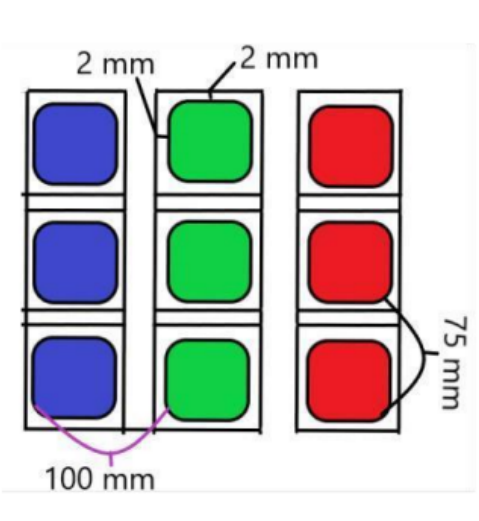
Du skal kunne finne posisjonslisten ved å klikke på «Window». Her er posisjonslisen og programmeringsfilen markert. Eksempelbildet er hentet fra Lab1.



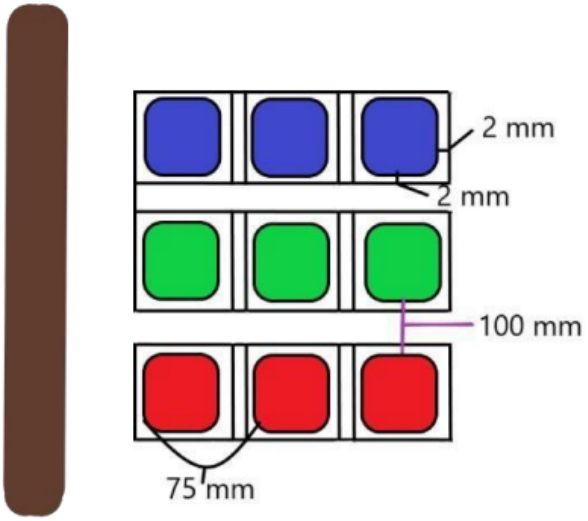
Dersom disse ikke ligger der, kan du finne dem på følgene måte:



3:



Figur 1 Startposisjoner



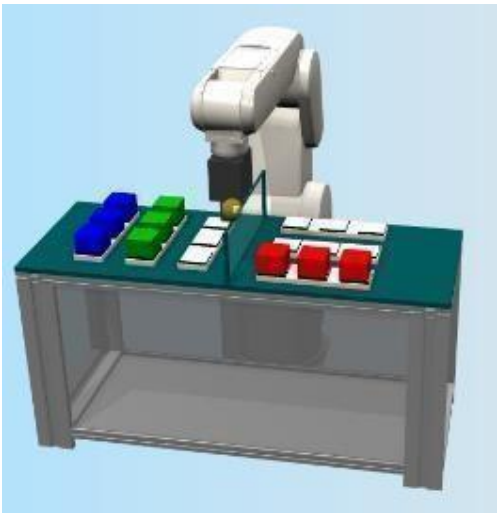
Figur 2 Plasseringsposisjoner

TIPS: Ta utgangspunkt i avstanden mellom palettene for å finne posisjonen. Svarte differanse linjer er oppgitt i section og rosa streker er oppgitt i world.

Åpne programmerings vinduet. Hele programmet skal flytte boksene frem og tilbake til programmet stoppes. Vi deler opp koden for bedre oversikt.

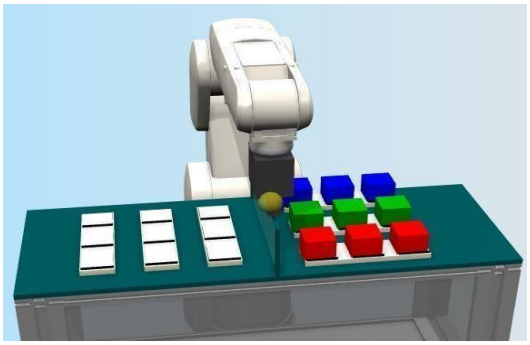
TIPS: Ha programvinduet oppe ved siden av posisjonslisten.

1. Skriv en kode som flytter de røde fra startposisjonen til plasseringsposisjonen.



2. Gjenta prosessen du nettopp gjorde, men for både de blå og grønne boksene.

3. Bygg programmet og kjør. Se at alt fungerer som det skal. Boksene skal nå bli plukket opp fra startposisjonene og plasseres på palettene.

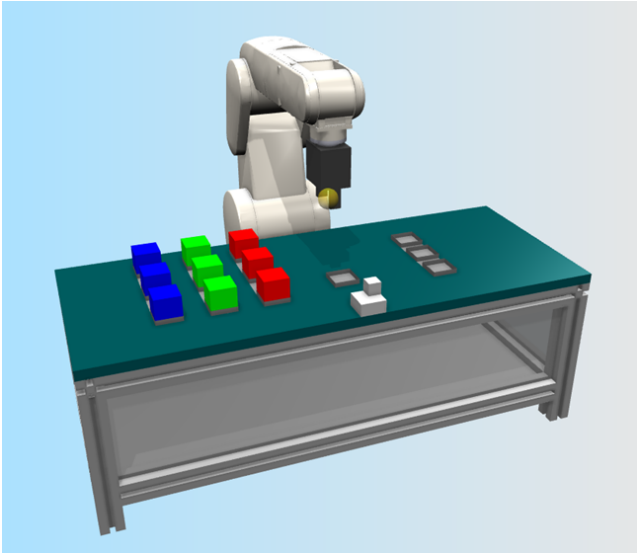


4. Gå tilbake til posisjonslisten og programmeringsvinduet ditt. Flytt boksene tilbake til startposisjon i samme kode
5. Bygg programmet og se at alle boksene flyttes til plasseringsposisjonene og returneres til startposisjonene.
6. Lag en loop som gjør at programmet kjører til det blir stoppet manuelt.

