Universitetet i Stavanger

DET TEKNISK-NATURVITENSKAPELIGE FAKULTET

BACHELOROPPGAVE

Studieprogram/spesialisering: Maskiningeniør.	Vårsemesteret, 2022 Åpen / Konfidensiell
Forfattere: Ruben Edland Narve Gilje Nordås	Ruben Edland (signatur forfatter) Narve Gilje Nordås (signatur forfatter)
Fagansvarlig: Professor Hirpa Gelgele Lemu, UiS. Veileder(e): Senioringeniør Yaaseen Ahmad Amith, UiS.	
Tittel på bacheloroppgaven: "Design og konstruksjon av vaffel-lagende robot." Engelsk tittel: "Design and construction of a waffle cooking robot."	
Studiepoeng: 20	
Emneord: Seksakset vaffel-lagende robotarm	Sidetall: **97** + vedlegg/annet: **173** Stavanger, 15.05.2022.

Forord

Formålet med denne bacheloroppgaven var å designe og konstruere en robotarm som skulle automatisere deler av prosessen rundt vaffelstekingen, som stort sett foregår hver fredag på personalrommet for det teknisk-naturvitenskapelige fakultet. Vi er to maskiningeniør-studenter ved Universitetet i Stavanger som søkte på denne bacheloroppgaven fordi den virket svært interessant, og en oppgave hvor en kunne ta i bruk de teoretiske kunnskapene ved å faktisk produsere/konstruere en fysisk maskin/robotarm. Faglig ansvarlig, professor Hirpa G. Lemu, informerte om at en robotarm som automatiserer deler av prosessen rundt vaffelstekingen har vært ønsket i flere år. Motivasjonen for å lykkes var derfor svært stor, da en slik mekanisme hadde vært etterlyst såpass lenge. Etter samtale med veileder fant man ut at hoved-oppgaven til robotarmen skulle være å designe og konstruere en robot som kan påføre vaffelrøre på minimum 2 vaffeljern, med plass til to vafler hver. Alle kostnader ved design og konstruksjon av robotarmen ble finansiert av universitet selv, ved det teknisk-naturvitenskapelige fakultet. Konstruksjonen av selve robotarmen foregikk på universitetsverkstedet. Det har vært en utfordrende, spennende og lærerik oppgave både teoretisk og praktisk. Det er først **etter** arbeidet med bacheloroppgaven en føler en kan gå ut i arbeidslivet, med tittelen "maskiningeniør", og tilføre et eventuelt selskap en nytteverdi.

Vi vil rette en takk til senioringeniør Yaaseen Ahmad Amith som har ordnet med bestilling av eksterne deler, tilrettelegging, og for å ha gitt oss oppmuntring med tanke på at roboten vil utføre dens tiltenkte arbeidsoppgave.
Vi vil også takke de ansatte på verkstedet ved Universitetet i Stavanger for god opplæring på ulike maskiner, råd, og HMS. De har alltid vært kjappe med å hjelpe hvis det trengs.
Ellers så må vi takke professor Hirpa Gelgele Lemu for å la oss utføre denne oppgaven som vi søkte på, hvor han da har troen på at vi klarer en slik oppgave. Samt for viktige og konstruktive tilbakemeldinger på oppgaven, i forbindelse med fysiske møter og e-post-korrespondanse.

Stavanger, 15.05.2022

Narve Gilje Nordås: Narve Gilje Nordås

Ruben Edland: Ruben Edland

Sammendrag

Bacheloroppgaven gikk ut på å fysisk konstruere en robotarm som skulle påføre røre på to vaffeljern, med plass til to vafler hver. Dette skulle kunne gjentas til det ikke var mer vaffelrøre igjen. Deretter skulle en vaskeprosess kunne igangsettes, for å rengjøre slanger, pumpe, tank og ventil. Man endte opp med et design der det var koblet en slange fra en 10l-gryte (som fungerte som tank for vaffelrøren) til en trykkvannspumpe. Pumpen pumpet så røren opp til ventilen, som var festet til tuppen av robotarmen. En ventil som var normalt lukket. Ventilen åpnet seg et gitt tidsintervall når robotarmen hadde nådd posisjonene hvor vaffelrøre skulle påføres. Slik fortsatte den til det hadde blitt påført røre ved alle de fire posisjonene. Deretter stoppet den opp ved utgangsposisjonen, og ventet på et signal startknappen for å kunne gjennomføre prosessen på nytt, helt til det var tomt for vaffelrøre. En oppgave som ville kreve at kunnskaper fra de fleste obligatoriske –og ikke obligatoriske programfag måtte tas i bruk. Spesielt obligatoriske fag som maskinkonstruksjon og 3D-modellering, samt valgfaget styringsteknikk. En var på forhånd også klar over at kunnskap som strakk seg utover pensumet måtte tilegnes. Med andre ord en spennende oppgave, men også utfordrende og krevende. Det har vært et gjennomgående fokus under utformingen av robotarmen at de totale kostnadene skulle holdes så lave som mulig, og nyinnkjøpte deler skulle være så få som mulig. Dette fordi det er universitet/fakultet som tok regningen, og styrte et budsjett, men også fordi gjenbruk og miljø var viktige stikkord. Målet var å få så lave kostnader at dette vil være et mer prisgunstig alternativ for fakultet enn å kjøpe en industriell robotarm og programmere denne til å utføre oppgavene. Dette innebar blant annet at samtlige bære-deler, bortsett fra kulelager, bolter og muttere, ble laget selv, av materialer fra "skraphaugen" på universitets-verkstedet. Ved kjøp av eksterne deler, som motorer, ledninger, ventiler m.m., har også lav pris vært et viktig stikkord. På grunn av dette så ble ikke det ytre designet det mest estetiske. Funksjon har trumfet estetikk i alle ledd. Det overordnede målet har til enhver tid vært at robotarmen skulle kunne utføre dens oppgaver. En konkluderte til slutt med at robotarmen klarte å gjennomføre de oppgavene den var tiltenkt. Den oppnådde ønskede posisjoner, og rett mengde røre ble tilført vaffeljernet ved disse posisjonene. Man var likevel ikke helt tilfreds med brukersikkerheten og brukeropplevelsen knyttet til håndteringen av produktet. Så man bestemte seg for, foreløpig, at robotarmen kun skulle tas fram ved spesielle anledninger, og kun tas i bruk av kvalifisert opplært personell. Man ønsker også at robotarmen er i stand til å gjennomføre flere oppgaver enn det den kan i dag, som f.eks. å løfte lokkene til vaffeljerna, og ta vafler av vaffeljerna.

Sammendrag (abstract)- engelsk

The bachelor-thesis main goal was to construct a waffle cooking robot arm, which could apply waffle-batter onto two different "waffle-irons", with room for two waffles each. This process should have the possibility to run, until there was no more batter left. After this, one should have the possibility to run a washing-program, where the pump, valve, pipes would be thoroughly cleaned. The final design ended up being a pipe, connected to a 10l pot, which was used as the tank for the waffle-batter. This pipe was so connected to a pump, which pumped the batter up to a solenoid valve, which was located at the end-effector of the robot arm. The valve vas NC, and opened at the right positions, for the right amount of time. It did this at all 4 positions. After one round, the robot arm would stop, and wait for another signal from an installed momentary pushbutton. One could rerun this process until the there was no more batter in the tank. This task implied that the authors of this thesis would have a wide spread of knowledge, including programming, CAD, mechanical design and so forth. Knowledge, which was already acquired, and knowledge which one had to acquire during the construction. In others words a demanding and challenging, but exciting task. It was always a main focus to keep the total costs as low as possible, because the university is paying for these costs. Environment was also a key word. Because of this one decided to construct all carrying-parts of already used material, from the "trash-pile" in the university-workshop. With the costs of external parts, as motors, drivers, power supplies, low price has also been important. The main goal with was to keep the costs so low, that this robot arm would be a cheaper alternative than buying an industrial robot arm. Because the costs were kept as low as possible, the final design was not the most aesthetic. Functionality was prioritized above aesthetics. The overall goal was to make a robot arm, which executed its predefined tasks. The robot arm did exactly this. It achieved its intended positions and applied the right amount waffle-batter at these positions. However, one was still not fully satisfied. To get an even better product, one would make sure user-safety and user-experience would get thoroughly improved. One would also see that the robot arm could perform more, and more complex tasks, as lifting the cover of the waffle-irons, and moving waffles.

Innholdsfortegnelse

Liste over figurer

Liste over tabeller

Liste over ligninger

1
Innledning

1.1 Oppbygging av oppgaven

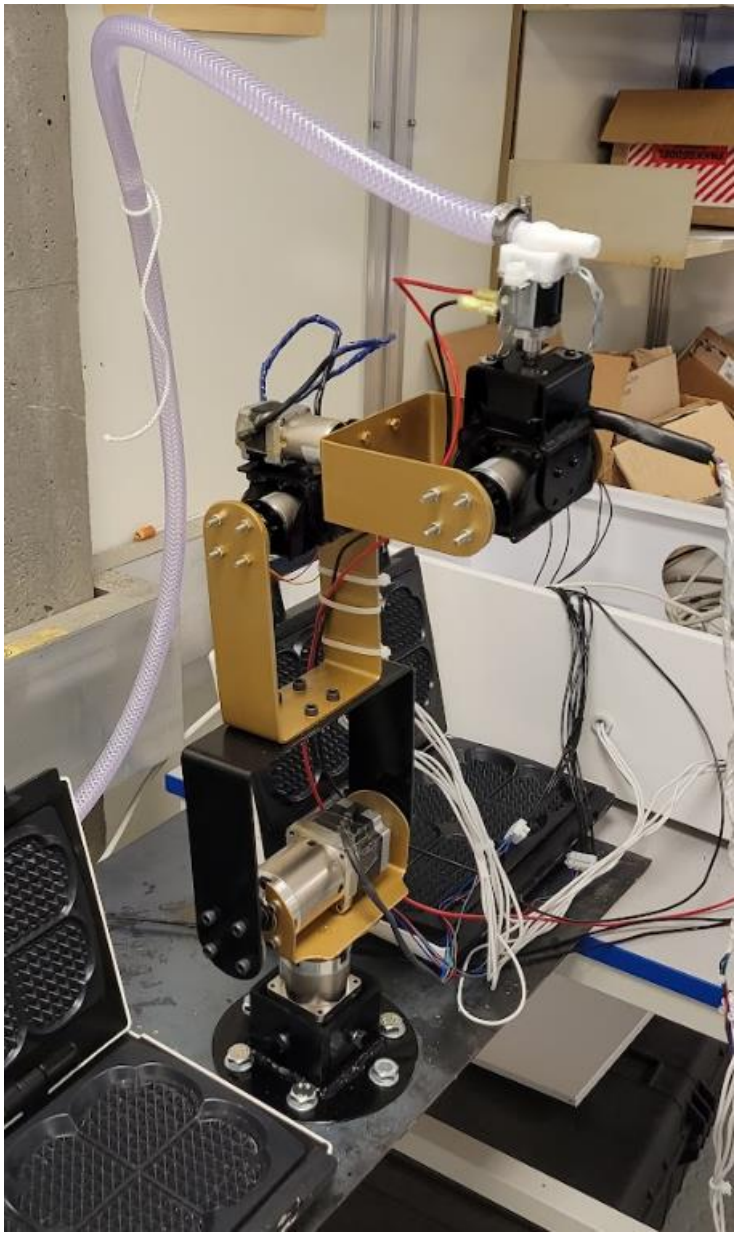
Under følger en liste over oppbyggingen av oppgaven i deler. Med en kort forklaring hva de ulike delene inneholder.

- Del 1, innledning. Innledningen består av forklaring rundt oppgavens problemstilling, en begrunnelse bak prosjektet, begrensinger, og robotarmens bruksområde.
- Del 2, teori og metode. Hvilke metoder som brukes for å oppnå ønskede. Teorien som er grunnlaget for de ulike metodene. Formler/likninger som blir brukt til å utforme resultatene. Her finner man blant annet lenker til styreprogramkode, dimensjoneringsberegninger, simuleringsverktøy m.m.
- Del 3, resultater. Her beskriver man resultater, som: -Fungerte roboten som den skulle? Ble ønskede posisjoner oppnådd? Var dimensjoneringen tilfredsstillende? Hvilke sikkerhetsfaktorer oppnådde man? Fungerte pumpesystemet som det skulle? Ble kostnadene så lave som ønsket m.m.
- Del 4, diskusjon. knyttet til resultatene, valgene vi har tatt, og hva vi kunne gjort annerledes. Om ønskede resultater ikke ble oppnådd, hvorfor var det slik? -Kan man se paralleller mellom teori/metode og ikke-oppnådde resultater? -Hva kan evt. forbedres, og hvordan? -Er resultatet, sett under ett, tilfredsstillende?
- Del 5, konklusjon. Det endelige resultatet, og kort om hva som evt. må forbedres.
- Avslutningsvis kommer en litteraturliste som viser våre kilder som vi har benyttet i denne oppgaven.
- Helt til slutt følger vedleggene, som f.eks. forstudierapporten, arbeidstegninger, utledninger, programvare for dimensjonering, styreprogram og diverse figurer og tabeller.

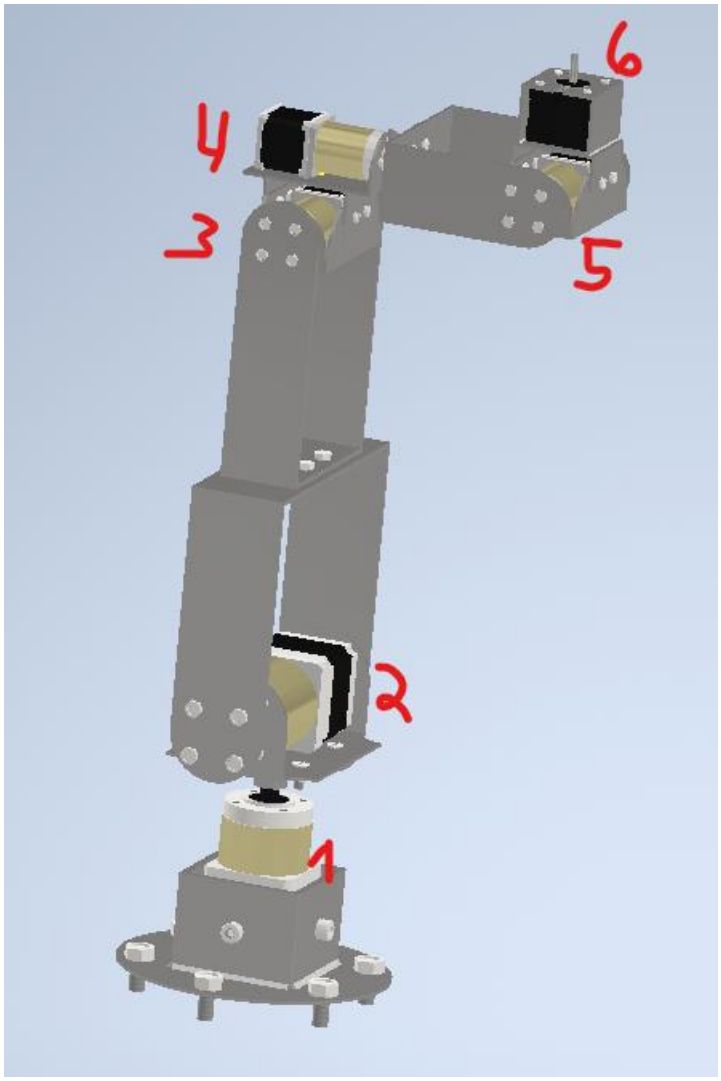
1.2 Bakgrunn

Hensikten med en slik oppgave i et studieprogram som maskiningeniør var å kunne vise kunnskap lært gjennom det treårige studiet. Faglig ansvarlig, Hirpa G. Lemu, informerte på første møte at vaffel-roboten lenge hadde vært ønsket av fakultet. Dette for å blant annet lette på arbeidsmengden i.fm. vaffelstekingen. Samme oppgaven ble blant annet gitt til en annen

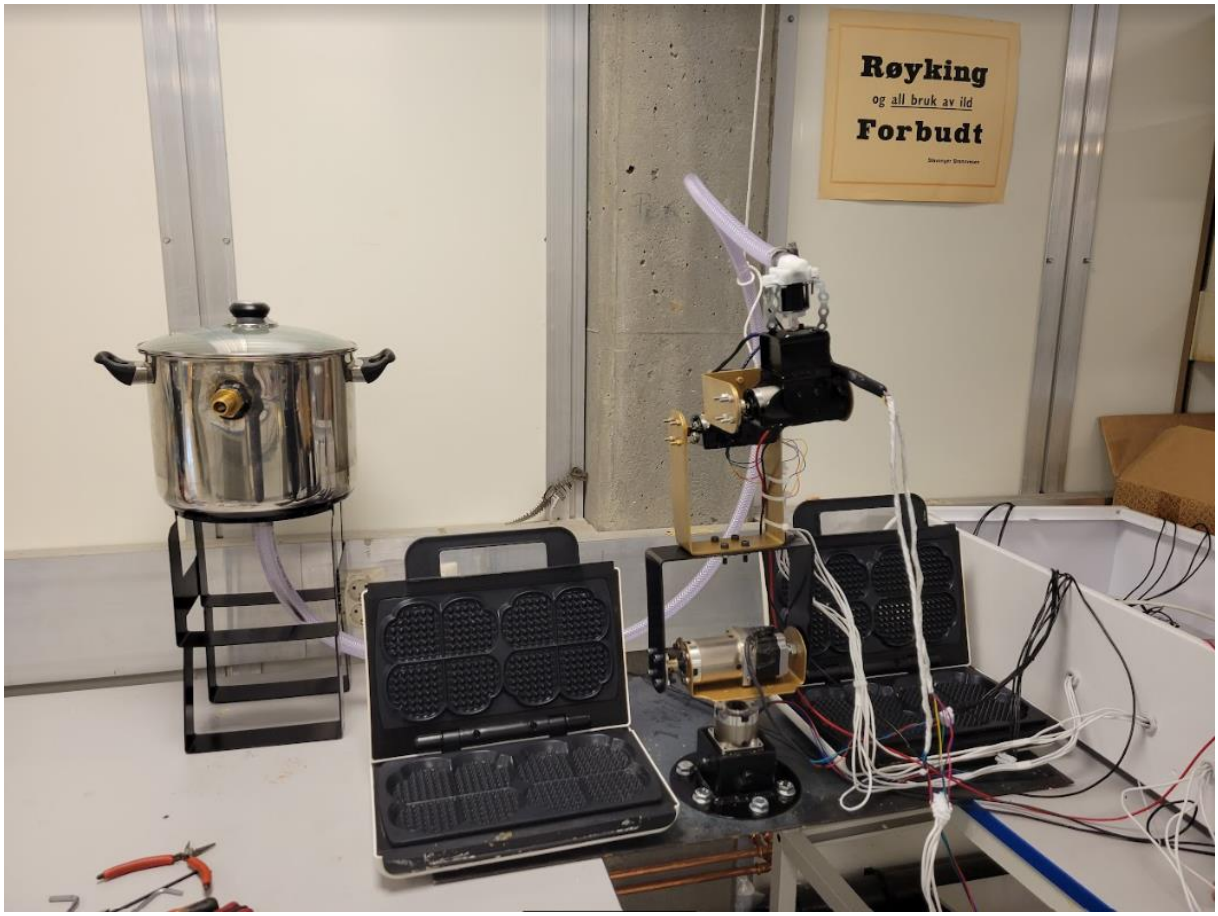
student våren 2021, men Lemu fortalte på samme møte, at roboten aldri ble montert på pauserommet. Med andre ord så er det en stor forventning fra hele fakultetet at det står klar en vaffelrobot i midten av mai 2022. Motivasjonen for å lykkes var derfor stor, samt ansvaret ved denne oppgaven. Det kan også nevnes at det fra starten av var planen at denne oppgaven skulle utføres i samarbeid med en student fra elektro-linjen. Slik ble det ikke, så man kan si at læringsutbyttet fra prosjektet ble enda større. Man endte til slutt opp med å konstruere en robotarm, som vist i figur 1.2.1 og 1.2.2 under. Hvordan hele systemet endte opp med å se ut, kan ses i figur 1.2.3 under der igjen.



Figur 1.2.1-Det ferdige produktet av robotarmen, med ventil og slange montert fast.



Figur 1.2.2-Hele robotarmen, illustrert som en sammenstilling i Autodesk Inventor [1]. Uten ventil og slanger. Tallene i rødt markerer hvilken motor som blir betraktet som motor nummer 1, 2,3 osv.



Figur 1.2.3-Robotarmen med væskesystemet, flankert av to vaffeljern, i sine tiltenkte posisjoner.

1.3 Begrunnelse

Industrielle robotarmer er gjennom tiden blitt mer og mer aktuelt. I samfunnet og bedrifter kan man se dem overalt. Dette for å automatisere prosesser som tidligere har blitt gjort av mennesker. Vaffelsteking er også en slik prosess. Målet ved denne oppgaven var at robotarmen skulle kunne lette på arbeidsmengden til menneskene som betjener vaffeljernene i steke-prosessen, slik at kapasitet kunne bli frigjort til det man anser som «viktigere» arbeid.

1.4 Begrensinger

Det vil selvsagt alltid være økonomiske begrensninger. Budsjettet styres av fakultet, og en måtte til enhver tid holde oss innenfor disse rammene. De dyreste og beste delene kunne ikke alltid tas i bruk, og en måtte ta høyde for å utføre mesteparten av det fysiske arbeidet selv, for å spare produksjonskostnader. Som gjorde at man kom inn på en annen begrensning. Vi er mennesker, og ikke maskiner, så presisjonen var til enhver tid ikke 100%. Skjevheter kan derfor forekomme, og en del ble ikke alltid som man forventet på forhånd. Dette førte til at

det endelige produktet ikke ble like velfungerende/estetisk sammenlignet med om man skulle fått samtlige deler produsert eksternt. Billigere deler kunne også medført også lavere kvalitet/slitestyrke. Som kunne føre til behov for oftere vedlikehold. En billig motor har, som regel, mindre levetid enn en dyr.

En annen begrensning var kunnskap. Vi er bachelorstudenter, ikke professorer. Mye, og relevant kunnskap har en selvsagt tilegnet seg i løpet av de tre årene på studiet. Likevel er det ikke til å stikke under en stol at enda mer kunnskap hadde gjort flere arbeidsoppgaver enklere, som f.eks. tidsbruken rundt beregninger ved dimensjonering. Så kan det også legges til at vi er maskiningeniørstudenter, ikke elektro- eller dataingeniørstudenter. Oppgaven inneholder, unektelig, mange utfordringer som knytter seg mer til elektro- og dataingeniører enn maskingeniører. Kunnskapen på dette feltet prøvde man å tilegne seg synkront med konstruksjonen av robotarmen. Kunnskapen var åpenbart tilstrekkelig til å utforme et fungerende produkt, men med bakgrunn fra et elektro- eller dataingeniørstudium ville man f.eks. tilført det endelige produktet mer orden/struktur, samt større brukersikkerhet knyttet til håndteringen av programvaren og det elektriske. Når en maskiningeniør tar fatt på oppgavene til en elektro- eller dataingeniør kan løsningene fort bli hakket for pragmatiske.

1.5 Bruksområde

Denne roboten skulle stå på pauserommet til de ansatte på UiS og skal hjelpe med å påføre vaffelrøre på minimum 2 vaffeljern, med plass til 2 vafler hver. Planen var at den skulle kunne modifiseres videre til å gjøre andre oppgaver også. Det kan for eksempel monteres en klypearm på enden som kan plukke opp vafler fra vaffeljernene, eller løfte opp vaffeljernets lokk. Dette ble det også planlagt at robotarmen skulle dimensjoneres for, slik at den tåler belastningene ved arbeidsoppgavene som blir nevnt.

2

Teori og metode

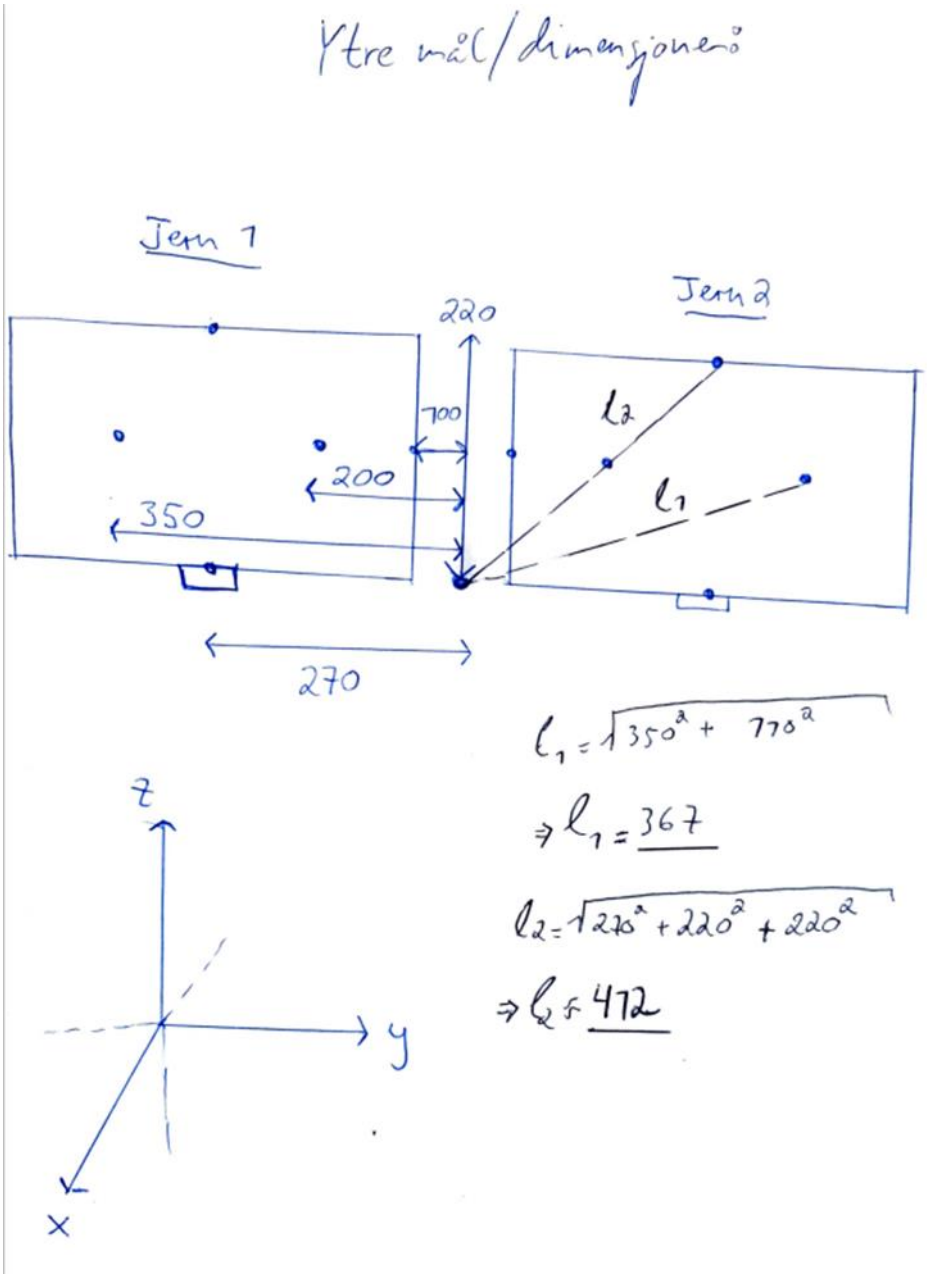
2.1 Ytre mål og grovstruktur for robotarmen

Før noe som helst kunne foretas måtte man ha en grovskisse av hvordan robotarmen skulle se ut, og hva det minimale ytre dimensjonene var. En ville selvsagt at robotarmen skulle være minst mulig, ta minst mulig plass, da mindre materialer impliserer en lavere totalsum. Og en mindre robotarm er til mindre visuell sjenanse. Likevel så skulle robotarmen kunne

gjennomføre et sett med oppgaver, som hadde minimumskrav til målene for robotarmen. Disse oppgavene var:

- Påføre røre ved senter til nærmeste vaffel.
- Påføre røre ved senter til vaffel lengst borte.
- Løfte lokk på vaffeljernet helt opp med gripearm som kan være rotert i fra 0 til 360 grader (dimensjoneres for, men iverksettes ikke med mindre man får ekstra tid til disposisjon).

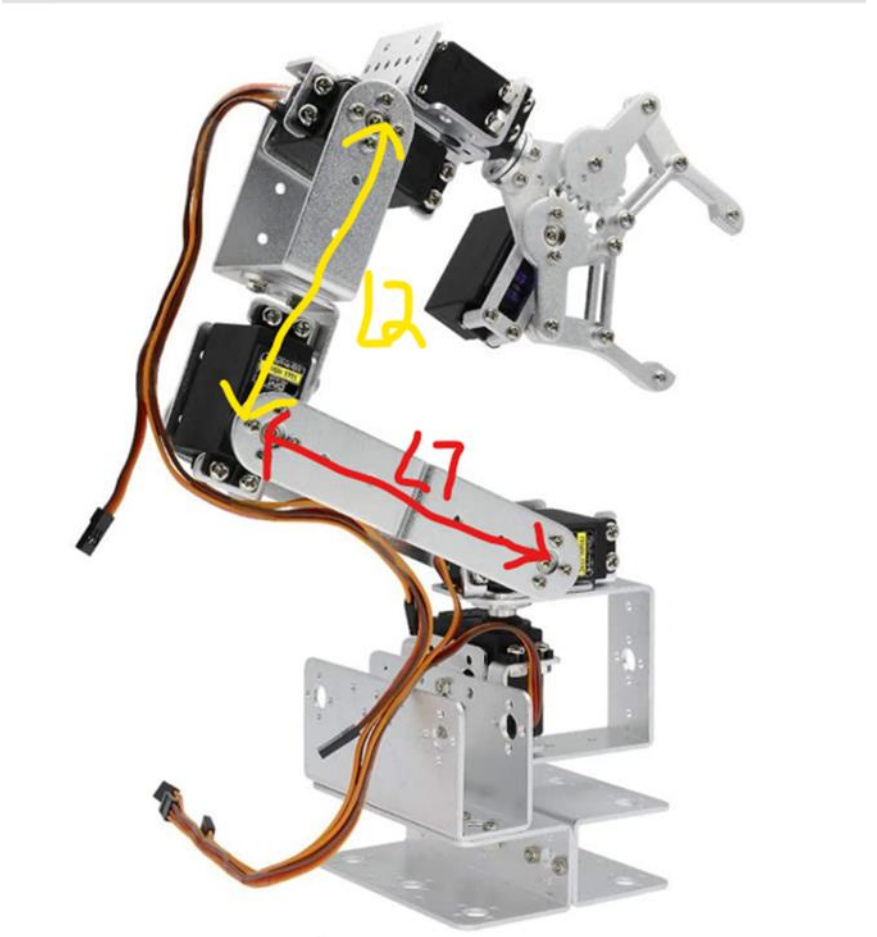
De ytre målene for vaffeljernene kan sees i figur 2.1.1 under. Det ble kun tatt mål av ett vaffeljern, da de to jernene var identiske, og robotarmen etter planen skulle midt imellom jernene, slik at det ble symmetrisk. Her ble det tatt høyde for 100mm avstand fra senter av robotarm til kanten av vaffeljernene.



Figur 2.1.1-Ytre mål og dimensjoner for de 2 vaffeljernene, hvor alle avstander er fra sentrum til robotarmen.

En kom over noen ulike konsepter av en vaffelrobot i forstudierapporten [vedlegg A, kap. 4], laget av studenter og videregående-elever, men det som kjennetegnet alle disse konseptene var at robotarmen hadde blitt kjøpt inn eksternt. I løpet av litteratursøket i forstudierapporten kom en ikke over en eneste vaffelstekende robot, hvor studentene faktisk hadde konstruert robotarmen selv. Det man likevel kom over var en robotarm som kunne kjøpes på Amazon. Det ble derfor planlagt et visuelt design inspirert av robotarmen som ble funnet på Amazon i

våre forundersøkelser [2], vist i figur 2.1.2 under. Denne robotarmen er laget av lettmetall, og styres av enkle/billige servomotorer. En ytre belastning som ved f.eks. løfting av lokket på et vaffeljern ville den derfor ikke tåle. Man kan derfor si at det kun er det enkle konseptet med u-braketter, enkle motorholdere og bolteforbindelser som var inspirasjonen fra denne robotarmen. Når det kom til dimensjonering, måtte en ha en helt annen tilnærming. En annen forskjell er at man planla 6 grader av frihet mot denne armens 5 (feilaktig presentert som 6). Planen fra start ble å montere motor 2 oppå motor 1, motor 4 oppå 3 og motor 6 oppå 5, med brakett-forbindelser imellom her igjen. Med motor 1 menes nederste motor, motor 2 som nest nederste motor, helt opp til motor 6, som er øverste motor. Som tidligere nevnt [vedlegg A, kap. 3.1], så vil en seksakset robotarm kunne betraktes som en menneske-arm, hvor man deler den inn i «skulder», «albue» og «håndledd». Med den planlagte konfigurasjonen vil motor 1 og 2 fungere som skulderledd, motor 3 og 4 som albueledd, og motor 5 og 6 som håndleddet.



Figur 2.1.2- Robotarm av aluminium på Amazon.com, brukt som inspirasjon til visuell utforming [2]

Med dette i bakhånd, kunne man igjen studere kravene til ytre mål for robotarmen: Ved påføring av røre var det en distanse på 350mm i y-retning og 110mm i x-retning (halve lengden av lokket) fra senter av robotarmen. Lengdeendring i z-retning ble foreløpig betraktet som 0, da høyden på skulderleddet var tiltenkt å ha samme høyde som vaffeljernet. Med litt enkel trigonometri (Pytagoras), kunne man slå fast at denne lengden var på 367mm (se figur 2.1.1 over). Ved løfting av lokk helt til toppen var lengdeendring i y-retning 270mm, i x-retning 220mm (lengde av lokk), og i z-retning 220mm (lengde av lokk). Den totale distansen ble da **412mm**. Siden gripearmen skulle kunne være rotert en vilkårlig vinkel fra 0 til 360 grader, var det distansen fra skulderleddet til håndleddet som måtte være minimum 412mm. Avstand fra skulderledd til albueledd ble betraktet som **L1,** og avstand fra albueledd til håndledd ble betraktet som **L2.** Den totale lengden på L1+L2 måtte derfor være minimum 412mm. For å ha litt klaring ble en vilkårlig lengde på 450mm valgt, hvor L1 ble satt til 300mm og L2 til 150mm (vist i figur 2.1.2 over). Grunnen til at L2 ble satt til halvparten av L1, er fordi vekten av motor 5 og 6 skaper et vrimoment om aksene til motor 2 og 3, selv i hvileposisjon. Så man ønsket at L2 var så liten som mulig, samtidig som lengden på forbindelsen var lang nok til at det var tilstrekkelig plass for alle motorer. Begge dimensjonene kunne endres utover i designet, men nå hadde man i alle fall et utgangspunkt for videre beregninger.

2.2 Hvilke motorer og hvor mange?

Etter at de ytre målene var på plass, og det var definert/laget en skisse av de ytre målene for robotarmen, ble utvelgelse av motorer det neste steget. Første spørsmålet var jo: «Hvor mange motorer trenger man for å styre robotarmen med 6 grader av frihet?» -Svaret på det var ganske enkelt. Man trengte en motor for hver grad av frihet, altså 6 motorer. Deretter måtte man bestemme seg for hvilken type motor som var best egnet. Da var svaret på det spørsmålet også ganske enkelt. En bipolar stegmotor ville være det desidert beste alternativet.

Så hvorfor var en bipolar stegmotor det beste alternativet? -Jo, enkelt fortalt fikk man den nøyaktigheten man var ute etter, samt en motor som yter et høyt vrimoment, sammenlignet med f.eks. servomotorer og unipolare stegmotorer [3]. En stegmotor kan ved hjelp av f.eks. halv- fjerdedels- eller åttendedelssteg bevege under en grad for hvert «steg» [4]. Posisjonen til tuppen/gripearmen til robotarmen ble jo styrt av vinklene/posisjonene til hvert ledd, gjerne med flere gjeldende siffer, ut ifra nøyaktighetsgraden. En servomotor ville derfor ikke

tilfredsstille kravene til nøyaktighet, da 90 grader input fra kontrollen kan gi en output-verdi som er langt mindre eller større enn dette [3]. Servomotorer kan som regel heller ikke rotere mer enn maksimum 180 grader. En siste ulempe med servomotorer er at de går direkte fra en posisjon til en annen. Det er ingen steg-for-steg-prosess. Dette fører ofte til høy rotasjonshastighet, samt akselerasjon. Høy akselerasjon fører til større krefter som virker, og et større nødvendig tilført vrimoment fra motoren. Større motor impliserer mer vekt, og mer vekt impliserer større krefter/belastning på robotarmen. Og slik kunne man fortsette i det uendelige. Med en stegmotor kontrollerer en rotasjonshastigheten og akselerasjonen av mer nøyaktig karakter, slik at motoren kan yte det maksimale dreiemomentet. Et fortrinn sammenlignet med unipolare stegmotorer er at bipolare stegmotorer bare har 4 inngangsledninger sammenlignet med de unipolares 5-6 ledninger [5]. De er også relativt enkle å kontrollere med hver sin separate motor-driver, som vist i YouTube-videoen til «DroneBot Workshop» [5].

2.3 Mekanikk for å beregne motors dreiemoment: «Statikk eller dynamikk?» «Newton-Euler og/eller Lagrange?»

Nå hadde man konstatert at robotarmen trengte 6 bipolare stegmotorer, men trengte hver motor å yte det samme vrimomentet? -Det korte svaret på det, var at de gjør de ikke. Noen ledd i en robotarm yter større vrimoment enn andre ledd, og må også kompensere for vekten av de andre motorene. Motoren i det første og nederste leddet må f.eks. kompensere for vekten av hele robotarmen, samt vekten av de 5 andre motorene, i tillegg til en eventuell ytre belastning. Hvert ledd har derfor sitt unike nødvendige vrimoment, og disse måtte defineres.

Å gyve løs på dette med en statikk-tilnærming, med robotarmen i «hvileposisjon», der summen av krefter og momenter er lik 0, ble selvsagt for naivt. Robotarmen skulle jo være i konstant bevegelse mens den utførte oppgavene sine, og vinklene for hvert ledd ville følgelig variere. Å gå inn i dynamikkens verden ville derfor bli en langt mer korrekt tilnærming [6]. Nødvendig tilført vrimoment fra hver motor ville derfor bli en funksjon av hvert ledd sin posisjon, samt ledd-hastighetene- og akselerasjonene. Man kunne holdt seg i den velkjente Newton-Euler-mekanikken for å beregne disse vrimomentene, men det finnes en annen type mekanikk som gjør det langt enklere å systematisere informasjonen fra robotarmen, og få en output i form av 6 likninger, en for hvert vrimoment. Denne mekanikken ga kun tilført vrimoment, og ikke reaksjonskrefter/bøyemomenter, så en måtte på et tidspunkt over på Newton-Euler mekanikken uansett når en begynte med dimensjonering av bæredelene. Enn

så lenge så var det vrimomentet man var interessert i, så da var det en fin plass å starte. Det ga også en ekstra sikkerhet rundt resultatene ved at en benyttet seg av to metoder for å komme fram til (forhåpentligvis) det samme svaret.

Mekanikken som ble beskrevet heter Lagrange-mekanikk. Kort fortalt bygger hele mekanikken på to tilsynelatende enkle likninger [7]. Lagrange-likningen og Euler-Lagrange-likningen [7]. Lagrange-likningen er lik kinetisk energi minus potensiell energi (likning 2.3-1 under), hvor Euler-Lagrange-likningen er den tidsderiverte av den delvis deriverte av Lagrange-likningen, med hensyn til hastighet, minus den delvis deriverte med hensyn til posisjon (likning 2.3-2 under). Den sistnevnte likningen, som fungerer som en ekvivalent til Newtons andre lov, ga til slutt gi vrimomentene for hver motor. Neste punkt på listen ble å definere en Euler-Lagrange-likning for den 6-aksede robotarmen. Beviset følger i figur 2.3.1 og 2.3.2 under, inspirert av utledningen i YouTube-videoen til «Engineering Educator Academy» [8]. Forklaringer til figur 2.3.1 og 2.3.2 følger i neste avsnitt. Uttrykkene for kinetisk- og potensiell energi kan ses i likning 2.3-3 og 2.3-5 under. Så kunne en jo spørre seg nødvendigheten ved å utlede akkurat denne formelen/bevegelseslikningen. For det første var det essensielt for forståelsen av en helt ny form for mekanikk, samtidig var det viktig for å validere formlene som ble brukt for å beregne vrimomentene. Videoen hadde tross alt ikke mer enn 157 visninger (per 03.02.2022) [8], og skulle det vise seg at informasjonen ikke stemte, ville man i ytterste konsekvens endt opp med feildimensjonerte motorer, som kunne ført til at vrimomentene ikke var tilstrekkelige.

$$Lagrangelikingen: L = K_e - P_e \qquad (2.3-1)$$

$$Euler - Lagrange - likningen: \frac{d}{dt}\left(\frac{\partial L}{\partial q}\right) - \left(\frac{\partial L}{\partial q}\right) = \tau \qquad (2.3-2)$$

$$Hvor\ q\ er\ posisjonen\ til\ hvert\ ledd, og\ \tau\ er\ dreiemomentene.$$

$$Uttrykk\ for\ kinetisk\ energi: K_e = \frac{1}{2}\dot{q}^T D(q)\dot{q} \qquad (2.3-3)$$

$$\dot{q}\ er\ leddhastighetene\ i\ en\ kolonnevektor\ og\ \dot{q}^T\ er\ den\ transponerte.$$

$$Treghetsmatrsien, D(q) = \sum_{i=1}^{6}(m_i J_{vi}^T J_{vi} + J_{wi}^T R_i I_i R_i^T J_{wi}) \qquad (2.3-4)$$

J_{vi} *er den lineære komponenten av Jacobi*

$-$ *matrisen,* J_{wi} *er rotasjomskomponenten*

J_{vi}^T *og* J_{wi}^T *er de transponerte*

R_i *er rotasjonsmatrise fra globale koordinater til lokale koordinater,* R_i^T *er den transponerte*

I_i *er treghetstensoren som* $3x3 -$ *matrise for hvert ledd, i lokale koordinater.*

$$Og\ den\ potensielle\ emergien: P_e = \sum_{i=1}^{6} g^T m_i \Delta z_i \qquad (2.3-5)$$

Likning for bevegelse for 6-grad-av-frihet-robot:

① $L = KE - PE$, $KE = KE\text{-translatorisk} + KE\text{-rotasjon}$

② $\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \left(\frac{\partial L}{\partial q}\right) = \tau$

$\dot{q}$ = fart ved ledd, q = leddposisjon.

Kinetisk energi ved hvert ledd:

$i = 1:6:\ KE_i = \frac{1}{2} m_i v_i^2 + \frac{1}{2} I_i \omega_i^2$

$\Rightarrow KE = \sum_{i=1}^{6} \frac{1}{2} m_i v_i^2 + \frac{1}{2} I_i \omega_i^2$

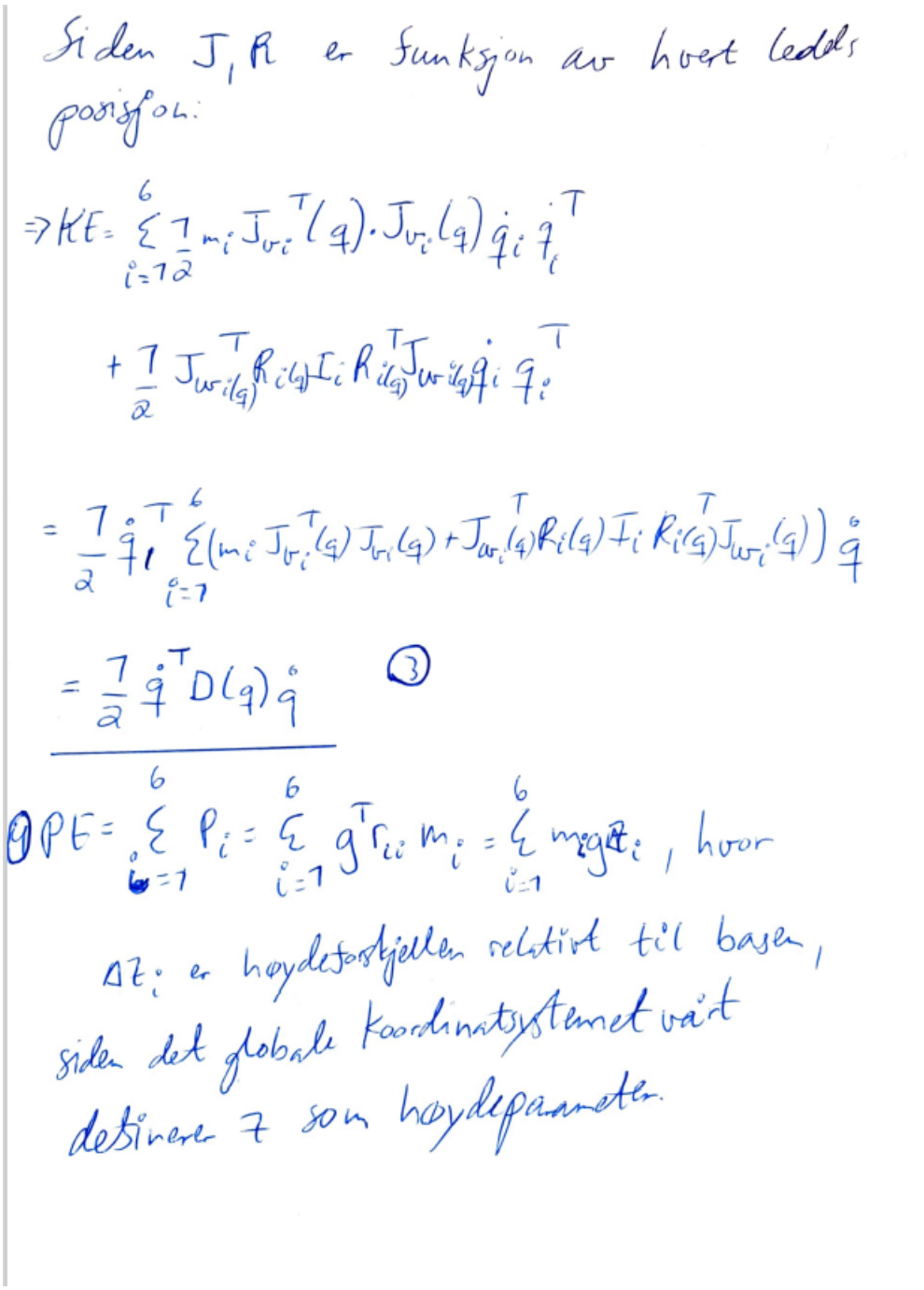
$v_i = J_{vi} \dot{q}_i$, $\omega_i = J_{wi} \dot{q}_i$

$\Rightarrow v_i^2 = J_{vi} \dot{q}_i J_{vi}^T \dot{q}_i^T$, $\omega_i^2 = J_{wi} \dot{q}_i J_{wi}^T \dot{q}_i^T$

I ~~stil~~ base ~~fra~~ koordinatsystem for hvert ledd:

$I_i = I_i R_i^2 = R_i I_i R_i^T$, hvor $R_i = R_0^1, R_0^2, R_0^3 \ldots$

Figur 2.3.1-Første del av utledningen hvor en finner et generelt uttrykk for kinetisk- og potensiell energi, samt de generelle Lagrange- og Euler-Lagrange-ligningene



Figur 2.3.2-De endelige uttrykkene for kinetisk- og potensiell energi.