C.3 Lab 3 - Sortering av bokser med fargesensor

Lab 3

Oppgave: Sortering av bokser med fargesensor



Formål:

- Forstå bruken av referanserammer
- Lære å lage et program med løkker
- Ta i bruk I/O panel

Gjennomføring:

- Last ned og åpne Lab_3_StudentVersjon
- Lage posisjonsliste
- Opprett kode som gjennomfører ønsker resultat med if-setning og for-løkke.
- Bruke I/O panelet for indeksering
- Du skal levere besvarelsen som èn zip-fil som inneholder både Lab_3_StudentVersjon.pos og Lab_3_StudentVersjon.mb4 (dette er filene for posisjonslisten og programkoden).

Utstyr:

- PC med Ciros Education 6.4
- Ivanti Secure Access Client

Innføringsinfo:

I denne laboppgaven skal vi bruke modulen som er satt opp til å sortere bokser etter farge.

<u>Oppgave 1 – oppsett av grip points, posisjoner og koblinger til I/O</u>

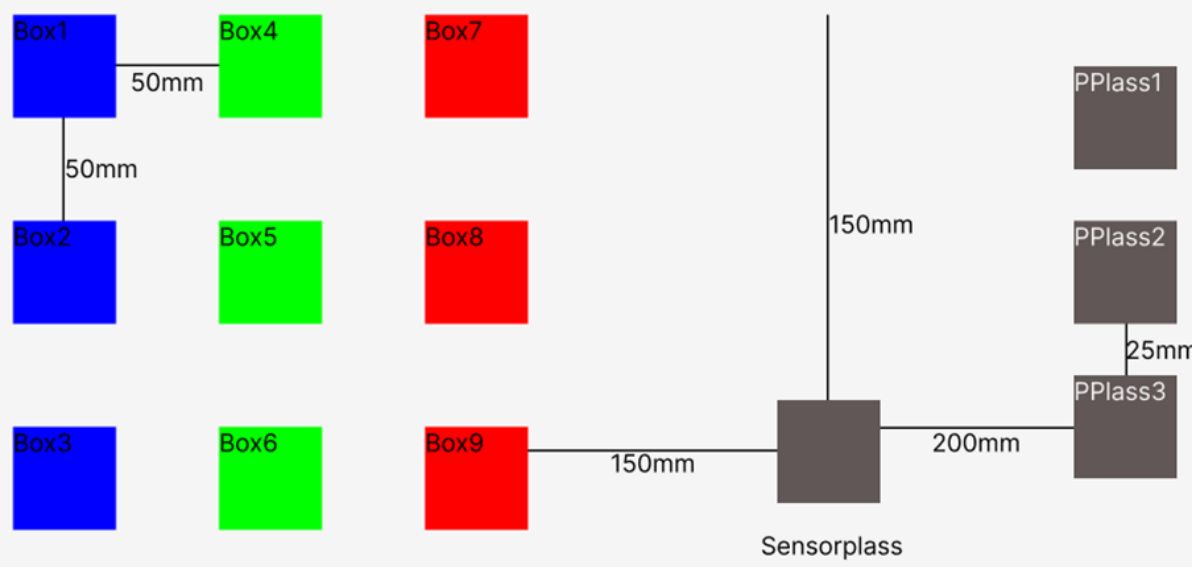
Fullfør posisjonslisten. Følgenede skal være med:

- StartPos.
- PlukkeBox1 til og med PlukkeBox9.
- Sensorplass.
- PallPlass1 til og med Pallplass3.

TIPS:

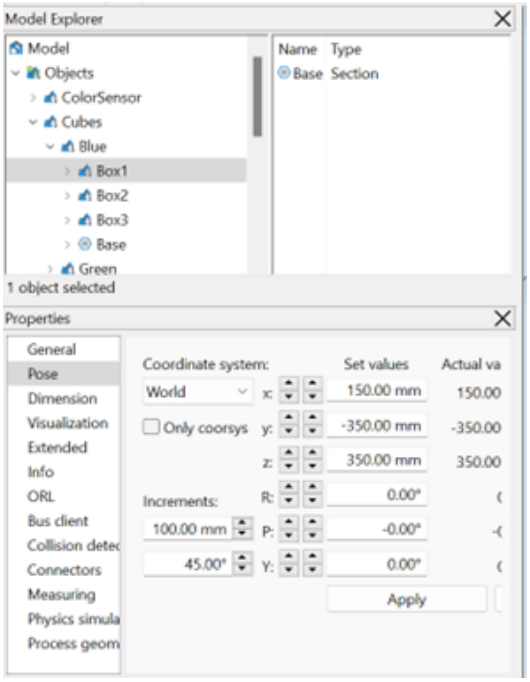
Bruk målene fra figur 1 og posisjonene som allerede er satt inn til å finne frem de resterende posisjonene. Boksene er 50*50 mm.

Figur 1.



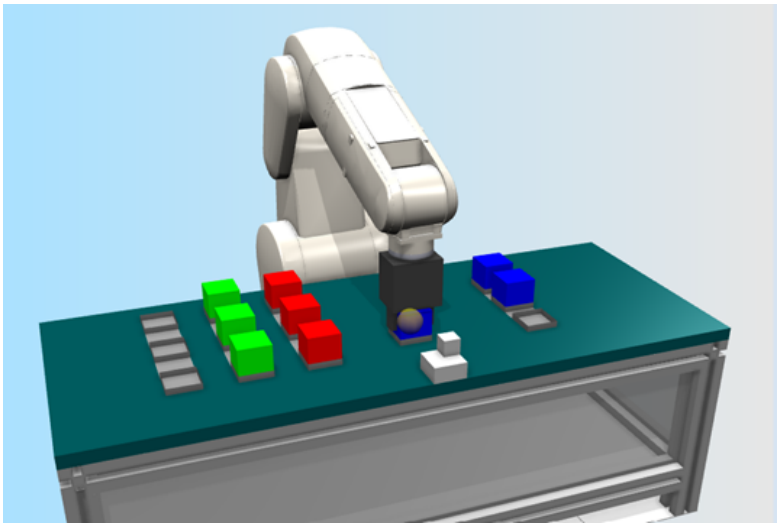
Eventuelt gå til MODELING→Model Explorer→Objects→Cubes→Blue→Høyreklikk Box1→Properties→Pose. Her finner du plasseringen til Box1. Se på sammenhengen mellom disse verdiene og verdiene som er satt i posisjonslisten PlukkeBox1.

P2	175.0,-325.0,375.0	-180,-0,-180,R,A,N	PlukkeBox1



<u>Oppgave 2 – Bruk av if-setning</u>

Lag et program som flytter de blå boksene via sensorplassen for så å plassere de på pallplassen hvis boksene er blå. Hvis de ikke er blå blir de plassert tilbake. Denne koden skal inneholde en if-setning.



IF-setning:

IF (betingelse) THEN	Hvis betingelse = sann à kjører kode under
Kode…	
ELSE	Hvis ikke betingelsen = sann à kjører denne koden
Kode…	
ENDIF	Ferdig

TIPS:

Gå til I/O panelet for å finne hvilken input index dere skal bruke: MODELING→I/O panel. Her ser dere at indexnummeret er 6,7 og 8.

For å kalle på en input skriver man: M_IN(indexnummer). Da skriver du eksempelvis (Verdier i denne koden er **IKKE** nødvendigvis riktig):

```
100 IF M_IN(3) = 1 THEN
110 //kode some flytter boks til PallPlass
120 ELSE
130 //Kode som flytter boks tilbake til sin orginale plass
140 ENDIF
```

Oppgave 3 – Bruk av for-løkke

Gjør koden fra oppgave 2 kortere ved hjelp av en FOR-løkke.

FOR-løkke:

DEF INTE PosX Du definerer en variabel som jeg kaller PosX

FOR PosX = 1 TO 7

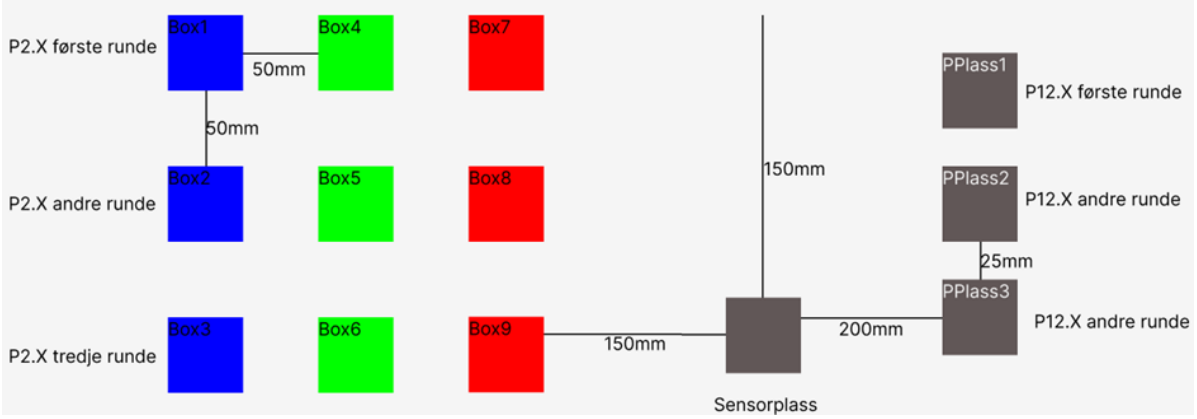
Kode..

NEXT PosX Kjører koden fra topp til bunn 7 ganger

TIPS:

Du må oppdatere posisjoner etter kode, men før NEXT Variabel. Eksempelvis kode(Verdier i denne koden er **IKKE** nødvendigvis riktig):

```
100 DEF INTE PosX                   //Definerer en variabel
110 FOR PosX = 1 TO 3               //Velger hvor mange runder koden kjører
120 IF M_IN(3) = 1 THEN             //Velger betingelse
130 //kode some flytter boks til PallPlass
140 ELSE
150 //Kode som flytter boks tilbake til sin orginale plass
160 ENDIF
170 MOV P1                          //Gå til StartPos etter hver runde
180 P2.X = P2.X + 50                //Oppdaterer x-verdiene
190 P12.X = P12.X + 50              //Oppdaterer slik at vi plasserer riktig
200 NEXT PosX                       Neste runde
```