Som man så i likning 2.3-4 over, var den kinetiske energien ved hvert ledd en sum av den lineære/translatoriske energien, og energien som følge av rotasjon. Jacobimatrisen er en matrise som inneholder de delvis deriverte [9]. For en robotarm med 6 ledd, ville den være av størrelsen 6x6, med en kolonne for hvert ledd. Ganget med ledd-hastighetene i form av en

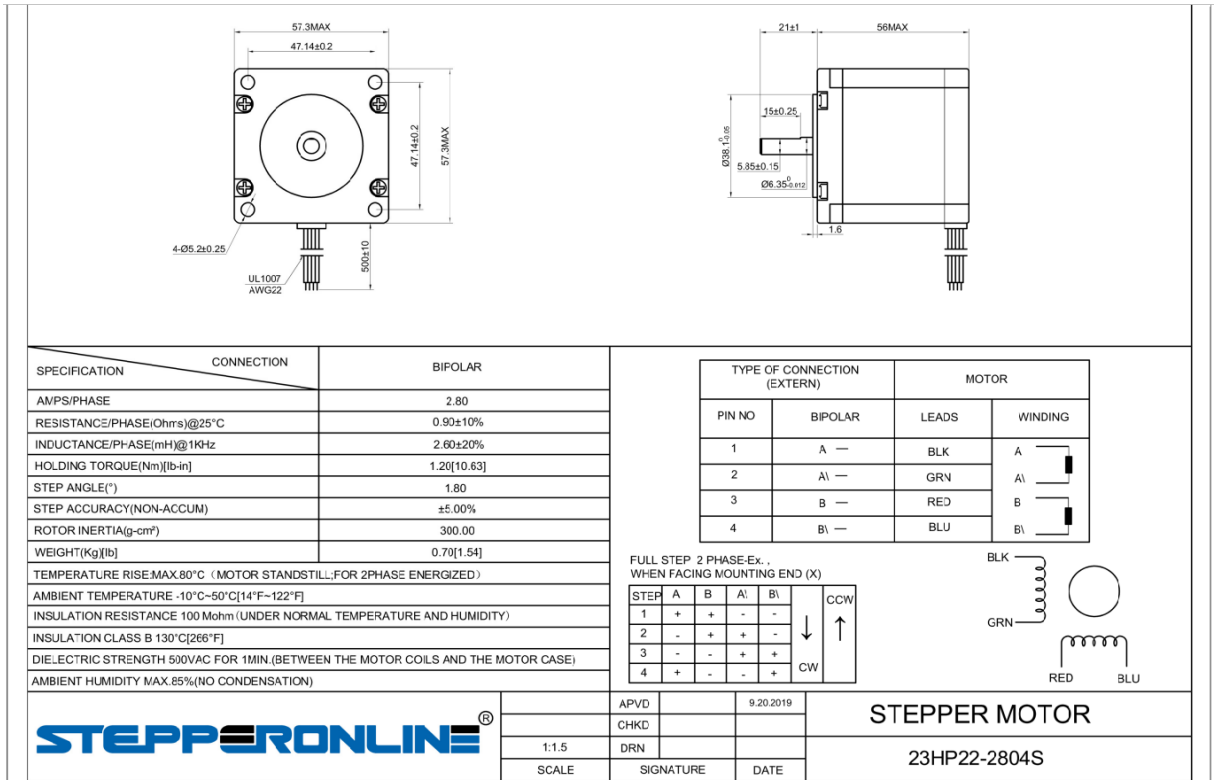
kolonnevektor, ga de tre øverste radene den lineære hastigheten-, mens de tre nederste radene ga rotasjonshastigheten for hvert ledd. Vi kunne derfor omforme uttrykket for kinetisk energi. Eneste forskjellen var at når vi opphøyde hastigheten i andre, måtte matrisen ganges med sin egen transponerte matrise. Når det gjelder treghetsmomentet for hvert ledd, var det følgelig i matriseform(3x3), da rotasjon kunne skje om alle 3 aksene [10]. For å få treghetsmomentet relativt til det globale koordinatsystemet måtte man gange med rotasjonsmatrisene fra basen til aktuelt ledd, samt den inverse/transponerte rotasjonsmatrisen. Man endte da til slutt opp med et uttrykk for den kinetiske energien i likning 3. Hvor D ofte blir kalt for treghetsmatrisen [8]. Med den potensielle energien var det hakket enklere. Det var bare summen av massen ganget med tyngdeakselerasjon og en eventuell høydeforskjell relativt til basen(z-retning). Uttrykk vist i likning 2.3-5 over.

2.4 Utforming av fremover kinematikk

Man så at om man skulle regne ut et uttrykk for den kinetiske- og potensielle energien, var det en del parametere som manglet. F.eks. rotasjonsmatriser, Jacobimatrise, treghetsmomenter og høydeforskjeller. For å finne disse parameterne trengte man en modell for fremover kinematikk [11]. En modell som forteller hva posisjonen til tuppen av robotarmen er, basert på rotasjonsvinklene til leddene. For å definere robotarmen i hvileposisjon tok man et utgangspunkt i de maksimale/minimale ytre målene som hadde blitt foretatt (figur 2.1.1 og 2.1.2).

For å definere koordinatsystemer for hvert ledd trengte man å vite dimensjonene på motorene. En måtte derfor bare velge en vilkårlig motor innenfor de rammene man trodde var egnede. Da ville man også få en masse å forholde seg til i hvert ledd. Man så i de framtidige beregningene bort fra massen til materialet som forbinder motorene, så det var derfor svært viktig å få definert motorvekten m.tp. mest mulig korrekte beregninger. En tok høyde for en sikkerhetsfaktor på 3, dvs. at bremsemomentet på motoren skulle være 3 ganger større enn nødvendig tilført vrimoment. Forhåndsparameterne ble redefinert for hver motorutvelgelse (startet øverst med motor 6, og jobbet nedover mot motor 1), og beregningene ble gjennomført på nytt. Noen ledd ville mest sannsynlig måtte tilføre større vrimoment enn den forhåndsutvalgte motoren, og andre mindre. Motoren som ble brukt i beregningene var en NEMA 23 bipolar stegmotor (som vist i figur 2.4.1 og 2.4.2 under), med bremsemoment på 1,26Nm [12].

Figur 2.4.1-NEMA 23 57mmx57mm bipolar stegmotor [12]



SPECIFICATION / CONNECTION	BIPOLAR
AMPS/PHASE	2.80
RESISTANCE/PHASE(Ohms)@25°C	0.90±10%
INDUCTANCE/PHASE(mH)@1KHz	2.60±20%
HOLDING TORQUE(Nm)[lb-in]	1.20[10.63]
STEP ANGLE(°)	1.80
STEP ACCURACY(NON-ACCUM)	±5.00%
ROTOR INERTIA(g-cm²)	300.00
WEIGHT(Kg)[lb]	0.70[1.54]
TEMPERATURE RISE:MAX.80°C（MOTOR STANDSTILL;FOR 2PHASE ENERGIZED）	
AMBIENT TEMPERATURE -10°C~50°C[14°F~122°F]	
INSULATION RESISTANCE 100 Mohm (UNDER NORMAL TEMPERATURE AND HUMIDITY)	
INSULATION CLASS B 130°C[266°F]	
DIELECTRIC STRENGTH 500VAC FOR 1MIN.(BETWEEN THE MOTOR COILS AND THE MOTOR CASE)	
AMBIENT HUMIDITY MAX.85%(NO CONDENSATION)	

TYPE OF CONNECTION (EXTERN)		MOTOR	
PIN NO	BIPOLAR	LEADS	WINDING
1	A —	BLK	A
2	A\ —	GRN	A\
3	B —	RED	B
4	B\ —	BLU	B\

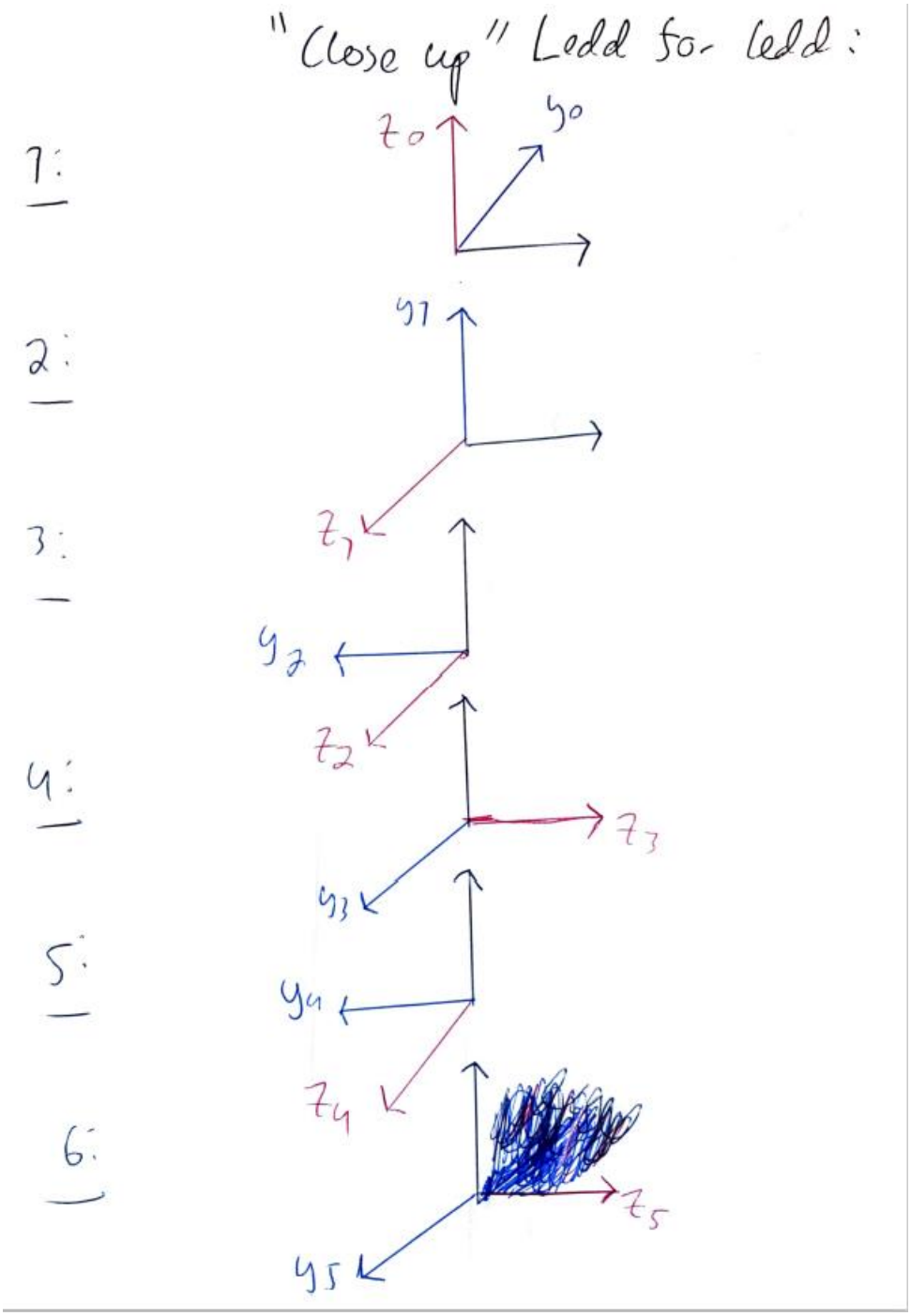
STEP	A	B	A\	B\
1	+	+	-	-
2	-	+	+	-
3	-	-	+	+
4	+	-	-	+

Figur 2.4.2-Dimensjoner for NEMA 23 bipolar stegmotor [12]

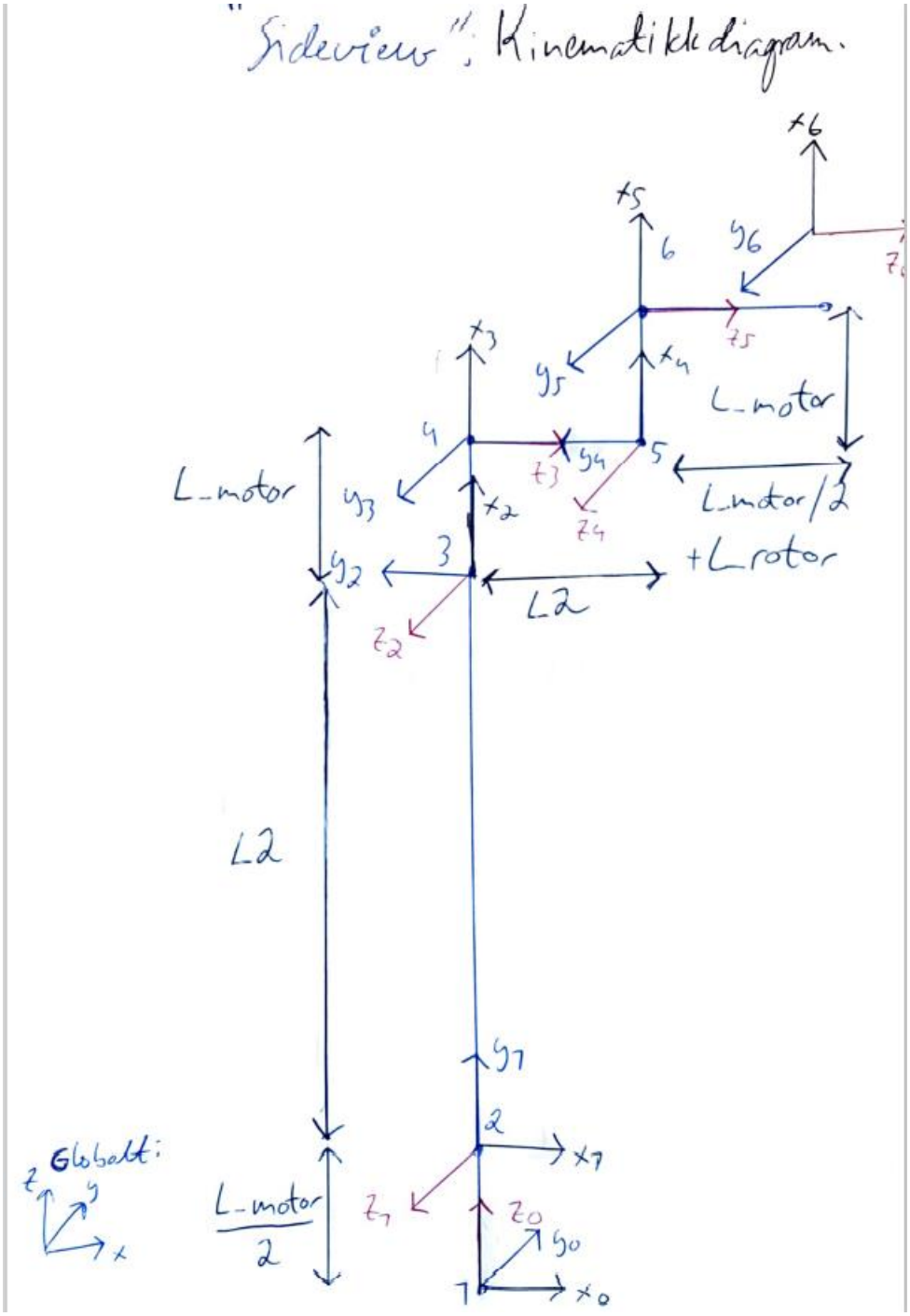
Når man så hadde ytre mål, en grovstruktur for design og dimensjoner på eksempel-motor for hånden, kunne en fremover-kinematikk-modell utformes. Det første steget for å gjøre dette ble å utforme koordinatsystemer for basen og for hvert av leddene, for å deretter finne de såkalte Denavit-Hartenberg-parametrene [13]. Det er kort fortalt 4 parametere for hvert ledd, som er med å definere strukturen/sammensetningen til armen. Når det kom til definering av koordinatsystemer for hvert ledd, brukte man et sett med regler [14]:

1. z-aksen er rotasjonsaksen som hvert ledd roterer om (regner motsols rotasjon som positiv, og bruker høyrehåndsregel).
2. x-aksen må være vinkelrett på både sin egen z-akse, i tillegg til den forrige z-aksen (z_n og z_{n-1}). x-akse i motsatt retning av rotasjonen om z-aksen kommer til å bli brukt, men det finnes to lovlige konfigurasjoner (med z-rotasjon og mot z-rotasjon).
3. y-aksen blir definert etter høyrehåndsregelen.
4. x-aksen må ha skjæringspunkt med den forrige z-aksen(z_{n-1}). Bortsett fra ved ledd 1, da det selvsagt ikke finnes noen «forrige» z-akse her.
5. Om z-akse og forrige z-akse (z_n og z_{n-1}) er parallelle ligger x-aksen langs den vinkelrette linjen/normalen mellom disse to aksene.

Basert på disse reglene, grovstruktur til armen og dimensjonene til motorene ble det utformet koordinatakser for hvert ledd, samt et globalt koordinatsystem. Vekten av materiale ble neglisjert, da disse foreløpig var ukjent. Aksene ble først tegnet separat (figur 2.4.3 under) før de ble plassert sammen i et kinematisk diagram [15], hvor ovennevnte regler ble overholdt (figur 2.4.4 under der igjen).



Figur 2.4.3-Koordinatsystemer for hvert av de 6 leddene, etter gjeldende konvensjoner for DH-parametere.



Figur 2.4.4-Kinematikkdiagram for den 6-aksede robotarmen, tegnet i det globale koordinatsystemet.

Når koordinatsystemene var på plass, kunne de 24 Denavit-Hartenberg-parametrene (DH) defineres. Også her er det et sett med regler som var med på å definere hva de 4 ulike parameterne var [13]:

- **r** (noen ganger benevnt a): Forskyvning langs felles normal(x_{n+1}). Med andre ord lengdeforandring i x-retning.
- **$\boldsymbol{\theta}$:** Vinkel rotert om forrige z-akse(z_{n-1}). Så denne vinkelen blir rotasjonen om den forrige z-aksen, tillegg til vinkelen som den nye x-aksen(x_n) allerede har rotert om den forrige z-aksen i hvileposisjon.
- **d:** Avstand langs forrige z-akse(z_{n-1}) til felles normal.
- **α:** Vinkel rotert om felles normal(x_n), fra forrige- til nåværende z-akse(z_n). Det vil si en rotasjon om x-aksen.

Med disse parameterne i bakhånd ble det laget en DH-tabell (tabell 2.4). l_m er lengden av motoren, h_m er høyden av motoren og l_r er lengden av rotoren.

Tabell 2.4- Denavit-Hartenberg-parametre for robotarmen

	r	**α**	**d**	θ
1	0	90	$\frac{h_{m2}}{2}$	θ_1
2	L1	0	0	$\theta_2 + 90$
3	$\frac{h_{m3}}{2} + \frac{h_{m4}}{2}$	90	0	θ_3
4	0	-90	L2	θ_4
5	$\frac{h_{m5}}{2} + \frac{h_{m6}}{2}$	90	0	$\theta_5 - 90$
6	0	0	$\frac{l_{m6}}{2} + \frac{l_{r6}}{2}$	θ_6

Informasjon i DH-tabellen (tabell 2.4) ga grunnlaget for å lage en fremover-kinematikk-modell i MATLAB [vedlegg E]. Kort fortalt så måtte det lages transformasjonsmatriser for hvert ledd i forhold til basen, som til slutt ga transformasjon (rotasjon og translasjon) fra base til tuppen. Transformasjon fra et ledd til neste fikk man ved å gange rotasjon om x- og z-akse, samt translasjon langs disse to aksene (som vist i figur 2.4.5 under). Kommentarer rundt utforming står i koden. Det kan legges til at koden som er vedlagt, er med de endelige parameterne, og ikke med utgangsparameterne som tar høyde for at hver motor er den samme. Det være seg modifiserte DH-parametere, samt motordimensjoner, i tillegg til dimensjoner for ventilen.

$$\mathrm{Trans}_{z_{n-1}}(d_n) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_n \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$\mathrm{Rot}_{z_{n-1}}(\theta_n) = \left[\begin{array}{ccc|c} \cos\theta_n & -\sin\theta_n & 0 & 0 \\ \sin\theta_n & \cos\theta_n & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$\mathrm{Trans}_{x_n}(r_n) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & r_n \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$\mathrm{Rot}_{x_n}(\alpha_n) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_n & -\sin\alpha_n & 0 \\ 0 & \sin\alpha_n & \cos\alpha_n & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

This gives:

$${}^{n-1}T_n = \left[\begin{array}{ccc|c} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & r_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & r_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

where R is the 3×3 submatrix describing rotation and T is the 3×1 submatrix describing translation.

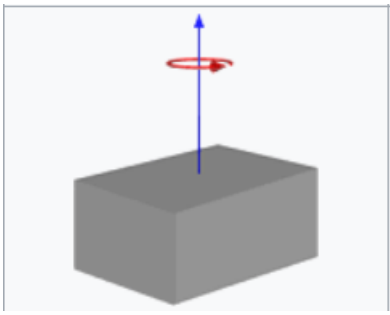
Figur 2.4.5-Utforming av transformasjonsmatriser med DH-parametere [13].

2.5 Utarbeiding av treghetsmomenter og massesentre

Når man så hadde funnet rotasjons- og lineære komponenter av Jacobimatrisen [vedlegg E] trengte man bare treghetsmomentet for hvert ledd for å starte på beregningen av vrimomentene. Følgende antagelser ble gjort ved beregning av treghetsmomenter [10]:

- Treghetsmomentet for hvert ledd er summen av treghetsmomentet som trengs for å rotere rotoren, samt vekten av selve leddet som skal roteres, i tillegg til vekten fra en eventuell ytre belastning.
- Man antok at vekten at motorene var såpass mye større enn vekten av forbindelsene mellom, at massesenteret for hele leddet var massesenteret til motoren.
- Rotasjonen skjer om massesenteret. Pga. antagelsen i det forrige punktet antok man at rotasjonen skjedde om koordinataksene som allerede er definert i figur 2.4.4 over.
- En så bort fra friksjon inni selve motoren, girkasser, og mellom forbindelsene.
- Antok at motorene kunne betraktes som rette prismer når en plasserte massesenter og beregnet treghetsmatrise (se figur 2.5.1 under).

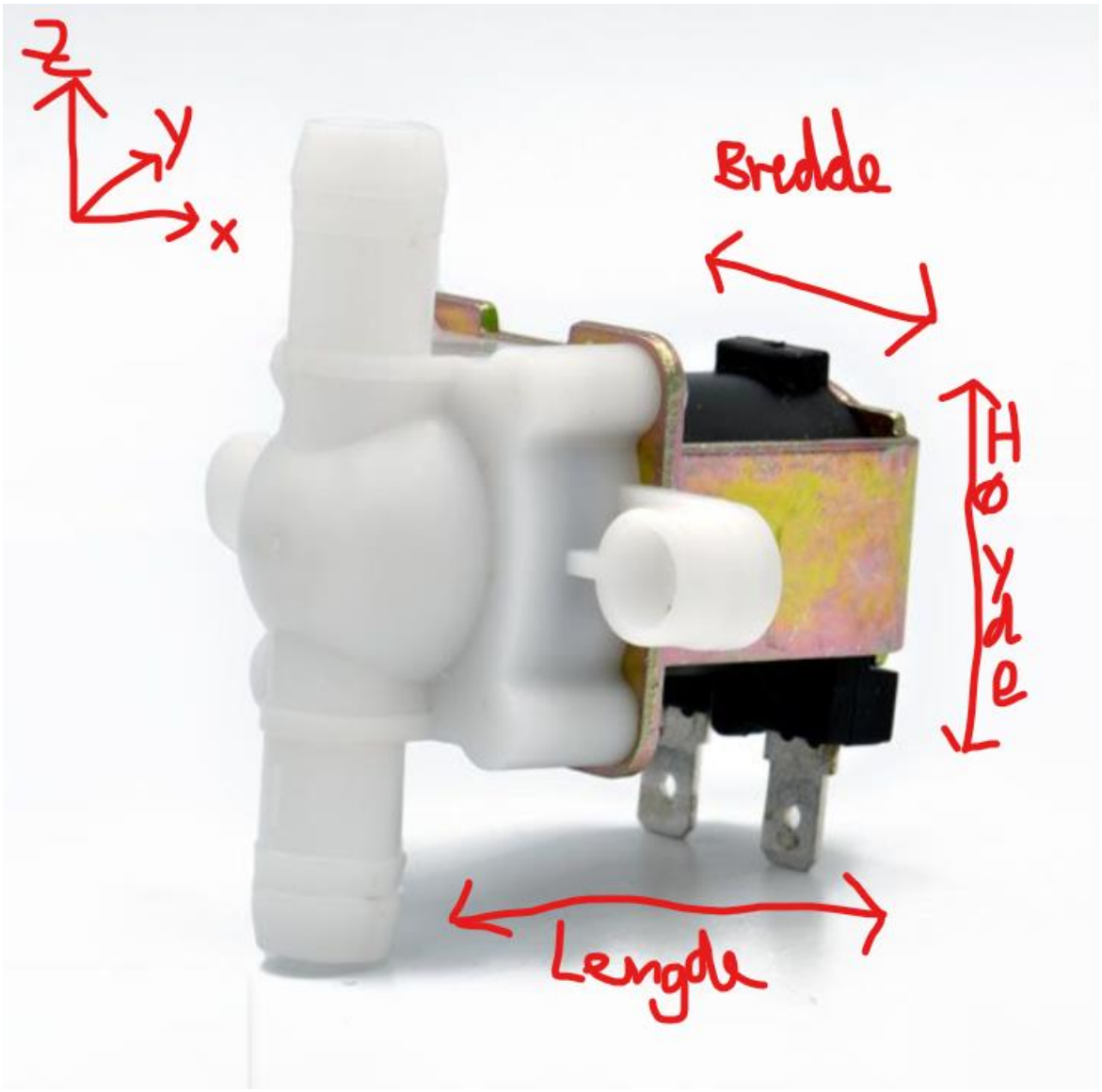
- Rotasjon om 3 akser tilsa at treghetsmomentet ble beskrevet som en 3x3 matrise/treghets-tensor [10].
- En antok at alle forbindelser mellom motorene, og motorene selv, kunne betraktes som ubøyelige objekter.



$$I = \begin{bmatrix} \frac{1}{12}m(h^2+d^2) & 0 & 0 \\ 0 & \frac{1}{12}m(w^2+d^2) & 0 \\ 0 & 0 & \frac{1}{12}m(w^2+h^2) \end{bmatrix}$$

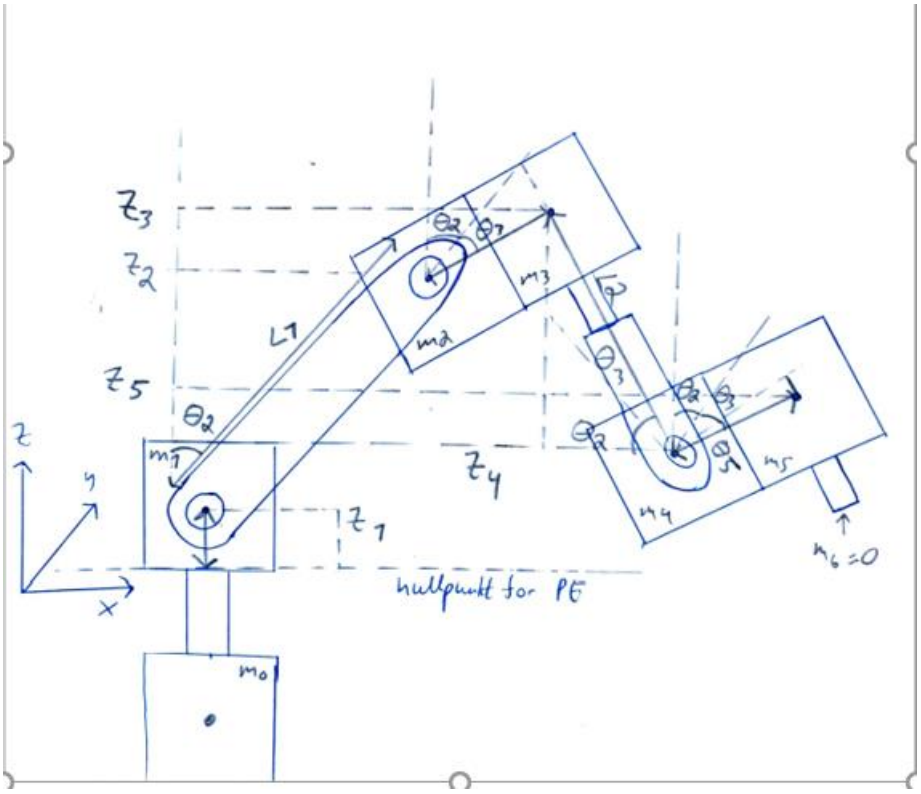
Figur 2.5.1-Treghetsmatrise for kube i rommet [10].

Treghetsmomentet for rotasjon av rotoren var oppgitt i spesifikasjonene til eksempelmotoren, NEMA 23 [12]. Det ble siden endret, etter hvert som nye motorer ble valgt. Formel for treghetsmoment av kube med masse *m* er vist i figur 2.5.1 over. Disse parameterne ble også endret for hver ny motor som ble valgt, og ga derfor både større og mindre dreiemoment for de andre motorene, Når det gjaldt massen som motor 6 måtte rotere, var dette snakk om massen av en ventil. Det ble kjøpt inn en magnetventil, som senere ble festet til akslingen på motor 6 [16], med dimensjoner vist i figur 2.5.2 under.



Figur 2.5.2-De tiltenkte målene på ventilen [16]

Trigonometrien som er skissert i figur 2.5.3 under ble benyttet for å bestemme den globale z-koordinaten til massesentrene for hver motor, som jo er parameteren som bestemte den potensielle energien. Uttrykk for hver z-koordinat er vist i likning 2.5-1 til 2.5-5 under der igjen. Massesenter til motor nummer 1 ble ikke funnet ikke relevant, da det ikke var noen andre motorer som måtte «bære vekten av den». Nullpunktet for potensiell energi ble derfor satt på enden av rotoren til motor 1. Man kunne spesielt legge merke til at den potensielle energien kun var funksjon av leddposisjonene til ledd 2,3 og 5 ($\theta_2, \theta_3 \; og \; \theta_5$). Og dette ga mening, da massene kun skapte et vrimoment om disse leddene. En kunne selvsagt også bare tatt z-komponenten til hver at de 6 transformasjonsmatrisene (rad 3, kolonne 4), for å finne massesenteret, men å igjen benytte seg av 2 ulike framgangsmåter, og samtidig få samme resultat var med på å validere at transformasjonsmatrisene var korrekte.



Figur 2.5.3 -Trigonometrisk skisse av robotarmen, som dannet grunnlaget for uttrykkene for massesentrene i ligning 5-9.

$$\boldsymbol{Massesenter\ 1}: \boldsymbol{z_1 = \frac{h_{m2}}{2}} \qquad (\mathbf{2.5-1})$$

h_{m2} *er høyden til motor* 2.

$$\boldsymbol{Massenter\ 2}: \boldsymbol{z_2 = z_1 + \cos\theta_2 L1 = \frac{h_{m2}}{2} + \cos\theta_2 L1} \qquad (\mathbf{2.5-2})$$

$L1$ *er distansen fra rotasjonsaksen til motor* 2 *til rotasjonsaksen til motor* 3.

$$Massesenter\ 3: z_3 = z_2 + \cos(\theta_2 + \theta_3) * \left(\frac{h_{m3}}{2} + \frac{h_{m4}}{2}\right)$$

$$\rightarrow\ \boldsymbol{z_3 = \frac{h_{m2}}{2} + \cos\theta_2 L1 + \cos(\theta_2 + \theta_3) * \left(\frac{h_{m3}}{2} + \frac{h_{m4}}{2}\right)} \qquad (\mathbf{2.5-3})$$

h_{m3} *er høyden til motor* 3 *og* h_{m4} *er høyden til motor* 4.

$$massesenter\ 4: z_4 = z_3 + \sin(\theta_2 + \theta_3) * L2$$

$$\rightarrow\ \boldsymbol{z_4 = \frac{h_{m2}}{2} + \cos\theta_2 L1 + \cos(\theta_2 + \theta_3) * \left(\frac{h_{m3}}{2} + \frac{h_{m4}}{2}\right) + \sin(\theta_2 + \theta_3) * L2} \quad (\mathbf{2.5-4})$$

$L2\ er\ distansen\ fra\ senter\ av\ motor\ 4\ til\ rotasjonsaksen\ til\ motor\ 5$

$$massesenter\ 5{:}\ z_5 = z_4 + \cos(\theta_2 + \theta_3 + \theta_5) * \left(\frac{h_{m5}}{2} + \frac{l_{m6}}{2}\right)$$

$$\rightarrow\ \boldsymbol{z_5 = \frac{h_{m2}}{2} + \cos\theta_2 L1 + \cos(\theta_2 + \theta_3) * \left(\frac{h_{m3}}{2} + \frac{h_{m4}}{2}\right) + \sin(\theta_2 + \theta_3) * L2}$$

$$\boldsymbol{+ \cos(\theta_2 + \theta_3 + \theta_5) * \left(\frac{h_{m5}}{2} + \frac{l_{m6}}{2}\right)} \qquad (\mathbf{2.5-5})$$

$h_{m5}\ er\ høyden\ til\ motor\ 5\ og\ l_{m6}\ er\ lengden\ til\ motor\ 6.$

2.6 Utarbeiding av metode for å finne Christoffel-symboler av første orden

Nå hadde man et uttrykk for både den totale kinetiske- og potensielle energien. Begge som funksjon av leddposisjonene (q). Lagrangelikningen var derfor definert (L=KE-PE). Problemet var bare at dreiemomentene skulle beregnes, og for å gjør dette måtte man definere Euler-Lagrange-likningen. Likningene var selvsagt for store og stygge til å deriveres for hånd, så en måtte ha en metode/struktur som forenklet denne prosessen, og ga et uttrykk for vrimomentene. Det fantes heldigvis en metode som gjorde nettopp dette, og denne metoden involverte noe som heter Christoffel-symboler [17], som kort fortalt beskriver coriolis-akselerasjon [18] og sentripetalakselerasjon [18]. Utledningen av hvordan en endte opp med disse symbolene, som et resultat av de delvis deriverte og tidsderiverte av Lagrange-likningen følger likning 2.6-1 til 2.6-3 under. Utledningen finnes i figur 2.6.1 til 2.6.3 under der igjen. Når det gjelder utledningen av den generelle formuleringen har ble det hentet inspirasjon fra YouTube-videoen «Dynamics of the n-link Robotic Manipulator (Multi-body Mechanical Systems): General Formulation» , av Ph.d. Ali Raza [19]. Under, etter figur 2.6.3, i likning 2.6-4 følger også en fremgangsmåte for å samle de 216 Christoffel-symbolene i den 6x6 «C-matrisen» [19]. Utledning for dette følger i figur 2.6.4.

$Formel\ for\ den\ \ delvis\ deriverte\ av\ den\ tidsderiverte\ av\ den\ kinetiske\ energien$:

$$\boldsymbol{\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj}\ddot{q}_j + \sum_{i,j} \frac{\partial d_{kj}}{\partial q_i}\dot{q}_i\dot{q}_j} \qquad (\mathbf{2.6-1})$$

$Hvor\ \dot{q}\ er\ leddhastighetene, \ddot{q}\ er\ akselerasjonene\ og\ d\ er\ treghetsmatrisen$

Formel for den delvis deriverte av kinetisk – og potensiell energi:

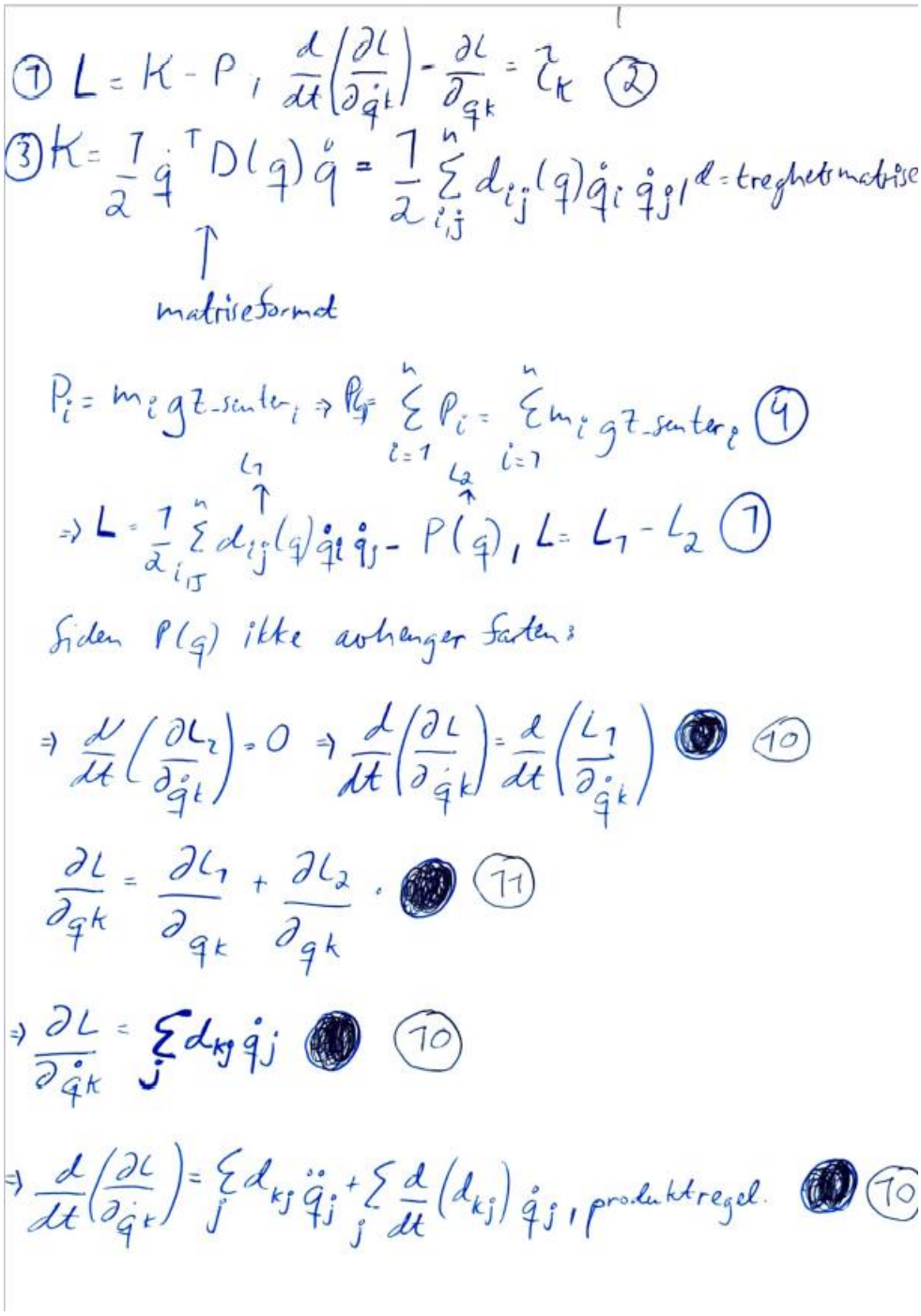
$$\frac{\partial L}{\partial q_k} = \frac{1}{2}\sum_{i,j}\frac{\partial d_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j - \frac{\partial P_e}{\partial q_k} \qquad (2.6-2)$$

Omforming av andre ledd i likning (10):

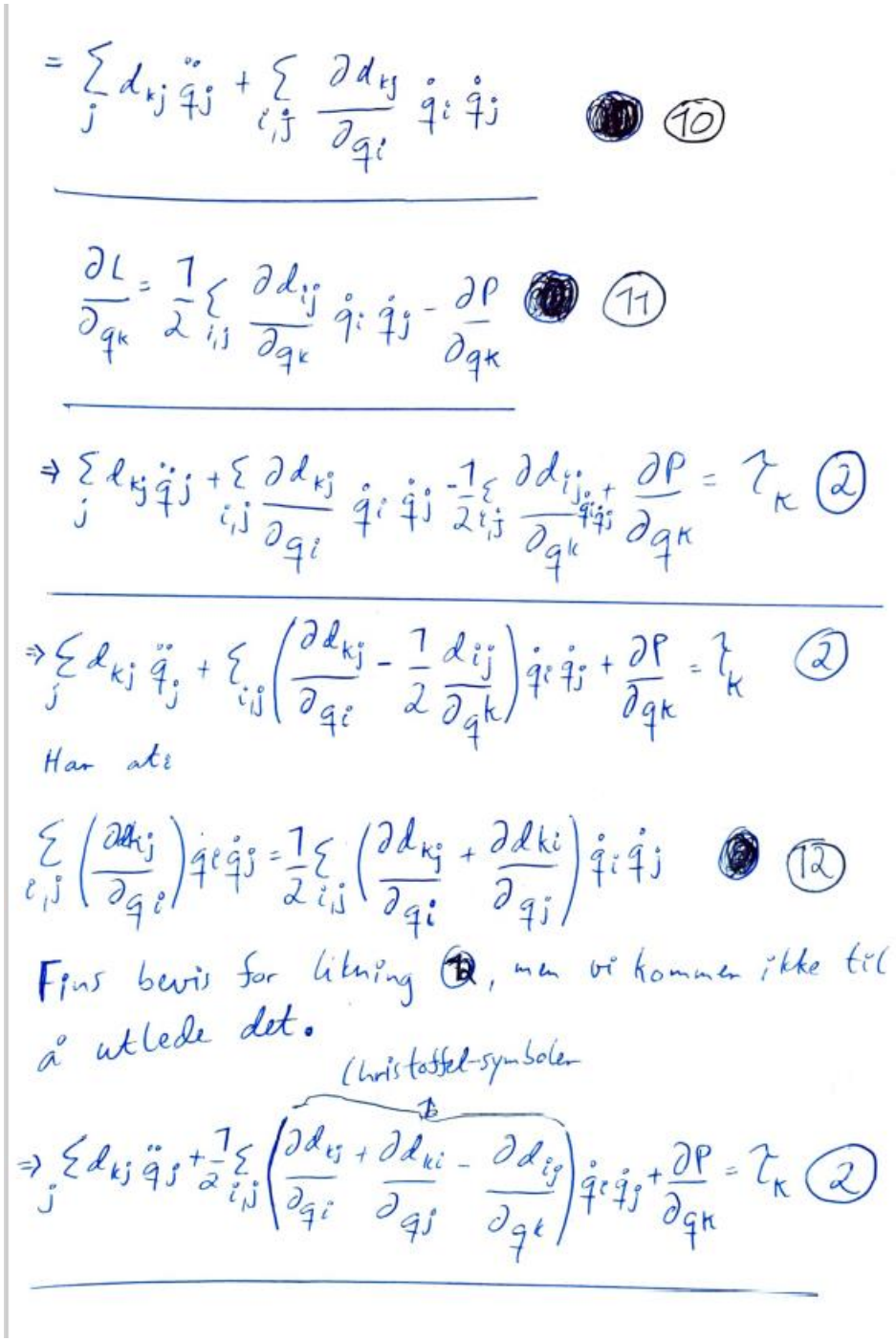
$$\sum_{i,j}\frac{\partial d_{kj}}{\partial q_i}\dot{q}_i\dot{q}_j = \frac{1}{2}\sum_{i,j}\left(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j}\right)\dot{q}_i\dot{q}_j$$

Formel for Christoffel – symboler av første grad:

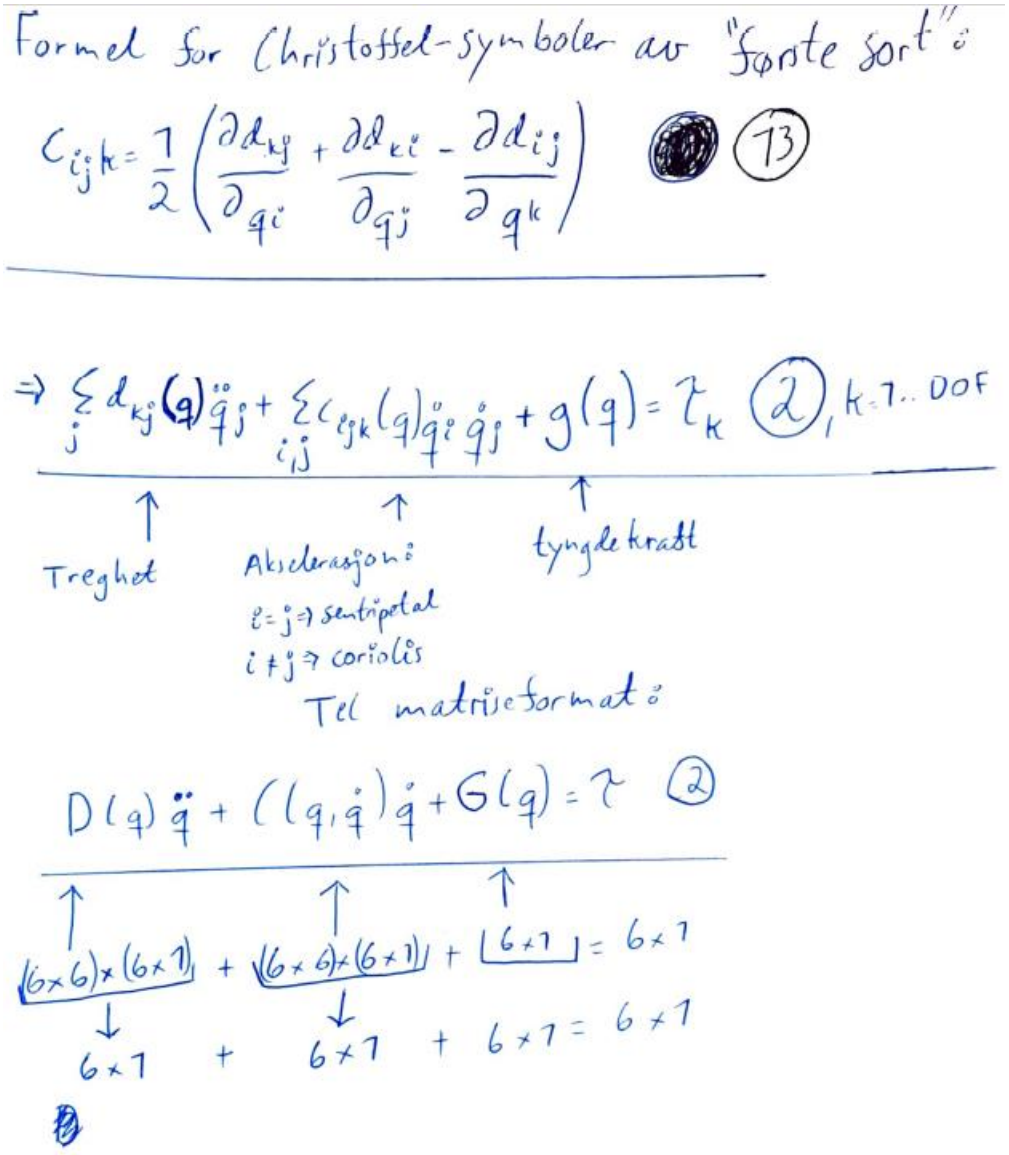
$$c_{ijk} = \frac{1}{2}\left(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k}\right) \qquad (2.6-3)$$



Figur 2.6.1-Første del hvor en fant den tidsderiverte og delvis deriverte av kinetisk -og potensiell energi.