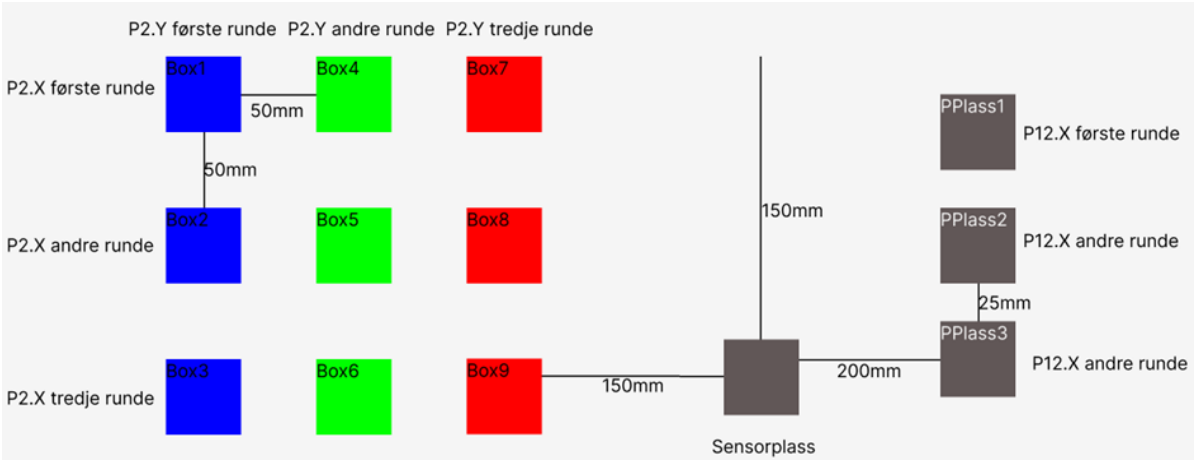


Oppgave 4 – forbedre programmet

Utvid programmet igjen slik at du kan flytte på de boksene du vil ved å endre index nummeret. Programmet fungerer etter sin hensikt når du klarer å flytte enten en blå, grønn eller rød boks til pallplass ved å endre index nummeret i koden.

TIPS:

Definer en ny variabel for PosY og lag enda en FOR-løkke utenfor hele koden. Husk å oppdatere posisjoner før neste runde.



Eksempelvis kode(Verdier i denne koden er **IKKE** nødvendigvis riktig):

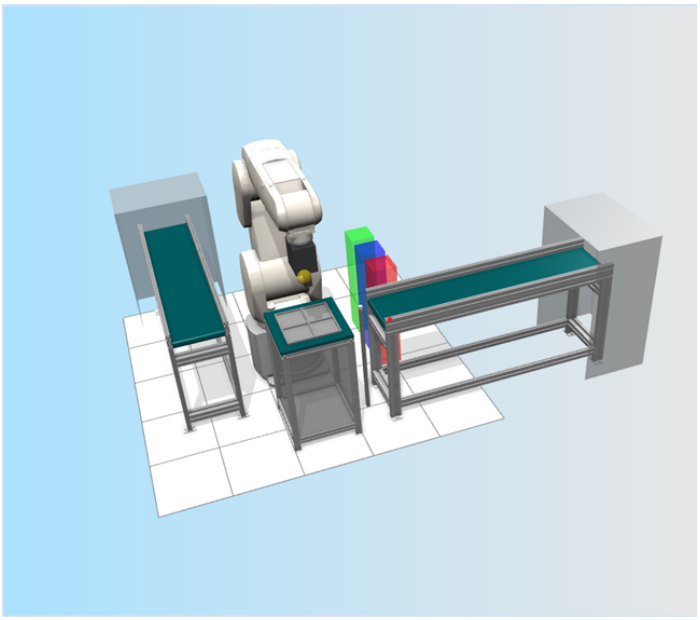
```
100 DEF INTE POSY                  //Definerer en variabel
110 DEF INTE PosX                  //Definerer en variabel
120 FOR PosY = 1 TO 3              //Velger hvor mange runder koden kjører
130 FOR PosX = 1 TO 3              //Velger hvor mange runder koden kjører
140 IF M_IN(3) = 1 THEN            //Velger betingelse
150 kode some flytter boks til PallPlass
160 ELSE
170 Kode som flytter boks tilbake til sin orginale plass
180 ENDIF
190 MOV P1                         //Gå til StartPos etter hver runde
200 P2.X = P2.X + 50               //Oppdaterer x-verdiene
210 P12.X = P12.X + 50             //Oppdaterer slik at vi plasserer riktig
220 NEXT PosX                      //Neste runde
230 P2.X = 100                     //Oppdaterer x-verdi
240 P2.Y = P2.Y + 50               //Oppdaterer Y-verdi
250 P12.X = 150                    //Oppdaterer pallplassering
260 Next PosY
```

Fagskulen Vestland
Vestland fylkeskommune

C.4 Lab 4 - Produksjonsbånd, sorter og send videre

Lab 4

Oppgave: Produksjonsbånd, sorter og send videre



Formål:

- Finne posisjoner ved bruk av referanserammer
- Bruke programmeringsmetoder for å oppnå et fabrikkautomatisk system
- Ta i bruk I/O panel

Gjennomføring:

- Last ned og åpne Lab_4_StudentVersjon
- Lage posisjonsliste
- Opprett kode som gjennomfører ønsker resultat med hjelp av forskjellige programmeringsmetoder.
- Bruke I/O panelet for indeksering
- Du skal levere besvarelsen som èn zip-fil som inneholder både Lab_4_StudentVersjon.pos og Lab_4_StudentVersjon.mb4 (dette er filene for posisjonslisten og programkoden).

Utstyr:

- PC med Ciros Education 6.4
- Ivanti Secure Access Client

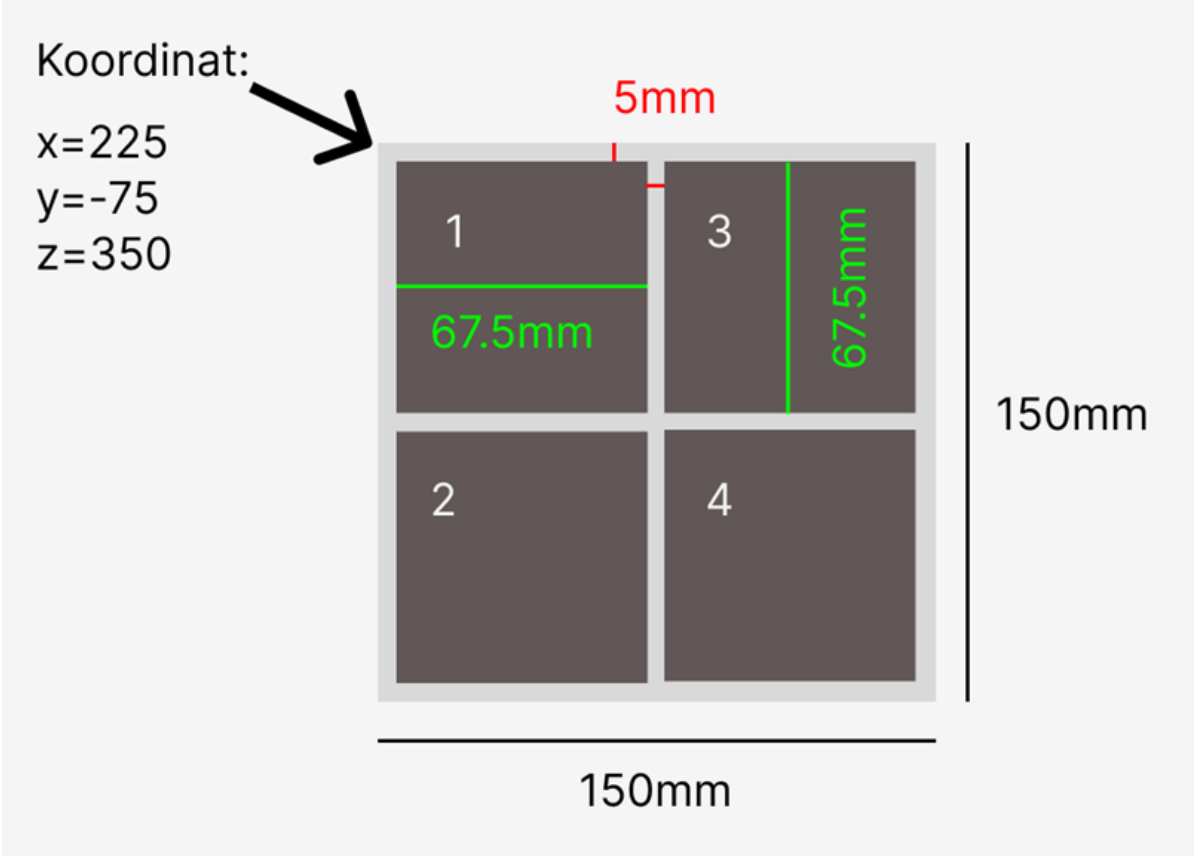
Innføringsinfo:

I denne laboppgaven skal vi bruke modulen som er satt opp til å sortere bokser etter farge.

Oppgave 1 – Finn posisjon ved bruk av referanseramme

I denne oppgaven skal du fylle inn posisjonene PallPlass1, PallPlass2, PallPlass3 og PallPlass4.

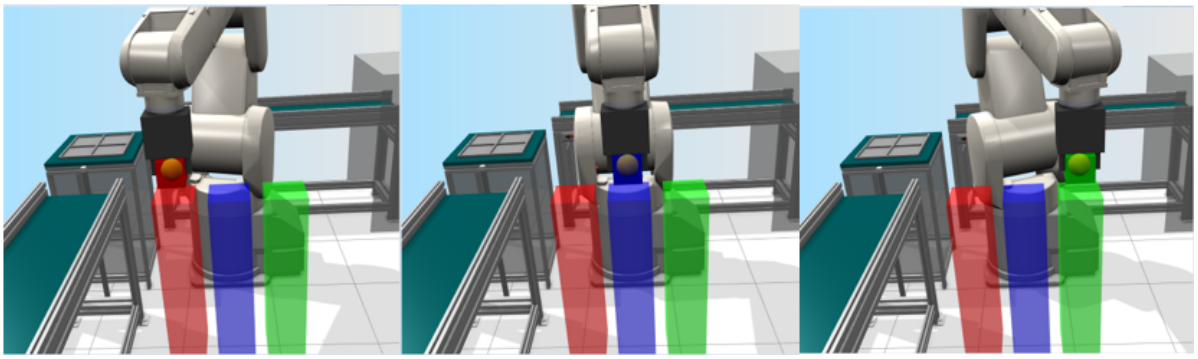
Pallen er plassert world kordinatet som vist under. Punktene skal være sentrert innenfor sin bestemte ramme med høyde 25mm over pallens z-koordinat.



Oppgave 2 – Lag programkode

Fullfør programkoden slik at de fargede boksene blir plukket opp fra P2(PlukkeBox) og kastet riktig sted. Her er det viktig at dere gjør dere kjent med indekseringen av de forskjellige digitale inputs/outputs. Programmet skal sortere uendelig.

I koden bruker vi en while-løkke. Den fungerer slik at så lenge kravet som er satt er sant utfører den koden inne i while-løkken.



Her ser dere relevante inputs og outputs for systemet. Dere kan også trykke på MODELLING→Model Explorer→Objects→RV-4FL→Inputs/Outputs for å få oversikten.

Input	Index	Type	Value	Connected output
STOP	000	Digital	0	-
SRVOFF	001	Digital	0	-
ERRRESET	002	Digital	0	-
START	003	Digital	0	-
SRVON	004	Digital	0	-
IOENA	005	Digital	0	-
ItemAvailable	006	Digital	0	PartAtEnd
IsRed	007	Digital	0	DetectRed
IsBlue	008	Digital	0	DetectYellow
IsGreen	009	Digital	0	DetectGreen

Output	Index	Type	Value	Connected inputs
[System]	-001	Digital	0	0
START	000	Digital	0	0
SRVON	001	Digital	0	0
ERRRESET	002	Digital	0	0
IOENA	003	Digital	0	0
Generate	004	Digital	0	1
StartStop_INN	005	Digital	0	1
RemoveRed	006	Digital	0	1
RemoveBlue	007	Digital	0	1
RemoveGreen	008	Digital	0	1

Inputs som brukes:
ItemAvailable: Når verdien = 1 er boksen på enden av transportbåndet.
IsRed: Når verdien = 1 har fargesensoren detektert en rød boks.
IsBlue: Når verdien = 1 har fargesensoren detektert en blå boks.
IsGreen: Når verdien = 1 har fargesensoren detektert en grønn boks.