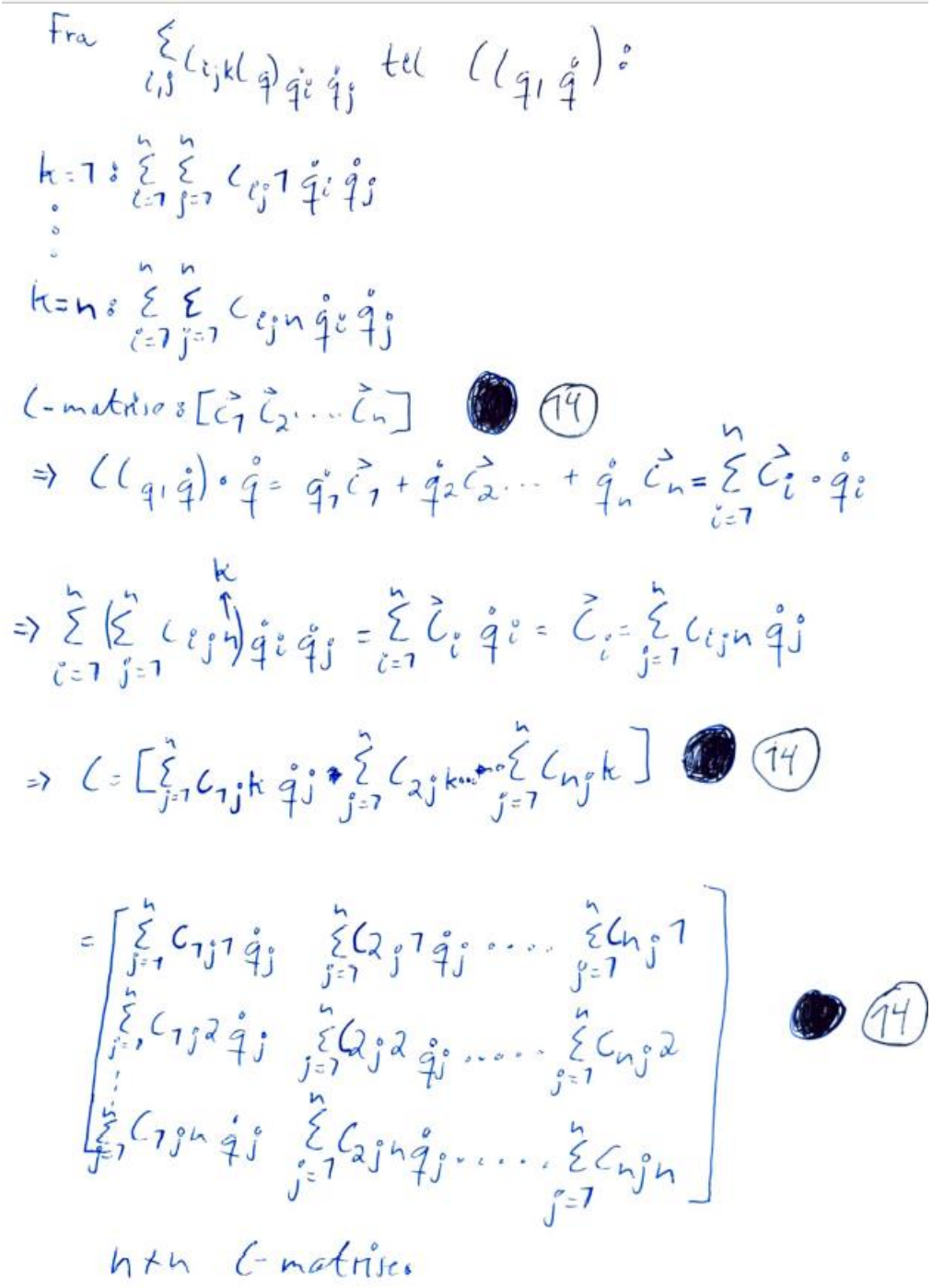
Figur 2.6.2-Fant et uttrykk for Christoffel-symboler av første grad som et resultat av dette.



Figur 2.6.3-Det endelige uttrykket for Euler-Lagrange-likningen i matriseform.

Fremgangsmåte for å samle Christoffel − symbolene i den nxn C − matrisen:

$$\boldsymbol{C} = \begin{bmatrix} \sum_{j=1}^{n} \boldsymbol{c_{1j1}\dot{q}_j} & \cdots & \sum_{j=1}^{n} \boldsymbol{c_{nj1}\dot{q}_j} \\ \vdots & \ddots & \vdots \\ \sum_{j=1}^{n} \boldsymbol{c_{1jn}\dot{q}_j} & \cdots & \sum_{j=1}^{n} \boldsymbol{c_{njn}\dot{q}_j} \end{bmatrix} \qquad (\mathbf{2.6-4})$$



Figur 2.6.4-Fremgangsmåte for å samle de 216 Christoffel-symbolene i den «6x6 C-matrisen», i Euler-Lagrange-likningen i matriseformat.

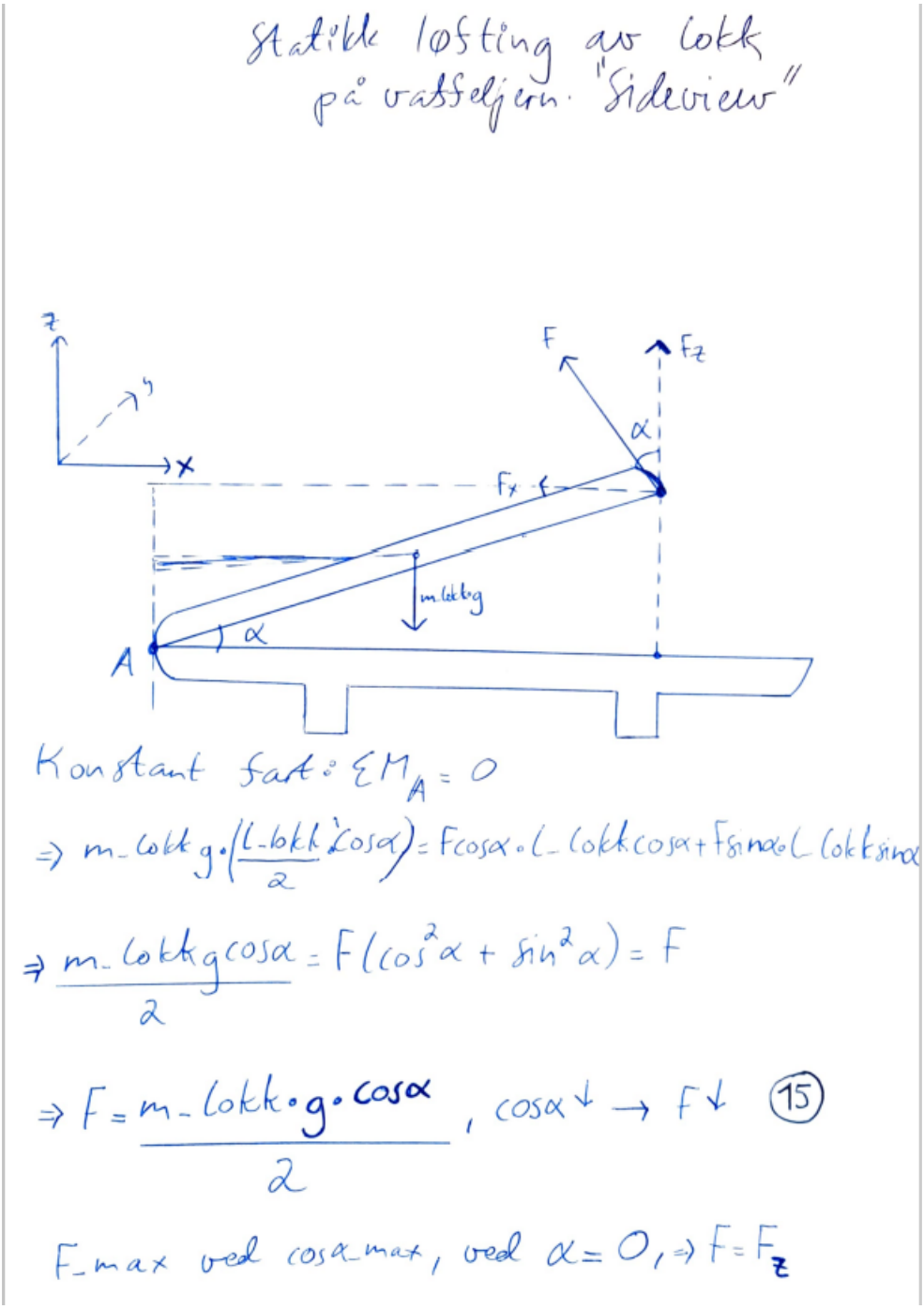
2.7 Invers kinematikk

Det var ikke sikkert robotarmen ble utsatt for en nevneverdig ytre belastning, da det var knyttet noen utfordringer til det å løfte opp lokket på vaffeljernet uten av vaffelen satt fast. En

utfordring så stor at tiden dessverre ikke strakk til for å finne en fornuftig løsning på problemet. -Men: Det kunne være det fantes en løsning på problemet i framtiden, og da måtte robotarmen i det minste være dimensjonert for en ytre belastning. Uttrykket for den ytre belastningen er vist i likning 2.7-1, med utledning figur 2.7.1 under. Som man så, så var den størst når vinkelen var minst. Kraften var da en kraft som kun virket i z-retningen. Ved toppen ville den følgelig være tilnærmet lik 0.

$$\boldsymbol{F = \frac{m_{lokk} g \cos \alpha}{2}} \qquad (\mathbf{2.7-1})$$

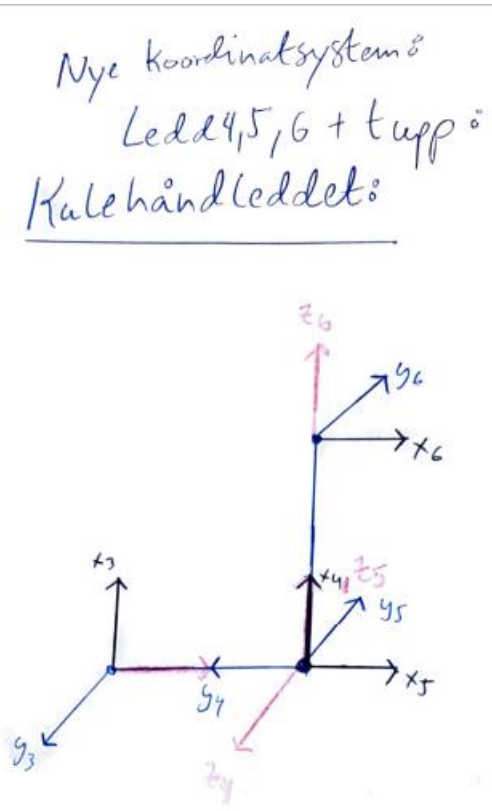
Hvor m_{lokk} er massen av lokket og α er vinkelen mellom lokket og bunnen.



Figur 2.7.1-Et uttrykk for den maksimale ytre kraften ved løfting av lokket til vaffeljernet, som bestod av en komponent i x-retning, og en i z-retning.

Vrimomentene var, som tidligere nevnt, en funksjon av leddvinklene. En måtte derfor vite den nøyaktige posisjonen til hvert ledd ved relevante arbeidsposisjoner for å kunne foreta en presis analyse. Det ble tidligere laget en fremover-kinematikk-modell, som fortalte oss hva posisjonen tuppen av robotarmen var ved ulike leddvinkler. Det man nå trengte å vite var det motsatte: «Hva er leddvinklene ved bestemte posisjoner?» For å vite dette måtte det lages en

invers-kinematikk-modell. Den ble laget etter et prinsipp hvor man delte robotarmen i to, og beregnet leddvinklene til de 3 første leddene ved hjelp av trigonometri. Såkalt «kinematisk frakobling» [20]. En tok også utgangspunkt i at det kun var de 3 første leddene som styrte posisjonen, mens de 3 siste leddene kun spilte en rolle når det kom til rotasjonen tuppen. tuppen. En forutsetning for at dette stemte, var at de 3 siste leddene sammen formet et «kulehåndledd», hvor rotasjonsaksene til hvert ledd skjærer hverandre. Studerte man de skisserte koordinataksene til robotarmen i figur 2.4.3 og 2.4.4 var det tydelig at dette ikke var tilfellet. Konfigurasjonen til den øverste motoren ble derfor endret. Nye koordinatakser for de 3 øverste leddene er vist i figur 2.7.2 under. De nye parameterne kan sees i tabell 2.7.1 under der igjen.



Figur 2.7.2-Modifiserte koordinatsakser for de 3 siste leddene, samt tuppen, for å oppfylle kravene til et «kulehåndledd»

Tabell 2.7.1- Modifiserte Denavit-Hartenberg-parametre for robotarmen.

	r	**α**	**d**	$\boldsymbol{\theta}$
1	0	90	$\frac{h_{m2}}{2}$	θ_1
2	L1	0	0	θ_2+90
3	$\frac{h_{m3}}{2}+\frac{h_m}{2}$	90	0	θ_3
4	0	-90	L2	θ_4
5	0	-90	0	θ_5-90
6	0	0	$\frac{h_{m5}}{2}+h_{m6}+l_{r6}$	θ_6

Nå som «håndleddet» til robotarmen var definert som et kulehåndledd, kunne leddvinklene til disse 3 leddene også beregnes. En fant leddvinkler for ledd 4,5 og 6, ved å først lage en rotasjonsmatrise fra basen til ledd 3, med kjente tallverdier (etter trigonometriske beregninger). Rotasjonsmatrisen fra base til tupp var allerede kjent, da det var man selv som bestemte denne konfigurasjonen. Det eneste man trengte var å finne rotasjon fra ledd 3 til 6. Det gjorde man ved å gange den inverse/transponerte rotasjonsmatrisen fra base til ledd 3 (som gir posisjon i ledd 3 sitt koordinatsystem) med rotasjonsmatrisen fra base til tupp. En sammenlignet deretter denne matrisen med tallverdier, med en matrise med variabler i form av leddvinkler 4-6, som dannet grunnlaget for utrykkene til disse leddvinklene. Utrykkene for hver leddvinkel kan ses i likning 2.7-2 til 2.7-8 under, og utledningen finnes i figur 2.7.3 til 2.7.9 under der. Det kan også legges til at man her kun antok at ledd 2 kan rotere mellom 0 og -90 grader («skulder fram/shoulder forward»), mens ledd 3 kan rotere mellom -90 og 90 grader («albue bak/elbow back» og «albue fram/elbow forward»). Dette ga derfor 2 mulige konfigurasjoner for robotarmen. Albue-fram ble satt som utgangskonfigurasjon, men om en posisjon ikke kunne nås med denne konfigurasjonen ble det forsøkt med albue-bak. Ved albue-bak-konfigurasjon var det nødvendigvis bare vinkelen til ledd 3 som var annerledes.

$$\boldsymbol{Leddvinkel\ 1}\textbf{: } \boldsymbol{\theta_1} = \mathbf{tan^{-1}}\frac{\boldsymbol{y_{hl}}}{\boldsymbol{x_{hl}}} \qquad (\mathbf{2.7-2})$$

Hvor x_{hl} *og* y_{hl} *er avstandene fra rotasjonsaksen til motor* 1 *til håndleddet*

$$\boldsymbol{Leddvinkel\ 2}: \boldsymbol{\theta_2} = \boldsymbol{\cos^{-1}\left(\frac{R1^2 + L1^2 - R2^2}{2R1L1}\right) + \tan^{-1}\left(\frac{b}{a}\right) - \frac{\pi}{2}} \qquad (\boldsymbol{2.7-3})$$

For $R1, L1$ *og* $R2$ *se figur* 2.7.4

$$a = \sqrt{x_{hl}^2 + y_{hl}^2}\ og\ b = z_{hl} - \frac{h_{m2}}{2}$$

z_{hl} *er høydeforskjellen fra rotasjonsaksen til motor* 2 *il håndleddet*

$$\boldsymbol{Albue\ ned, vinkel\ 3}: \boldsymbol{\theta_3} = \boldsymbol{\tan^{-1}\left(\frac{L2}{\frac{h_{m3}}{2} + \frac{h_{m4}}{2}}\right) + \cos^{-1}\left(\frac{R2^2 + L1^2 - R1^2}{2R2L1}\right) - \pi} \ (\boldsymbol{2.7-4})$$

For $L2$ *se figur* 2.7.4

$$\boldsymbol{Albue\ opp}: \boldsymbol{\theta_3} = \boldsymbol{\cos^{-1}\left(\frac{R2^2 + L1^2 - R1^2}{2R2L1}\right) - \tan^{-1}\left(\frac{\frac{h_{m3}}{2} + \frac{h_{m4}}{2}}{L2}\right) - \frac{\pi}{2}} \qquad (\boldsymbol{2.7-5})$$

For $L1, L2, R1$ *og* $R2$ *se figur* 2.7.6

$$\boldsymbol{\theta_4} = \boldsymbol{\tan^{-1}\left(\frac{R_{23}}{R_{13}}\right)} \qquad (\boldsymbol{2.7-6})$$

R_{23} *og* R_{13} *er elementer i rotasjonsmatrisen* R_3_6

Hvor R_3_6 *rotasjon fra ledd* 3 *til ledd* 6 (*se figur* 2.7.9)

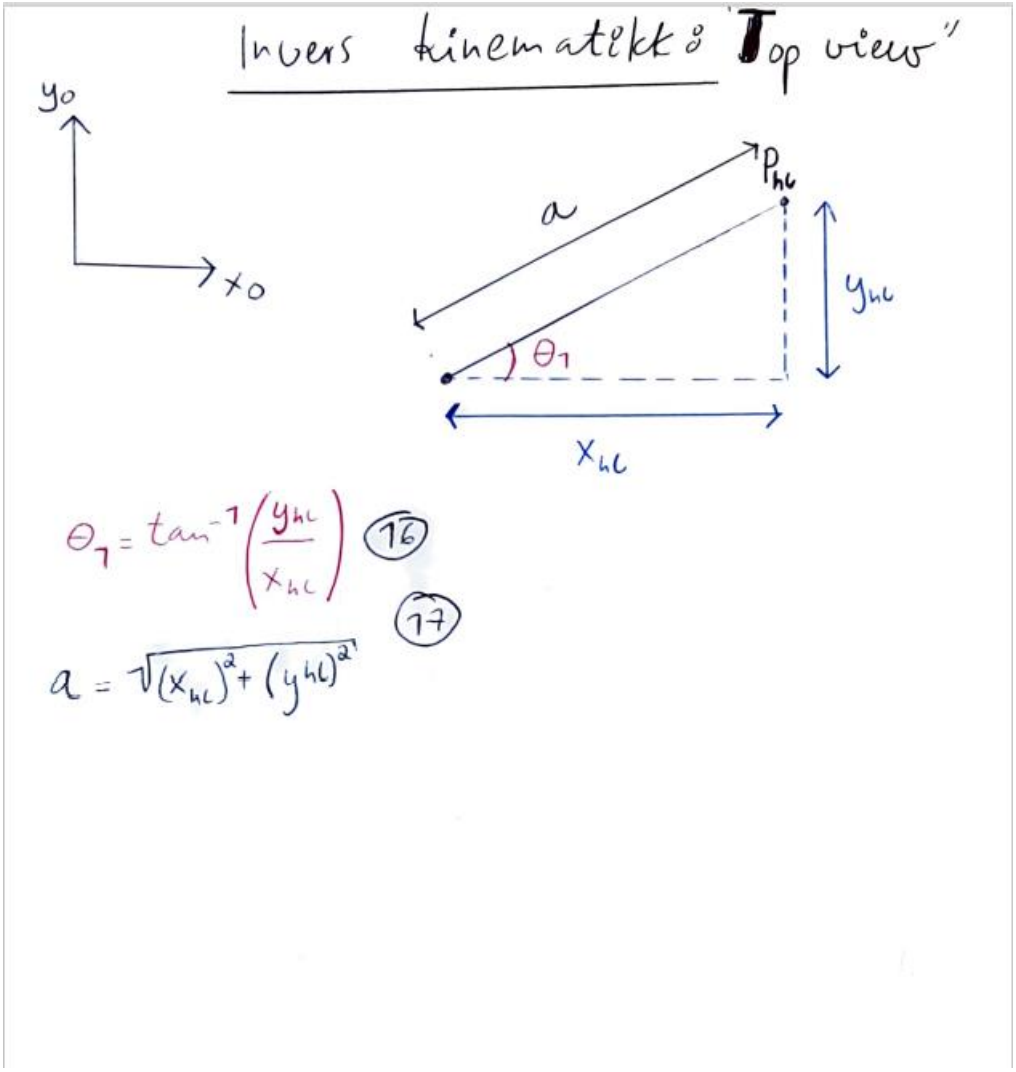
$$R\_3\_6: \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

$$\boldsymbol{\theta_5} = \boldsymbol{\cos^{-1}(R_{33}) - \frac{\pi}{2}} \qquad (\boldsymbol{2.7-7})$$

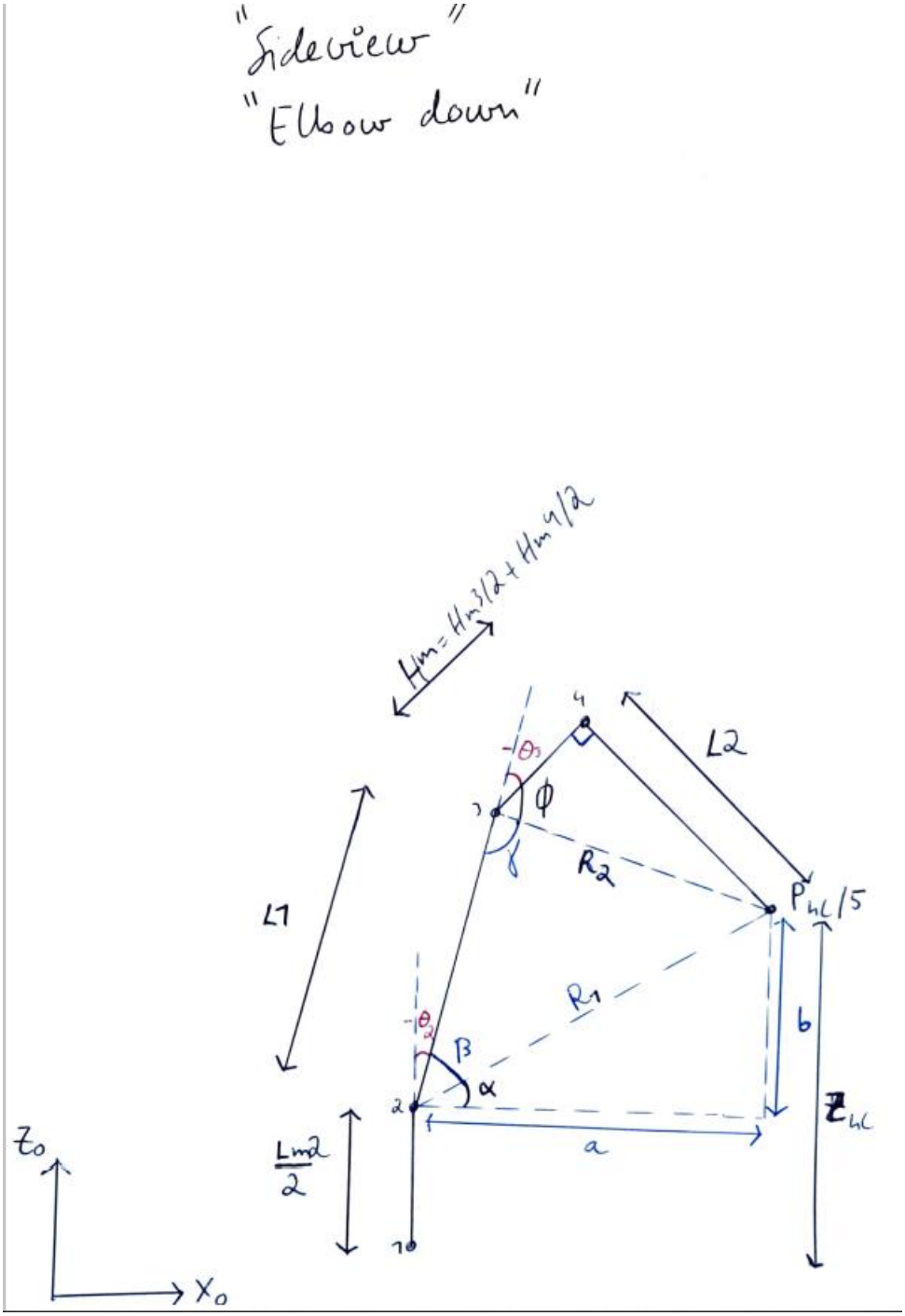
R_{33} *er et element i rotasjonsmatrisen* R_3_6

$$\boldsymbol{\theta_6} = \boldsymbol{-\tan^{-1}\left(\frac{R_{32}}{R_{31}}\right)} \qquad (\boldsymbol{2.7-8})$$

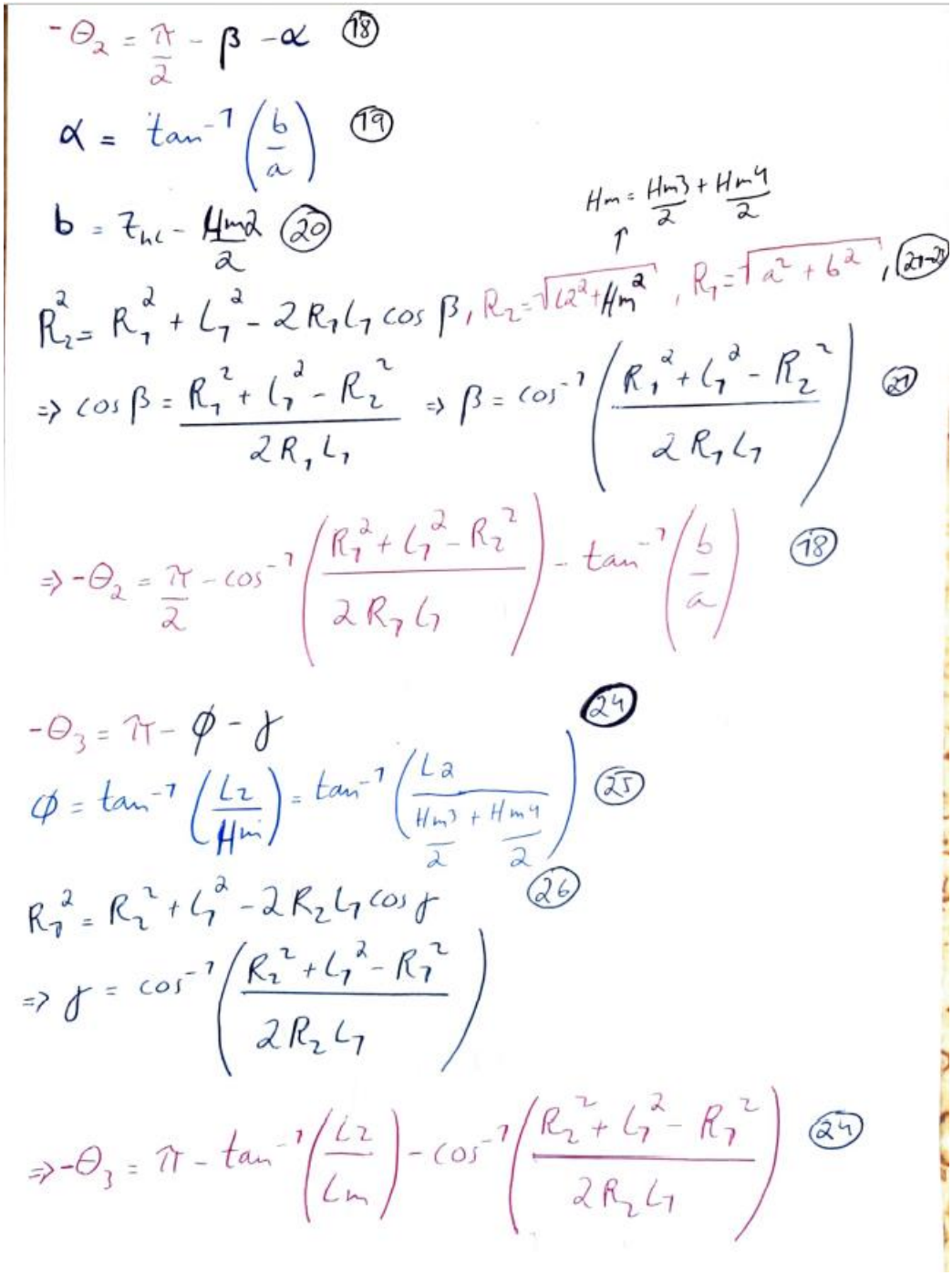
R_{32} *og* R_{31} *er elementer i rotasjonsmatrisen* R_3_6



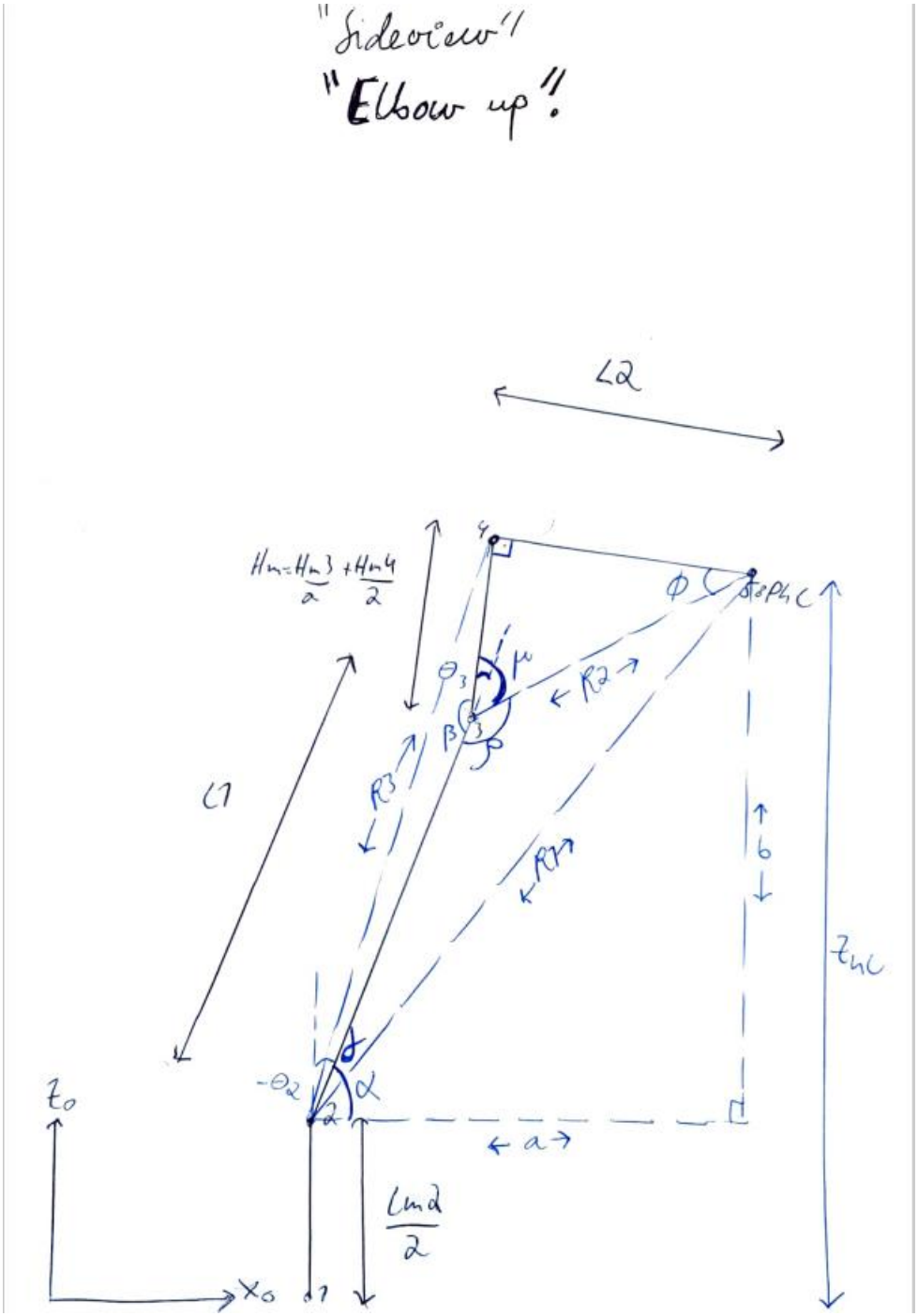
Figur 2,7.3-Et uttrykk for leddvinkel 1, som var den samme uavhengig av konfigurasjon.



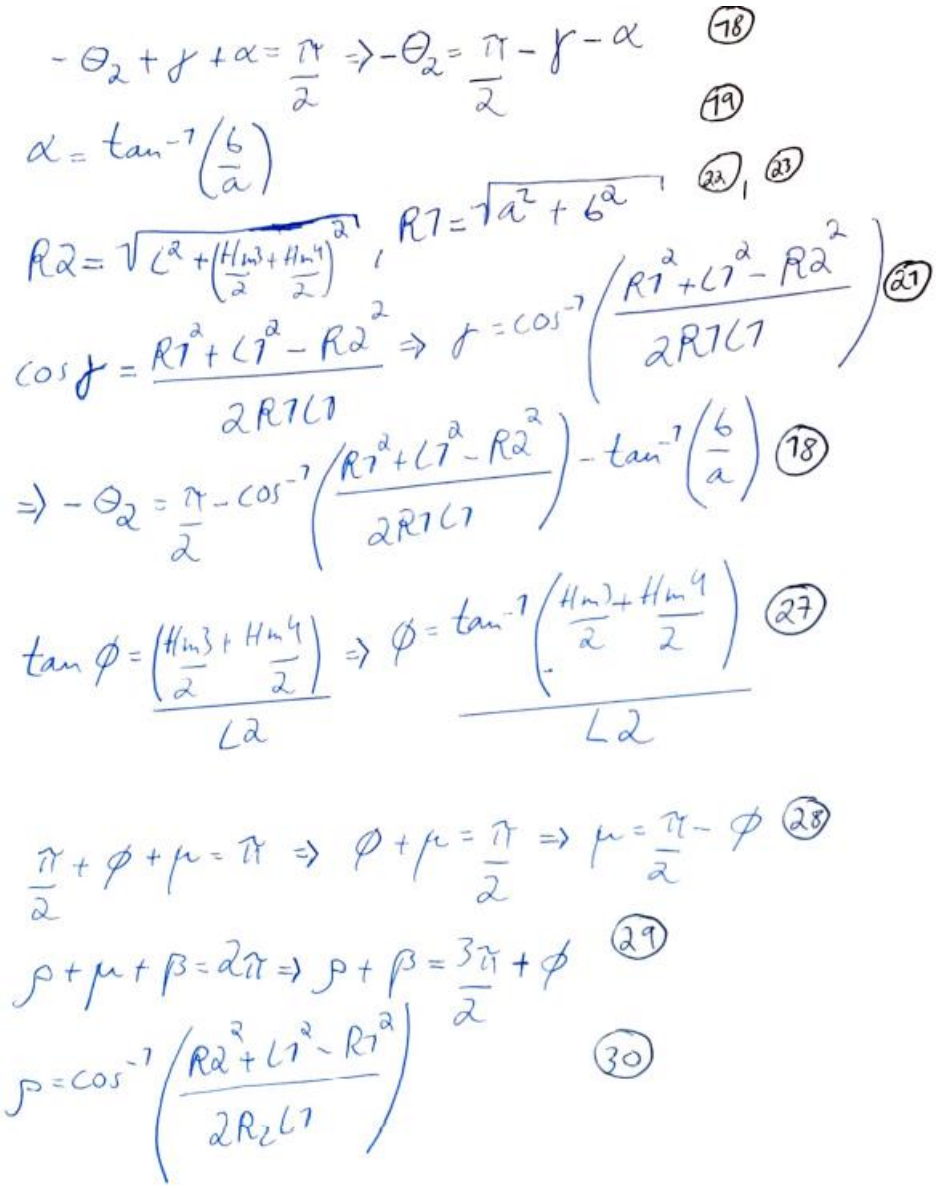
Figur 2.7.4-Trigonemetrien som dannet grunnlaget for leddvinkler 2 -og 3 i en «albue-ned»-konfigurasjon.



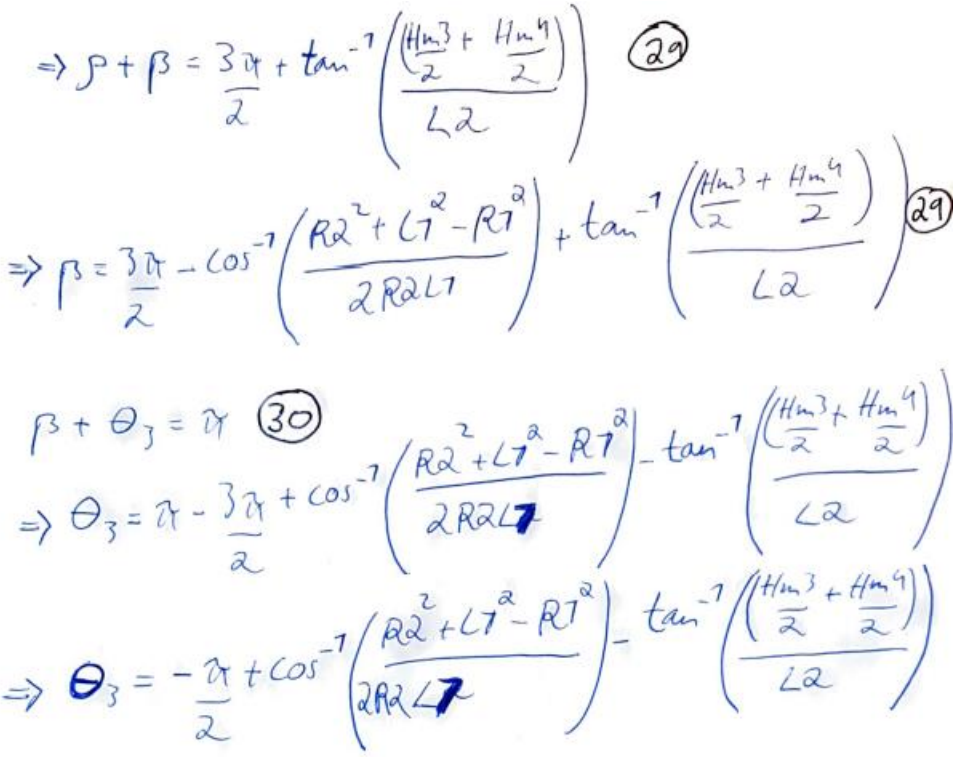
Figur 2.7.5-Utledning for leddvinkel 2 -og 3, i en «albue-ned»-konfigurasjon.



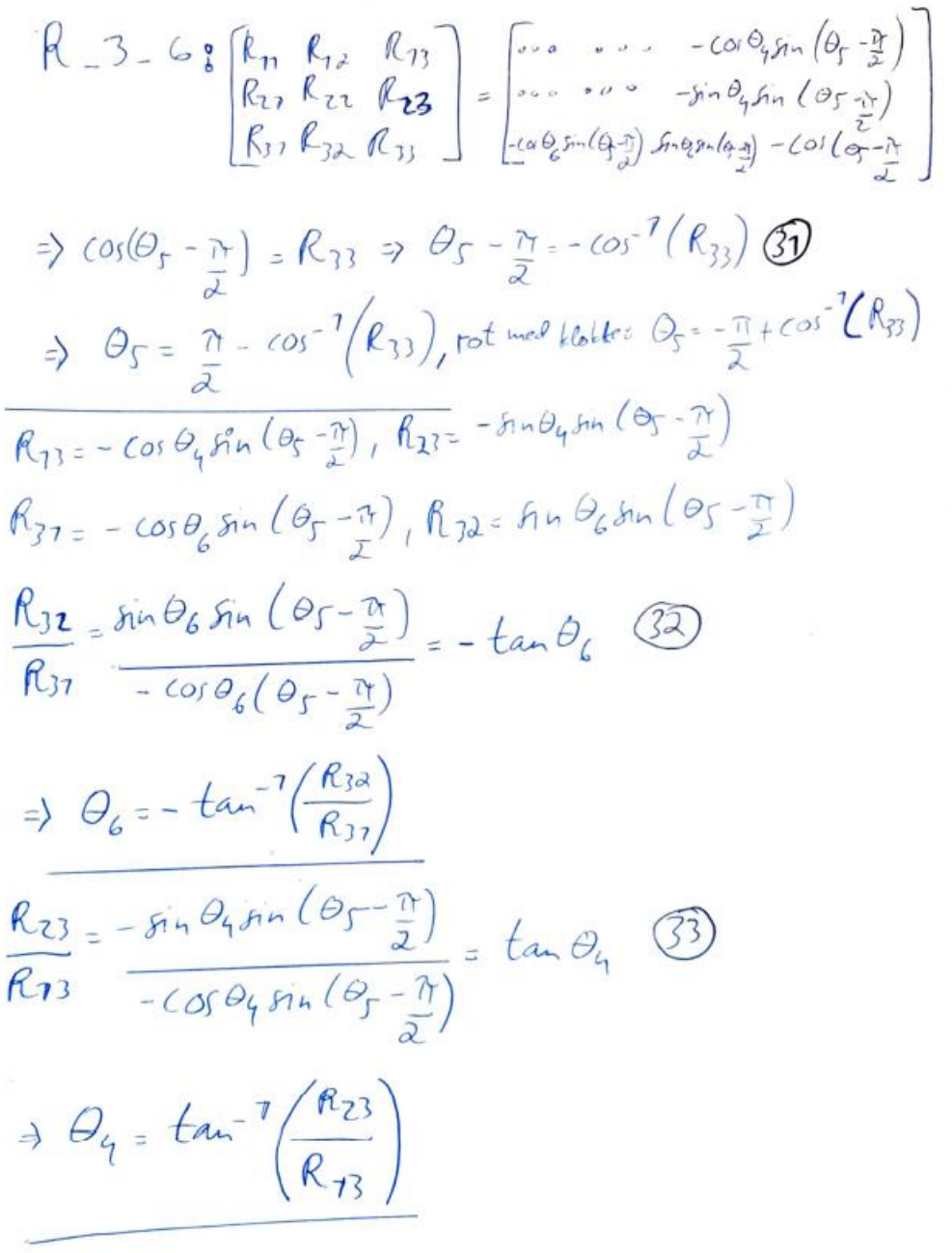
Figur 2.7.6-Trigonometrien som dannet grunnlaget for leddvinkler 2 og 3 i en «albue-opp»-konfigurasjon.



Figur 2.7.7-Utledning for leddvinkel 2 i en «albue-opp»-konfigurasjon, som følgelig var den samme som ved «albue-ned».



Figur 2.7.8-Utledning for leddvinkel 3 ved «albue-opp»-konfigurasjon.



2.7.9-Utledning for leddvinkler 4-6, som man så var en funksjon av leddvinkler 1-3.

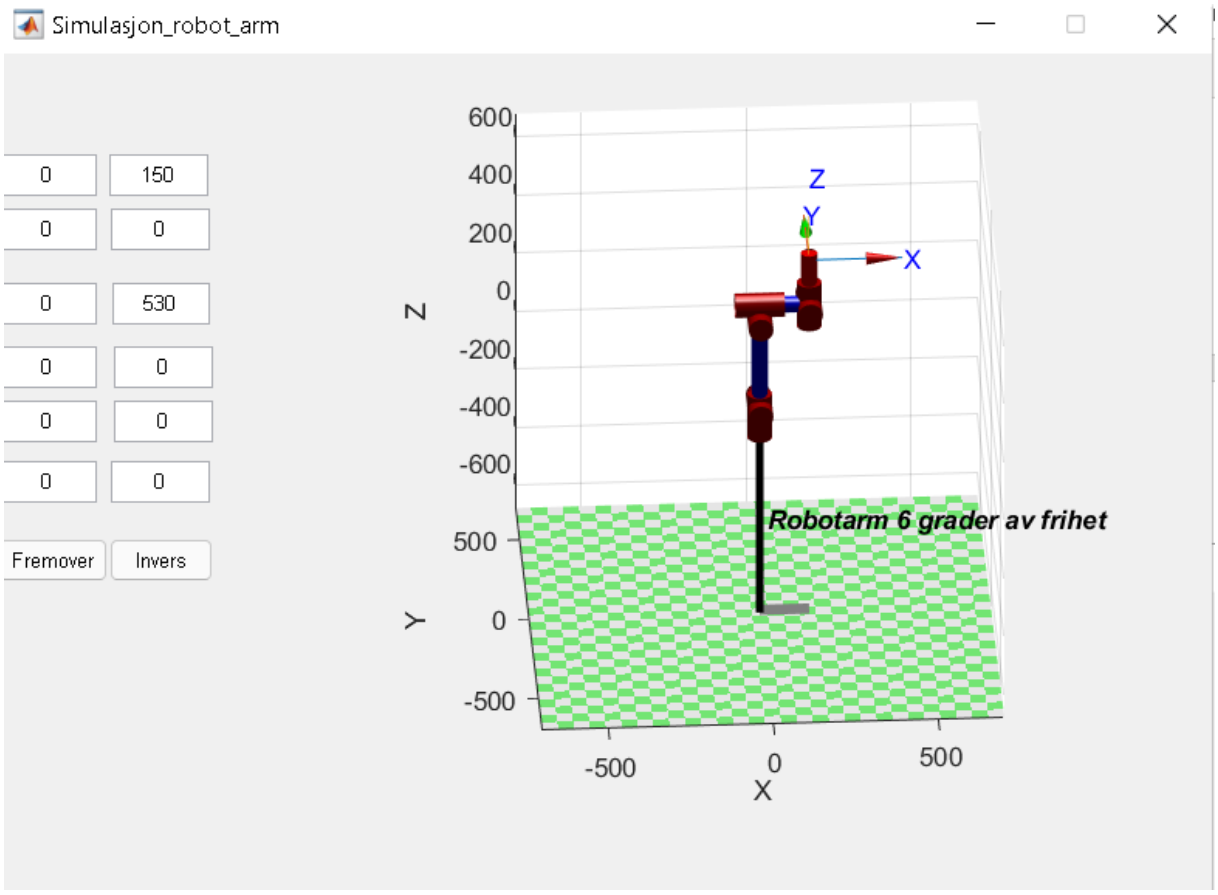
2.8 Relevante posisjoner til analyse ved hjelp av simulasjonsverktøy

Det var ikke alle posisjoner som var like relevante å studere, men kun de posisjonene som kunne klassifiseres som arbeidsposisjoner, og de posisjonene hvor det kunne tenkes at belastningen på leddene var maksimal. Det ble definert 5 slike posisjoner hvor nødvendig vrimoment ble beregnet. P.ga symmetri ble kun posisjoner ved ett vaffeljern (vaffeljernet til venstre for robotarmen, sett forfra) analysert. Så kan det legges til at det ikke ble utarbeidet et

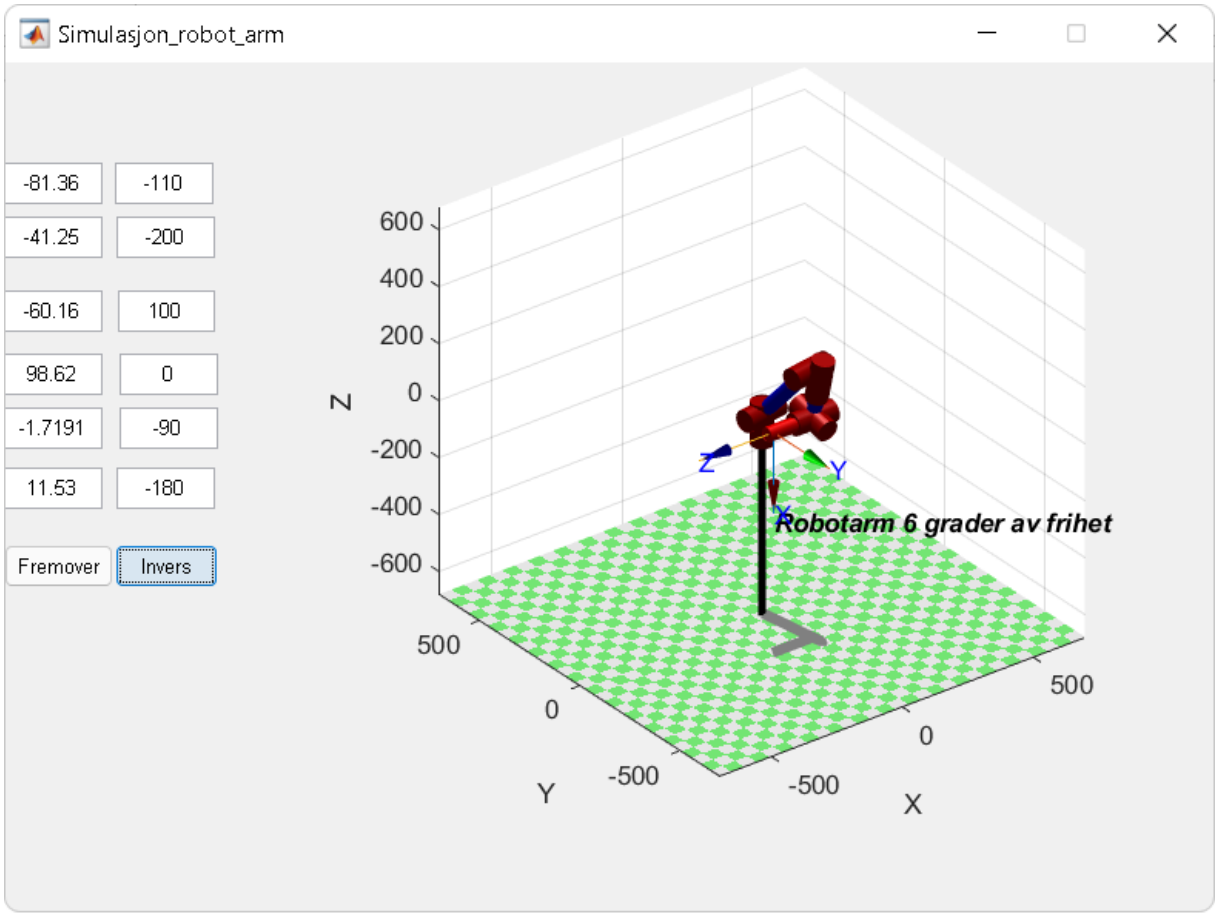
styre-program for posisjon 3-5, men som tidligere nevnt så skulle robotarmen være dimensjonert for å tåle belastningen ved alle disse 3 posisjonene.