Outputs som brukes:
Generate: Når verdien = 1 genereres en tilfeldig farget boks.
StartStop_INN: Når verdien = 1 beveger transportbåndet seg fremover.
RemoveRed: Når verdien = 1 fjerner rødt boss en boks(hvis boksen er plassert der).
RemoveBlue: Når verdien = 1 fjerner blått boss en boks(hvis boksen er plassert der).
RemoveGreen: Når verdien = 1 fjerner grønt boss en boks(hvis boksen er plassert der)

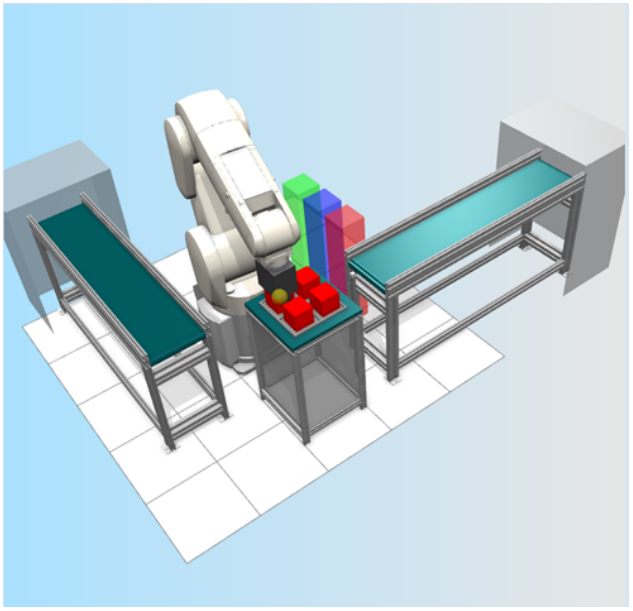
Oppgave 3 – Flytte røde bokser på pallen

Utvid programmet slik at vi sorterer de røde boksene på pallen. En måte å gjøre dette på er å bruke flere if-setninger(du kan bruke andre metoder hvis du vil).

Eksempel:

```
220 IF M_IN(7) = 1 THEN     //Red Box
230 MVS P2, -50
240 ANTALL = ANTALL + 1
250 IF ANTALL = 1 THEN
260 MOV P7, -50
270 MVS P7
280 HOPEN 1
290 DLY 0.5
300 ENDIF
310 IF ANTALL = 2 THEN
320 MOV P8, -50
```

Osv.



Oppgave 4 – Fullfør programmet, send boksene videre

Fullfør programmet ditt. Du skal nå utvide programmet ditt slik at du sender boksene videre når du har plassert fire røde bokser på pallen. En måte å gjøre det på er ved hjelp av for-løkke etter when-løkken. Til slutt kan du bruke kommando GOTO (ønsket linje) for å starte prosessen på nytt.

Husk at du må sette på transportbåndet.

Input	Index	Type	Value	Connected output
STOP	000	Digital	0	-
SRVOFF	001	Digital	0	-
ERRRESET	002	Digital	0	-
START	003	Digital	0	-
SRVON	004	Digital	0	-
IOENA	005	Digital	0	-
ItemAvailable	006	Digital	0	PartAtEnd
IsRed	007	Digital	0	DetectRed
IsBlue	008	Digital	0	DetectYellow
IsGreen	009	Digital	0	DetectGreen
ItemSend	010	Digital	0	PartAtEnd

Output	Index	Type	Value	Connected inputs
[System]	-001	Digital	0	0
START	000	Digital	0	0
SRVON	001	Digital	0	0
ERRRESET	002	Digital	0	0
IOENA	003	Digital	0	0
Generate	004	Digital	0	1
StartStop_INN	005	Digital	0	1
RemoveRed	006	Digital	0	1
RemoveBlue	007	Digital	0	1
RemoveGreen	008	Digital	0	1
SendVidere	009	Digital	0	1
StartStop_UT	010	Digital	1	1

Inputs som brukes:

StartStop_UT: når verdien = 1 beveger transportbåndet seg fremover.

SendVidere: når verdien = 1 fjernes boksen fra simuleringen.

ItemSend: når verdien = 1 er boksen på enden av transportbåndet.

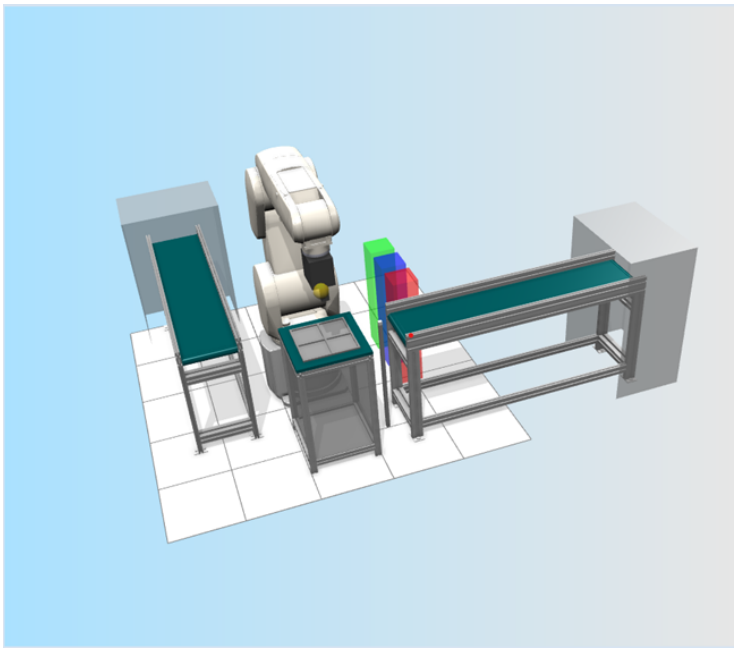
NB: FEIL I MODUL(ikke i din kode). Boksene skal egentlig forsvinne i enden av transportbåndet. Det er da ikke nødvendig å bruke output SendVidere(M_OUT(9)).