- Posisjon 1-Påføre røre ved senter til nærmeste vaffel.
- Posisjon 2-Påføre røre ved vaffel lengst borte.
- Posisjon 3-Løfte lokket til vaffeljernet, på toppen, uten ytre belastning.
- Posisjon 4-Løfte lokket til vaffeljernet, midt i løftingen, med ytre belastning i x- og z-retning (finner krefter fra figur 2.7.1).
- Posisjon 5-Løfte lokket til vaffeljernet, i det løfteprosessen starter, med maksimal ytre belastning, kun i z-retning (figur 2.7.1).

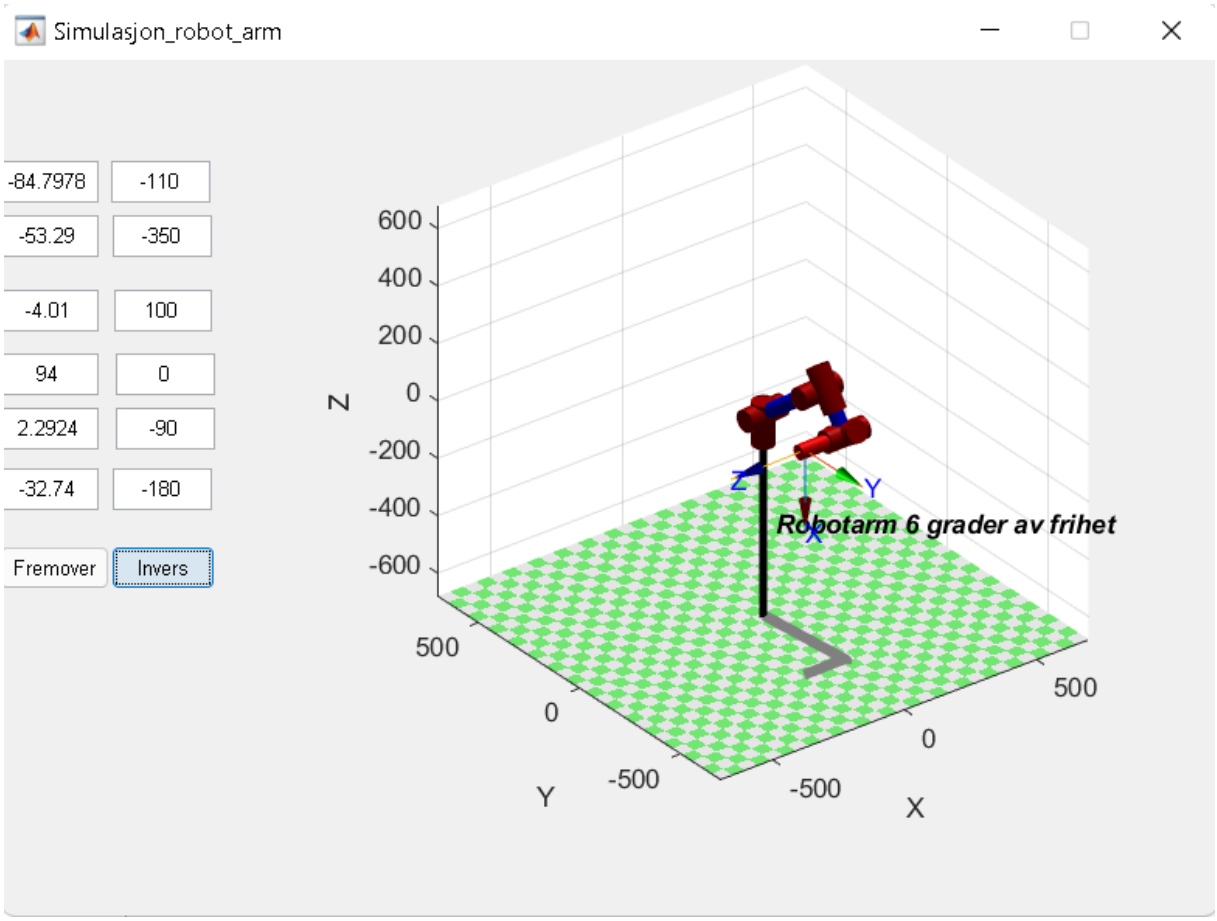
Ved hjelp av «Peter Corke Robotics Toolbox» [21] og MATLAB GUI [22] ble det laget et simulasjonsverktøy i MATLAB [vedlegg F] med utgangspunkt i beregningene man allerede hadde fra fremover- og invers kinematikk [vedlegg F, linjer 352-513], slik at samtlige posisjoner kunne visualiseres, og beregninger fra fremover- og invers kinematikk kunne valideres. Posisjoner 1-5, samt hvileposisjonen til robotarmen er vist i figur - under. Inputen til den inverse kinematikken fikk man fra de ytre målene i figur 2.1.1. Det kan også sies at vinklene man ser i figur 2.8.1 til 2.8.6 under er ved de endelige dimensjonene, og ikke ved utgangskonfigurasjonen, hvor alle motorene var de samme.



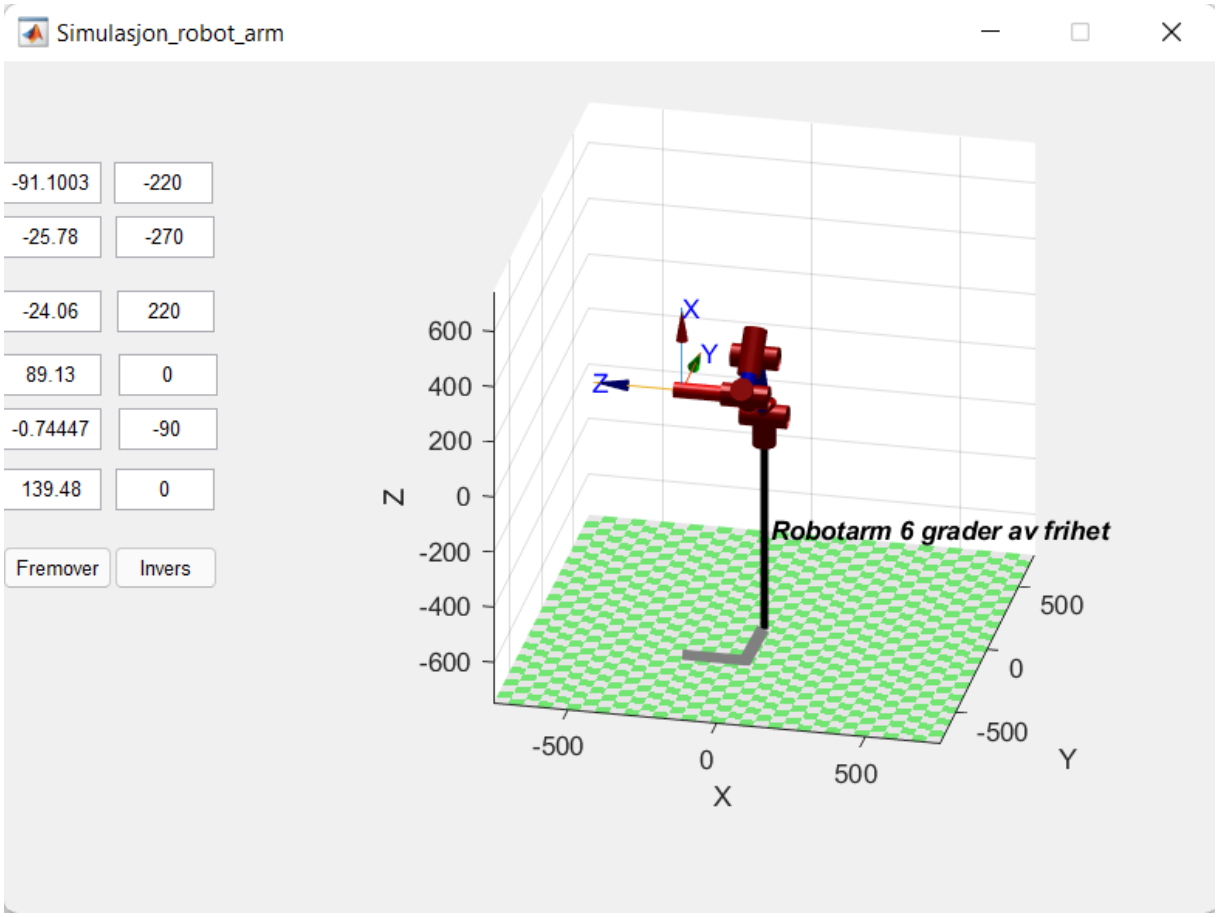
Figur 2.8.1-Leddvinkler ved hvileposisjon.



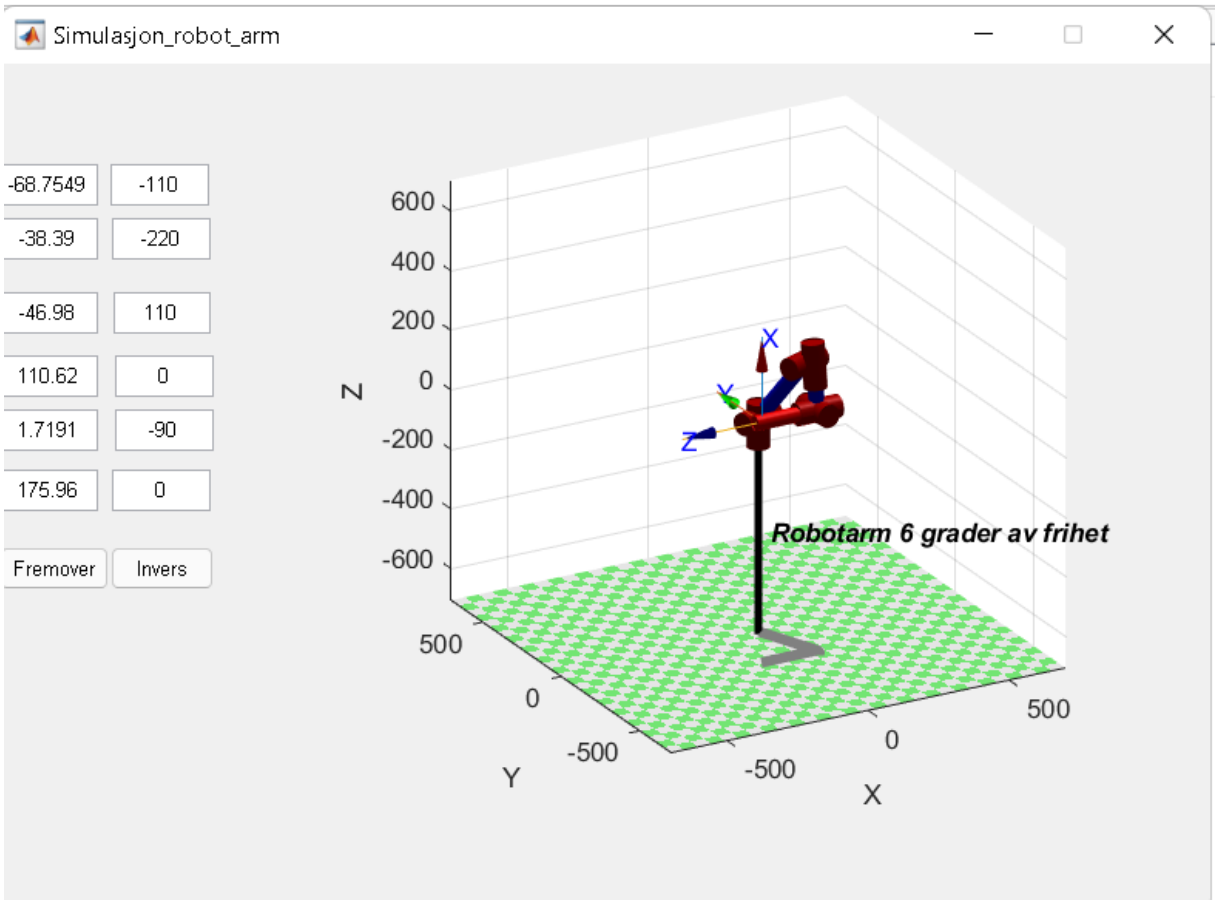
Figur 2.8.2-Leddvinkler ved posisjon 1-senter til nærmeste vaffel.



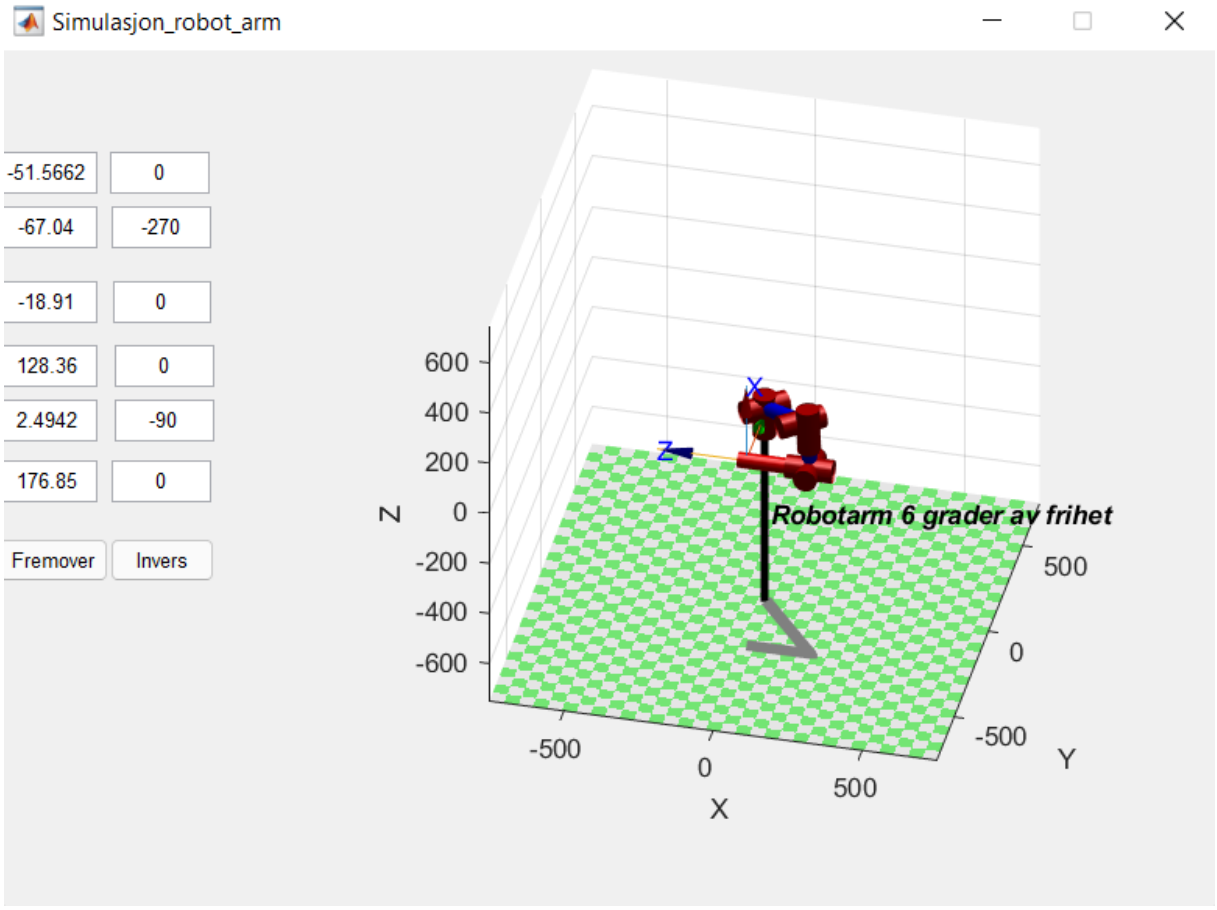
Figur 2.8.3-Leddvinkler ved posisjon 2-senter til vaffel lengst borte.



Figur 2.8.4-Leddvinkler ved posisjon 3-løfte lokk, topp. Ingen ytre belastning.



Figur 2.8.5-Posisjon 4-løfte lokk, midt på, ytre belastning i x- og z-retning.



Figur 2.8.6-Leddvinkler ved posisjon 5-løfte lokk, bunn. Med maksimal ytre belastning i z-retning.

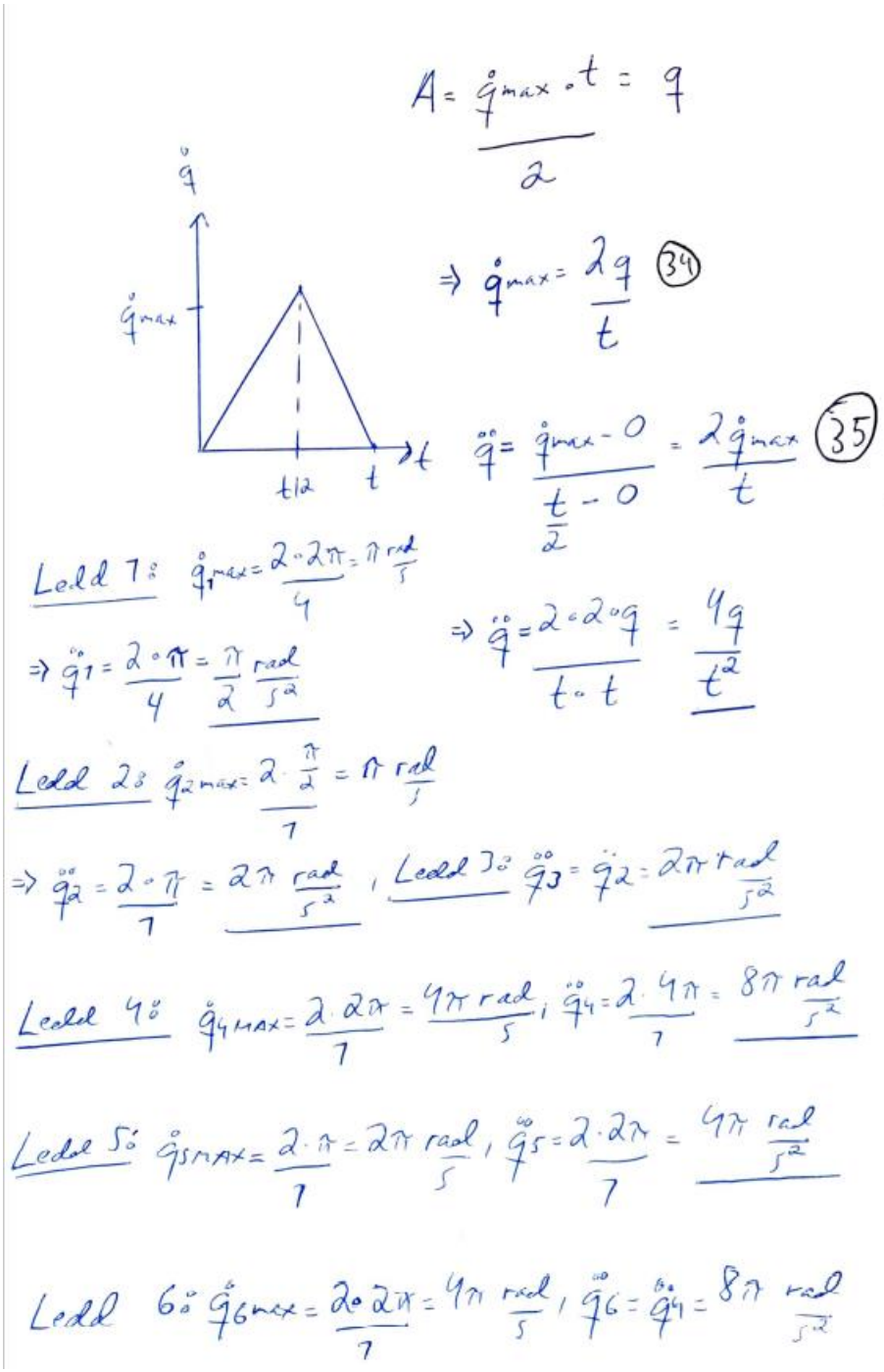
2.9 Definere maksimale ledd-hastigheter og akselerasjon.

Det siste steget før en kunne beregne de nødvendig tilførte vrimomentene var å definere maksimal ledd-hastigheter- og akselerasjon. For å finne disse måtte en bestemme hvor lang tid hvert ledd skulle bruke på en hel bevegelse. Ledd 1 kunne f.eks. rotere 360 grader, mens ledd 2 og 3 var satt til å ikke kunne rotere mer enn 90 grader om gangen. Det var selvsagt fristende å si at leddene skulle rotere i konstant hastighet, med null akselerasjon. Problemet med dette r at leddet er jo i ro på et tidspunkt, så når det da skal opp i konstant hastighet umiddelbart blir det en uendelig stor akselerasjon, som ikke er ønskelig. Det som derimot gir lavest mulig akselerasjon, i tillegg til hastigheter, er en konstant akselerasjon fra start til halvveis i bevegelsen, hvor det er en konstant deselerasjon fra midtveis i bevegelsen til slutten. Tidene per bevegelse ble satt vilkårlig, mens akselerasjoner og hastigheter ble beregnet etter likninger 2.9-1 og 2-9-2 under (utledning i figur 2.9.1). Disse parameterne ble satt inn i tabell 2.9.1 under der igjen.

$$\textbf{Formel for maksimale leddhastigheter: } \dot{q}_{max} = \frac{2q}{t} \qquad (2.9-1)$$

Hvor q er vinkel bevegd i radianer, og t er tid i sekunder.

$$\textbf{Formel for maksimale ledd - akselerasjoner: } \ddot{q}_{max} = \frac{2\dot{q}_{max}}{t} \qquad (2.9-2)$$



Figur 2.9.1-Uttrykk for maksimale ledd-hastigheter- og akselerasjoner ($\dot{q}$ og $\ddot{q}$)

Tabell 2.9.1- Maksimale ledd-hastigheter og akselerasjon.

	Tid per 1 revolusjon(s)	Maksimal hastighet ($\frac{rad}{s}$)	Maksimal akselerasjon $\left(\frac{rad}{s^2}\right)$
Ledd 1	4s per 1 revolusjon	π	$\frac{\pi}{2}$
Ledd 2	1s per ¼ rev	π	2π
Ledd 3	1s per ¼ rev	π	2π
Ledd 4	1s per 1 rev	4π	8π
Ledd 5	1s per ½ rev	2π	4π
Ledd 6	1s per 1 rev	4π	8π

2.10 -Beregninger av vrimomenter ved hjelp av Lagrange-mekanikk-

Nå som alle parameterne var på plass kunne omsider vrimomentene beregnes. Koden med fremover kinematikk ble utvidet [vedlegg G], og en benyttet seg av fremgangsmåtene og formlene som tidligere er beskrevet i kapittel 2.1 til 2.9. Mer detaljert beskrivelse av prosessen finnes i selve koden [vedlegg G]. Resultatene etter den første analysen er vist i tabell 3.3.1, under resultater.