Fagskulen Vestland
Vestland fylkeskommune

C.5 Lab 4 - Tolk og forklar programmet

Lab 4

Oppgave: Produksjonsbånd, sorter og send videre



Formål:

- Forstå hvordan programspråket Melfa Basic IV fungerer i Ciros.
- Skal kunne forstå bruken av inputs og outputs.

Gjennomføring:

- Last ned og åpne Lab_4_FerdigVersjon.
- Kommenter hvordan programkoden fungerer.
- Forklar hva de forskjellige inputs og outputs gjør.
- Du skal levere denne filen som en pdf.

Utstyr:

- PC med Ciros Education 6.4.
- Ivanti Secure Access Client.

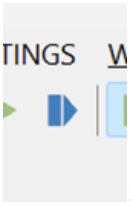
Innføringsinfo:

I denne laboppgaven skal vi ta i bruk en komplett modul med programkode. Du skal kommentere på hva koden gjør i hver linje. Som hjelpemiddel kan du kjøre programmet i sakte modus for å oppnå bedre forståelse.

Oppgave 1 –

Tolk og forklar programkoden

Du skal her skrive en kort kommentar på alle grønne linjer. I figuren under kan du se hva de forskjellige indekseringene er. Tips kjør programmet i step forward modus hvis du er usikker.



Step forward knapp.

Input	Index	Type	Value	Connected output
STOP	000	Digital	0	-
SRVOFF	001	Digital	0	-
ERRRESET	002	Digital	0	-
START	003	Digital	0	-
SRVON	004	Digital	0	-
IOENA	005	Digital	0	-
ItemAvailable	006	Digital	0	PartAtEnd
IsRed	007	Digital	0	DetectRed
IsBlue	008	Digital	0	DetectYellow
IsGreen	009	Digital	0	DetectGreen
ItemSend	010	Digital	0	PartAtEnd

Output	Index	Type	Value	Connected inputs
[System]	-001	Digital	0	0
START	000	Digital	0	0
SRVON	001	Digital	0	0
ERRRESET	002	Digital	0	0
IOENA	003	Digital	0	0
Generate	004	Digital	0	1
StartStop_INN	005	Digital	0	1
RemoveRed	006	Digital	0	1
RemoveBlue	007	Digital	0	1
RemoveGreen	008	Digital	0	1
SendVidere	009	Digital	0	1
StartStop_UT	010	Digital	1	1

```
100 DEF INTE ANTALL          //EKS:Her definerer jeg en variabel som jeg kaller ANTALL

110 ANTALL = 0               //

120 WHILE ANTALL < 4         //

130 M_OUT(4) = 1             //

140 M_OUT(4) = 0             //

150 M_OUT(5) = 1             //

160 WAIT M_IN(6) = 1         //

170 M_OUT(5) = 0             //

180 MOV P2, -50

190 MVS P2

200 HCLOSE 1
```

```
210 DLY 0.5


220 IF M_IN(7) = 1 THEN          //
230 MVS P2, -50
240 ANTALL = ANTALL + 1          //
250 IF ANTALL = 1 THEN           //
260 MOV P7, -50
270 MVS P7
280 HOPEN 1
290 DLY 0.5
300 ENDIF                        //
310 IF ANTALL = 2 THEN           //
320 MOV P8, -50
330 MVS P8
340 HOPEN 1
350 DLY 0.5
360 ENDIF                        //


370 IF ANTALL = 3 THEN           //
380 MOV P9, -50
390 MVS P9
400 HOPEN 1
410 DLY 0.5
420 ENDIF                        //
430 IF ANTALL = 4 THEN           //
```

```
440 MOV P10, -50

450 MVS P10

460 HOPEN 1

470 DLY 0.5

480 ENDIF                    //


500 IF M_IN(8) = 1 THEN      //

510 MVS P2, -50

520 MOV P4, -50

530 MVS P4

540 HOPEN 1

550 M_OUT(7) = 1             //

560 M_OUT(7) = 0             //

570 DLY 0.5


580 ELSE                     //

590 MVS P2, -50

600 MOV P5, -50

610 MVS P5

620 HOPEN 1

630 M_OUT(8) = 1             //

640 M_OUT(8) = 0             //

650 DLY 0.5

660 ENDIF                    //

670 ENDIF                    //
```

```
680 MOV P1                  //
690 WEND                    //

700 FOR ANTALL = 1 TO 2     //
710 M_OUT(10) = 1           //
720 MOV P7, -50
730 MVS P7
740 HCLOSE 1
750 DLY 0.5
760 MVS P7, -50
770 MOV P6, -50
780 MVS P6
790 HOPEN 1
800 DLY 0.5
810 MVS P6, -50
820 M_OUT(9) = 1            //
830 WAIT M_IN(10) = 1       //
840 MOV P1
850 P7.X = P7.X + 72.5      //
860 NEXT ANTALL             //
870 P7.X = P7.X – 145       //

880 FOR ANTALL = 1 TO 2     //
890 M_OUT(10) = 1           //
```

```
900 MOV P9, -50

910 MVS P9

920 HCLOSE 1

930 DLY 0.5

940 MVS P9, -50

950 MOV P6, -50

960 MVS P6

970 HOPEN 1

980 DLY 0.5

990 MVS P6, -50

1000 M_OUT(9) = 1              //

1010 WAIT M_IN(10) = 1         //

1020 MOV P1

1030 P9.X = P9.X + 72.5        //

1040 NEXT ANTALL               //

1050 P9.X = P9.X – 145         //


1060 GOTO 110                  //

1070 End                       //
```

NB: FEIL I MODUL(ikke i din kode). Boksene skal egentlig forsvinne i enden av transportbåndet. Det er da ikke nødvendig å bruke output SendVidere(M_OUT(9)).

Appendiks D Studentundersøkelse ved test av lab

D.1 - Student nr. 1

1. Hvordan syns du oppgaveformuleringen var?
   - ○ Dårlig
   - ○ Litt bra
   - ◉ Bra
   - ○ Veldig bra

2. I hvilken grad klarte du å løse oppgaven?
   - ○ 25%
   - ○ 50%
   - ○ 75%
   - ◉ 100%

3. Hvordan var det å finne frem i Ciros?
   - ◉ Enkelt
   - ○ Vanskelig
   - ○ Varierte

4. Hva synes du om nivået på oppgaven?
   - ○ For enkelt
   - ◉ Helt passelig
   - ○ For vanskelig

5. Hva synes du var bra i oppgaven?

   Bra layout på oppgavetekst. Passe stor oppgave.

6. Har synes du var negativt i oppgaven?

   Dette var første gang jeg programmerte denne roboten. så var ukjent med syntaksen, Kunne kanskje vært noe info om syntaks for løkker. Blir også ganske langt og repeterende program. kanskje det hadde holdt med for eksempel 6 klosser? Kunne og vært en ide å inkludere noen tips om linjenummering.

7. Andre tilbakemeldinger?
   Første punkt under gjennomføring sier vi skal laste ned Lab1_oppgaveversjon. riktige skal vel være Lab2_oppgaveversjon.

D.2 - Student nr. 2

1. Hvordan syns du oppgaveformuleringen var?

- ○ Dårlig
- ◉ Litt bra
- ○ Bra
- ○ Veldig bra

2. I hvilken grad klarte du å løse oppgaven?

- ○ 25%
- ○ 50%
- ○ 75%
- ◉ 100%

3. Hvordan var det å finne frem i Ciros?

- ◉ Enkelt
- ○ Vanskelig
- ○ Varierte

4. Hva synes du om nivået på oppgaven?

- ◉ For enkelt
- ○ Helt passelig
- ○ For vanskelig

5. Hva synes du var bra i oppgaven?

   Helt grei innføring av hvordan aksesystemet virker

6. Har synes du var negativt i oppgaven?

   Oppgaven starter med at man skal åpne LAB_1. Ble lurt der gitt, det skulle man ikke. Burde muligens stått flere hint om hvordan programmet bør være. jeg lagde 180 linjer, kan det skrives smartere/korter?

7. Andre tilbakemeldinger?

   Kanskje litt vel enkel oppgave. hovedsakelig dreide den seg om å finne posisjonene til kubene noe som var lett når mange posisjoner lå inne. Kanskje det hadde vært stilig å bygge videre på koden med flere funksjoner som en bonusoppgave.

D.3 - Student nr. 3

1. Hvordan syns du oppgaveformuleringen var?
   - ○ Dårlig
   - ◉ Litt bra
   - ○ Bra
   - ○ Veldig bra

2. I hvilken grad klarte du å løse oppgaven?
   - ○ 25%
   - ○ 50%
   - ○ 75%
   - ◉ 100%

3. Hvordan var det å finne frem i Ciros?
   - ◉ Enkelt
   - ○ Vanskelig
   - ○ Varierte

4. Hva synes du om nivået på oppgaven?
   - ○ For enkelt
   - ◉ Helt passelig
   - ○ For vanskelig

5. Hva synes du var bra i oppgaven?

Tipsene, bilder, veiledning til å finne fram posisjonsliste og programeringsviduet.

6. Har synes du var negativt i oppgaven?

Oppgåvetekst var litt rotete og ustrukturert. Bedre skille på oppgåvetekst og tips tekst osv. Litt overkill med 9 klosser. Veldig masse copy paste. Kanskje flytta mindre antall klosser til forskjellige plasser over fleire hinder?

7. Andre tilbakemeldinger?

Feil i oppgåveteksten. Står Lab1_oppgaveversjon og ikkje Lab2_Oppgaveversjon. Kanskje hatt med kva funksjon som er tenkt brukt for oppgåva. Så kan studenten heller lesa seg opp på syntaksen/virkemåten.

Appendiks E Andre filer

Vedlagt ligger zippet mappe med følgende filer:

Laboratoriesett:

Lab1: Innføring i robot bevegelser

- Lab1_studentversjon.zip
- Lab1_fullversjon.zip
- Lab1_Oppgavetekst.docx
- Lab1_Løsningsforslag.docx
- Lab1_Modellering.ptt

Lab2: Boks over hinder

- Lab2_studentversjon.zip
- Lab2_fullversjon.zip
- Lab2_Oppgavetekst.docx
- Lab2_Løsningsforslag.docx
- Lab2_Modellering.ppt
- Digitalt løsningsforslag: https://www.youtube.com/watch?v=gEXBLGYrq9w

Lab3: Sortering av bokser med fargesensor

- Lab_3_StudenVersjon.zip
- Lab_3_FerdigVersjon.zip
- Lab_3.docx
- Lab_3_Løsningsforslag.docx
- Lab_3_Modellering.docx

Lab4: Produksjonsbånd, sorter og send videre

- Lab_4_StudentVersjon.zip
- Lab_4_FerdigVersjon.zip
- Lab_4_docx
- Lab_4_Løsningsforslag.docx
- Lab_4_Modellering.docs
- Lab_4_Tolk og forklar programmet.docx
- Lab_4_Tolk og forklar programmet_Løsningsforslag.docx