Man så i resultatene under tabell 3.3.1 at det nødvendige vrimomentet varierte voldsomt alt etter som hvilken konfigurasjon man studerte. Det første steget ble å bestemme motor nummer 6. Etter man hadde bestemt denne motoren, endret man parametere som dimensjoner, vekt, rotortreghet m.m., og gjennomførte beregningene for de 5 resterende motorene etter modifiseringen av motor 6. Ved utvelgelse av samtlige motorer var det to kriterier som gjaldt: Sikkerhetsfaktor for nødvendig tilført vrimoment måtte være på rundt 3 eller høyere, og sikkerhetsfaktor for maksimale ytre krefter måtte være rundt 3 den også (aksiale- og radielle krefter). Man så at motor 6 hadde et maksimalt vrimoment på 3Ncm ved

posisjon 3-løfte lokk, på topp. Basert på disse kriteriene ble motoren ««Nema 11 Bipolar 1.8deg 12Ncm (17oz.in) 0.67A 6.2V 28x28x51mm 4 Wires» fra stepperonline.com første valget for motor 6 [23]. Her fikk man en sikkerhetsfaktor på 4 når det gjaldt vrimoment. Den inverse kinematikken måtte også gjennomføres med de nye parameterne. Nye vrimomenter etter modifiseringen av motor 6 er vist i resultater, under tabell 3.3.2.

En så i tabell 3.3.2, at selv etter modifisering av motor 6, som veide langt mindre enn forgjengeren, var fremdeles vrimomentene for de resterende motorene fryktelig høye. Vrimomentene var, som tidligere nevnt, en funksjon også av ledd-hastighet- og akselerasjon, så disse fant man ut ble satt i overkant høyt. Hastigheter og akselerasjoner ble senket/modifisert. Nye verdier er vist i tabell 2.10.1 under.

Tabell 2.10.1- Maksimale ledd-hastigheter og akselerasjon etter modifisering/senkning.

	Tid per 1 revolusjon(s)	Maksimal hastighet ($\frac{rad}{s}$)	Maksimal akselerasjon $\left(\frac{rad}{s^2}\right)$
Ledd 1	8s per 1 revolusjon	$\frac{\pi}{2}$	$\frac{\pi}{4}$
Ledd 2	4s per ¼ rev	$\frac{\pi}{4}$	$\frac{\pi}{8}$
Ledd 3	4s per ¼ rev	$\frac{\pi}{4}$	$\frac{\pi}{8}$
Ledd 4	4s per 1 rev	π	$\frac{\pi}{2}$
Ledd 5	2s per ½ rev	π	π
Ledd 6	1s per 1 rev	4π	8π

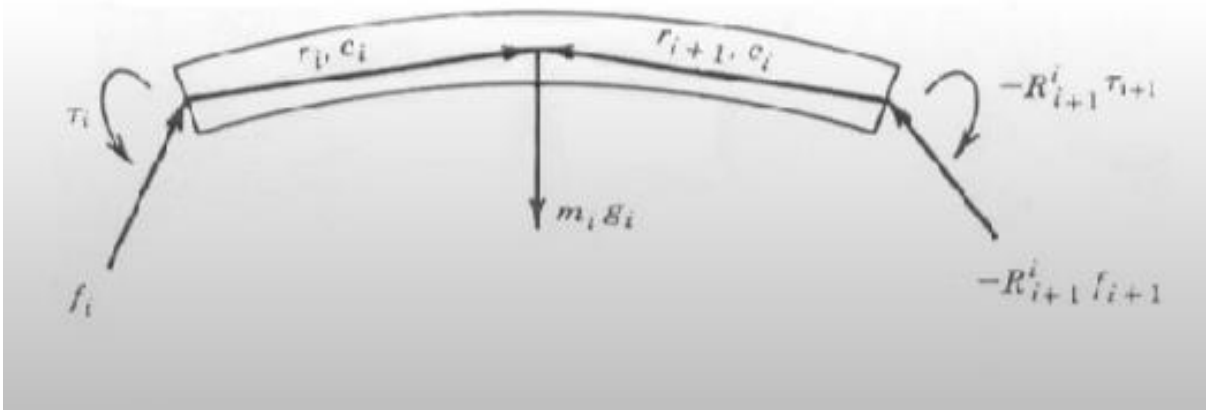
En valgte deretter ut en passende motor for ledd 5, og gjennomførte samme prosedyrer som etter utvelgelse av motor 6. Dette gjorde man helt fram til man har funnet motor 2, hvor man deretter kunne velge motor 1. Hvordan vrimomentene forandret seg utover i prosessen kan sees i tabell C1-C4, under vedlegg C. De viktigste egenskapene til hver motor etter den første analysen (Lagrange) er samlet i tabell 2.10.2 under.

Tabell 2.10.2-Viktige egenskaper for hver motor etter første analyse

	Motor 1	Motor 2	Motor 3	Motor 4	Motor 5	Motor 6
Maks bøyemoment tillatt(Nm)	25	25	25	6	6	X
Maks dreiemoment mulig å yte(Nm)	15	15	15	4	4	0.12
Masse(kg)	0.88	0.88	0.88	0.36	0.36	0.19
Strøm pr. fase(A)	1.68	1.68	1.68	1	1	0.67
Total lengde(m)	(0.039+0.0483)	(0.039+0.058)	(0.039+0.0483)	(0.034+0.0436)	(0.034+0.0436)	0.051
Maks tillatt aksialkraft (N)	X	X	X	50	50	10
Maks tillatt radielle krefter(N)	150	150	150	100	100	25
Ramme(m)	0.042*0.042	0.042*0.042	0.042*0.042	0.035*0.035	0.035*0.035	0.028*0.028
[Bremsemoment uten girboks(Nm), girfaktor, bremsemoment m. girboks.]	[0.39,30, 11.7]	[0.39,100, 39.0]	[0.39,30, 11.7]	[0.14,100, 14.0]	[0.14,100, 14.0]	[0.12,1,0.12]

2.11 Beregning av reaksjonskrefter og reaksjonsmomenter med Newton-Euler mekanikk

En hadde nå funnet 6 motorer som tilfredsstilte kravene til vrimoment basert på resultatene etter første analyse, ved Lagrange-mekanikk. Problemet var bare at rotorene/akslingene til hver motor også må tåle belastningen fra reaksjonskreftene og bøyemomentene den blir påført. Som tidligere nevnt, vil Newton-Euler-mekanikken gi disse parameterne [24]. Der Lagrange-mekanikken kun ga oss dreiemoment om z-aksen (rotasjonsaksen) til hver motor, vil Newton-Euler-mekanikken gi reaksjonskrefter i x-, y- og z-retning, samt dreiemomentene om de 3 respektive aksene. En benyttet seg av likninger 2.11-1 til 2.11-6 under for å beregne dette. Likningene tar utgangspunkt i «free-body»-diagrammet i figur 2.11.1 under [25]. Hvor en vilkårlig link blir betraktet, mellom to ledd, hvor $\sum F = ma$, og $\sum \tau = I\alpha$. Denne summen av krefter og momenter er om massesenteret til hver link. En måtte derfor først finne vinkelakselerasjon, vinkelhastighet, akselerasjon til hvert massesenter, samt akselerasjon av enden til hvert ledd, for å finne reaksjonskrefter og dreiemomenter. De fire førstnevnte parameterne fant en ved hjelp av «framlengs rekursjon» [26], mens krefter og momenter fant en ved «baklengs rekursjon» [26], da hastigheter/akselerasjoner ved basen er lik 0, og ytre krefter og momenter ved tuppen er forhåndsdefinert. Det kan også legges til at når man snakker om «interne koordinater» så er dette koordinatsystemet til det neste leddet. Så rotasjonen om rotoren til motor 1, som skjer i det globale koordinatsystemet, ble definert i koordinatsystemet til motor 2 osv. Formlene for krefter og momenter kan som nevnt leses ut av figur 2.11.1 under, mens formlene for blant annet lineær akselerasjon tar utgangspunkt i lover definert innen dynamikk for «solide kropper» [27]. Disse formlene ble ikke utledet, men beviset finner man i YouTube-videoen hvor formlene ble hentet fra [25]. En så også bort fra vekten av forbindelses-materiale i disse beregningene.



Figur 2.11.1-FBD for hver link, mellom to ledd [25], som gir utgangspunktet for likningene 2.11-1 til 2.11-6 under. Formlene for kraftvektorer og momentvektorer (likning 2.11-5 og 2.11-6) tar utgangspunkt i Newtons andre lov om massesenteret til hver link.

$$\boldsymbol{Vinkelhastighet}: \boldsymbol{\omega_i^i = R_{i-1}^i \omega_{i-1}^{i-1} + R_0^i z_{i-1}^0 \dot{q}_i^{i-1}} \qquad \boldsymbol{(2.11-1)}$$

$\boldsymbol{\omega_i^i}$ *er vinkelhastighet for hvert i ledd i* det *nåværende, interne koordinatsystemet.*

$\boldsymbol{R_{i-1}^i}$ *er rotasjonmatrise fra forrige ledd til nåværende ledd.*

$\boldsymbol{\omega_{i-1}^{i-1}}$ *er vinkelhastigheten* ***til*** det *forrige leddet, i* det *forrige leddets koordinatsystem.*

$\boldsymbol{R_0^i}$ *er rotasjon fra nåværende ledd til basen, altså inversen av rotasjon fra basen til leddet.*

$\boldsymbol{z_{i-1}^0}$ *er* $z-$*aksen til forrige ledd beskrevet i globale koordinater.*

$\boldsymbol{\dot{q}_i^{i-1}}$*er leddhastighet* ***fra*** *forrige ledd, altså leddets egen leddhastighet.*

$$\boldsymbol{Vinkelakselerasjon}: \boldsymbol{\dot{\omega}_i^i = \alpha_i^i = R_{i-1}^i \alpha_{i-1}^{i-1} + R_0^i z_{i-1}^0 \ddot{q}_i^{i-1} + \omega_i^i \times R_0^i z_{i-1}^0 \dot{q}_i^{i-1}} \boldsymbol{(2.11-2)}$$

$\boldsymbol{\dot{\omega}_i^i}$ *er vinkelakselerasjon for hvert i ledd i* det *nåværende, interne koordinatsystemet.*

Med andre ord den tidsderiverte av leddhastighetene, også benevnt $\boldsymbol{\alpha_i^i}$

$\boldsymbol{\alpha_{i-1}^{i-1}}$*er vinkelakselerasjonen* ***til*** det *forrige leddet, i* det *forrige leddets koordinatsystem.*

$\boldsymbol{\ddot{q}_i^{i-1}}$*er leddakselerasjon* ***fra*** *forrige ledd, altså leddets egen leddakselerasjon.*

$$\boldsymbol{Massesenter\ akselerasjon}: \boldsymbol{a_{s,i}^i = R_{i-1}^i a_{e,i-1}^{i-1} + \dot{\omega}_i^i \times r_{i,si}^i + \omega_i^i \times \left(\omega_i^i \times r_{i,si}^i\right)} \boldsymbol{(2.11-3)}$$

$\boldsymbol{a_{e,i-1}^{i-1}}$ *er den lineære akselerasjonen ved forbindelsen til det forrige leddet.*

$\boldsymbol{r_{i,si}^i}$ *er vektoren fra leddet til neste massesenter.*

$$\boldsymbol{Akselerasjon\ ende}: \boldsymbol{a_{e,i}^i = R_{i-1}^i a_{e,i-1}^i + \dot{\omega}_i^i \times r_{i,i+1}^i + \omega_i^i \times \left(\omega_i^i \times r_{i,i+1}^i\right)} \quad \boldsymbol{(2.11-4)}$$

$\boldsymbol{a_{e,i}^i}$ *er den lineære akselerasjonen ved forbindelsen til det neste leddet.*

$\boldsymbol{a^i_{e,i-1}}$ *er den lineære akselerasjonen ved forbindelsen til det forrige leddet.*

$\boldsymbol{r^i_{i,i+1}}$ *er vektoren fra forrige ledd til neste ledd.*

$$\boldsymbol{Reaksjonkrefter}: \boldsymbol{f_i - R^i_{i+1} f_{i+1} + m_i g_i = m_i a_{s,i}} \qquad (\mathbf{2.11-5})$$

$\boldsymbol{f_i}$ *er reaksjonskrefter i* $x-, y-, -og\ z-$ *retning ved hvert ledd.*

$\boldsymbol{R^i_{i+1}}$ *er rotasjonmatrise fra* det *neste leddet til nåværende ledd.*

$\boldsymbol{f_{i+1}}$ *er reaksjonskrefter i* $x-, y-, -og\ z-$ *retning ved* det *neste leddet.*

$\boldsymbol{m_i}$ *er massen ved massesenteret for hvert ledd.*

$\boldsymbol{g_i}$ *er tyngdekraftskomponenten i interne koordinater.*

$$\boldsymbol{Momenter}: \boldsymbol{\tau_i - R^i_{i+1}\tau_{i+1} + f_i \times r_{i,si} - R^i_{i+1} f_{i+1} \times r_{i+1,si} = I_i \alpha_i + \omega_i \times I_i \omega_i} \ (\mathbf{2.11-6})$$

$\boldsymbol{\tau_i}$ *er dreiemomenter om* $x-, y-$ *,* $-og\ z-$ *aksen ved hvert ledd.*

$\boldsymbol{\tau_{i+1}}$ *er dreiemomenter ved* det *neste leddet.*

$\boldsymbol{r^i_{i,+1,si}}$ *er vektoren fra neste ledd til massesenter, i nåværende ledds koordinatsystem.*

$\boldsymbol{I_i}$ *er treghetsmatrisen for massen ved massesenteret, som hvert ledd skal rotere.*

Før man begynte på utregningene måtte man forstå formlene, og sørge for at en gjorde de korrekte antagelsene, så her følger noen få kommentarer til hver likning (2.11-1 til 2.11-6):

- Likning 2.11-1: Vinkelhastigheten for hvert ledd i interne koordinater. For å finne denne måtte man forstå at vinkelhastigheten til hvert ledd er påvirket av leddets egen vinkelhastighet, samt vinkelhastigheten fra det forrige leddet. Hastigheten fra det forrige leddet ble ganget med rotasjonsmatrisen fra forrige ledd til nåværende ledd. Leddets egen vinkelhastighet måtte bli definert i interne koordinater, og siden denne hastigheten alltid peker i det forrige leddets z-retning ganget man z-aksen til det forrige leddet, beskrevet i globale koordinater med den inverse/transponerte

rotasjonsmatrisen fra base til ledd. Med andre ord rotasjon fra ledd til base. Vinkelhastigheten til motor 1/ledd 1 peker f.eks. i retning av ledd 1 sin z-akse, men man ønsket den definert i ledd 2 sitt koordinatsystem. En startet følgelig ved basen, som har en vinkelhastighet på 0, og jobbet seg fremover mot tuppen (framlengs rekursjon).

- Likning 2.11-2: Vinkelakselerasjonen for hvert ledd i interne koordinater. Tidsderiverte av vinkelhastigheten. Beregnet påvirkningen fra forrige ledd på samme måte som med vinkelhastigheten i likning 2.11-2. Her kan det også legges til at man tok kryssproduktet mellom vinkelhastigheten til selve leddet og ledd-hastigheten fra det forrige leddet, siden dette også skaper vinkelakselerasjon.
- Likning 2.11-3: Akselerasjon til massesentrene relativt til seg selv. Funksjon av akselerasjonen til forrige ledd relativt til seg selv, og kryssprodukter mellom vinkelakselerasjonen- og hastigheten og de respektive avstandene til massesenteret.
- Likning 2.11-4: Akselerasjon til enden av leddet relativt til seg selv.
- Likning 2.11-5: Reaksjonskrefter ved hvert ledd. 3x1 kolonne-vektor. Her måtte en huske på at tyngdekraften er en 3x1 kolonnevektor som alltid peker i negativ z-retning i det globale koordinatsystemet. Den måtte derfor ganges med de inverse rotasjonsmatrisene for hvert ledd relativt til basen, for å få den i interne koordinatsystem.
- Likning 2.11-6: Dreiemomenter. Om alle 3 akser.

Ytterligere kommentarer til beregninger står beskrevet i selve MATLAB-koden [vedlegg H]. F.eks. når det kommer til beregning av distanser i vektorformat. I tabell 2.11.1 under, ser man resultatene av en analyse ved hvileposisjon, med en ytre belastning på 10N i z-retning, for å validere koden. Man sammenlignet med hvileposisjon i figur 2.8.1, og resultatene virket svært logiske. Det var f.eks. kun vrimomenter om ledd 2, og 3, siden dette er de eneste leddene kraften har en vinkelrett distanse til, som er nøyaktig den samme distansen. Samt så fant man den desidert største normalkraften (i aksial retning) ved ledd 1, som ga mening, da dette leddet bærer vekten av alle de andre leddene.

Tabell 2.11.1-Eksempel for validering av koden: Reaksjonskrefter(N) og momenter (Nm) ved hvileposisjon med ytre belastning på 10N i global z-retning, hvor alle ledd-hastigheter- og akselerasjoner er lik 0.

	Ledd 1	Ledd 2	Ledd 3	Ledd 4	Ledd 5	Ledd 6

F_X(Radiell)	0	0	20.2	16.7	13.1	0
F_Y(Radiell)	0	23.7	0	0	0	0
F_Z(Aksial)	40.4	0	0	0	0	10.9
M_X(Bøyemoment)	0	0	0	0	0	0
M_Y(Bøyemoment)	-2.50	0	0	2.50	0	0
M_Z(Dreiemoment)	0	2.50	2.50	0	0	0

En beregnet deretter med MATLAB-kode [vedlegg H] reaksjonskrefter- og dreiemomenter ved alle de 5 posisjonene tidligere beskrevet. I stedet for å ha en tabell for krefter og momenter ved hver posisjon fant man de største absolutt-verdien blant alle de 5 posisjonene. under resultater. Fremgangsmåten finnes i koden, og det gjør også krefter og momenter ved samtlige posisjoner [vedlegg H].

I en ideell verden skulle dreiemomentene om z-aksene vært nøyaktig det samme ved alle posisjoner om man sammenlignet Lagrange-analysen og Newton-Euler-analysen . Det er de ikke. Ved nøyere ettersyn fant en flere småfeil i Lagrange-koden [vedlegg G], mens Newton-Euler-koden hadde færre av disse [vedlegg H]. Debuggingen avdekket i alle fall ingen store feil, og siden denne mekanikken var mer kjent, ble også gjennomgang av denne koden/formlene enklere. Motorutvelgelsen ble derfor til slutt basert på Newton-Euler-analysen, og man endte derfor opp med disse 6 bipolare stegmotorene, som tilfredsstilte kravene til ytre krefter og momenter:

- "Nema 23 Stepper Motor Bipolar L=56mm w/ gear-ratio 4:1 Planetary Gearbox", **motor 1**, antall: **1stk**, pris: **54.77$** [28]
- "Nema 23 Stepper Motor Bipolar L=56mm w/ gear-ratio 47:1 Planetary Gearbox", **motor 2**, antall: **1stk**, pris: **54.77$** [29]
- "Nema 14 Stepper Motor Bipolar L=33mm w/ gear-ratio 100:1 Planetary Gearbox", **motor 3, motor 4 og motor 5,** antall: **3stk**, pris: **3x 32.44$=97.32$** [30]
- «Nema 17 Bipolar 1.8deg 26Ncm 0.4A 12V 42x42x34mm 4 Wires», **motor 6**, antall: **1stk**, pris: **8.05$** [31]

Under, i tabell 2.11.2 kan man se de viktigste egenskapene til hver motor.

Tabell 2.11.2-Viktige egenskaper for hver motor etter Newton-Euler-analyse

	Motor 1	Motor 2	Motor 3	Motor 4	Motor 5	Motor 6

Maks bøyemoment tillatt(Nm) for girboks	30	60	6	6	6	4.84
Maks dreiemoment mulig å yte(Nm) for girboks	20	40	4	4	4	0.26
Masse(kg)	1.3	1.7	0.36	0.36	0.36	0.23
Strøm pr. fase(A)	2.8	2.8	1	1	1	0.4
Total lengde(m)	(0.056+0.0435)	(0.056+0.072)	(0.034+0.0436)	(0.034+0.0436)	(0.034+0.0436)	0.042
Maks tillatt aksialkraft (N)	100	100	50	50	50	60
Maks tillatt radielle krefter(N)	200	200	100	100	100	220
Ramme(m)	0.06*0.06	0.06*0.06	0.035*0.035	0.035*0.035	0.035*0.035	0.042*0.042
[Bremsemoment uten girboks(Nm), girfaktor, bremsemoment m. girboks]	[1.25,4,5.0]	[1.25,47, 58.75]	[0.14,100,14.0]	[0.14,100,14.0]	[0.14,100,14.0]	[0.26,1,0.26]

Til slutt gjennomførte man en siste Newton-Euler-analyse [vedlegg H], hvor alle parameterne ble oppdatert etter den endelige motorutvelgelsen. En fant de maksimale reaksjonskreftene og

dreiemomentene ved å sammenligne alle de 5 posisjonene. Det endelige resultatet kan man se under tabell 3.3.3 i resultater. Man sammenlignet verdiene i tabell 3.3.3 med tabell 2.11.2, og kunne slå fast at man var tilfredsstilt med sikkerhetsfaktorene for alle ledd. Kreftene og momentene i tabell ble senere brukt når det kom til dimensjonering av forbindelsene mellom motorene.

2.12 Utvelgelse av motor-drivere

Når det kom til utvelgelse av motor-drivere var kriteriet at driveren kunne tilføre motoren nok strøm. Siden motorene får strøm via to faser, måtte en gange strøm pr. fase (se tabell 2.11.2) med to for å finne nødvendig tilført strøm fra driveren. Driveren måtte selvsagt også være designet for å fungere sammen med de respektive motorene. Disse 6 driverne tilfredsstilte ovennevnte krav:

- «Digital Stepper Driver 0.3-2.2A 10-30VDC for Nema 8,11,14,16,17 Stepper Motor», driver for **motor 3,4,5, og 6**, antall: **4stk,** pris: **4x13.59$=54.36$** [32]
- «Digital Stepper Driver 1.8-5.6A 20-50VDC for Nema 23,24,34 Stepper Motor», driver for motor **1 og 2**, antall: **2stk,** pris: **2x22.66$=45.32$** [33]

2.13 Utvelgelse av spenningskilder

En spenningskilde kan ha mer enn en inngang når det kommer til positiv og negativ pol, så en trengte ikke nødvendigvis en spenningskilde pr. driver. Kravet til spenningskilden var at den opererte på driverens driftsspenning, samt at den tilførte tilstrekkelig strøm til alle motorene den var koblet opp til. Det ble derfor valgt disse 3 spenningskildene for å tilføre driverne med strøm:

- «LRS-35-24 MEAN WELL 35W 24VDC 1.5A 115/230VAC Enclosed Switching Power Supply", spenningskilde for **motor 6**, antall: **1stk,** pris: **8.15$** [34]
- «LRS-200-24 MEAN WELL 200W 24VDC 8.8A 115/230VAC Enclosed Switching Power Supply", spenningskilde **motor 3, 4 og 5**, antall: **1stk,** pris: **19.36$** [35]
- «350W 24VDC 14.6A 115/230VAC Switching Power Supply for CNC Router Kits", spenningskilde **motor 1 og 2**, antall: **1stk,** pris: **26.89$** [36]

2.14 Styring av motorer og ventil med Arduino

Motorene blir, som tidligere nevnt, styrt med drivere. Driverne har innebygde DIP-brytere som regulerer både hvor mange pulser som blir sendt ut per revolusjon, samt hvor mye strøm motoren kan trekke. Arduino Mega 2560 ble brukt som kontroller [37]. Programvaren ble laget i Arduino IDE 1.8.19 [vedlegg J]. Ventilen ble styrt ved hjelp av en MOSFET-transistor [38]. Styreprogram for motorer ble laget ved hjelp av det innebygde biblioteket «Accelstepper» [39]. I Arduino var alle parametere definert ved pulser per revolusjon, så de maksimale hastighetene og akselerasjonene fra tabell 2.10.1 måtte omformes til pulser i stedet for SI-enheter. Flere pulser gir større nøyaktighet, mens færre pulser gjør at et større dreiemoment kan ytes, så tallene under er et resultat av testing underveis. En måtte også ta høyde for gir-forhold. Verdier i form av pulser sendt ut fra kontrolleren er vist i tabell 2.14.1 under.

Tabell 2.14.1-Hastighet og akselerasjon i form av pulser.

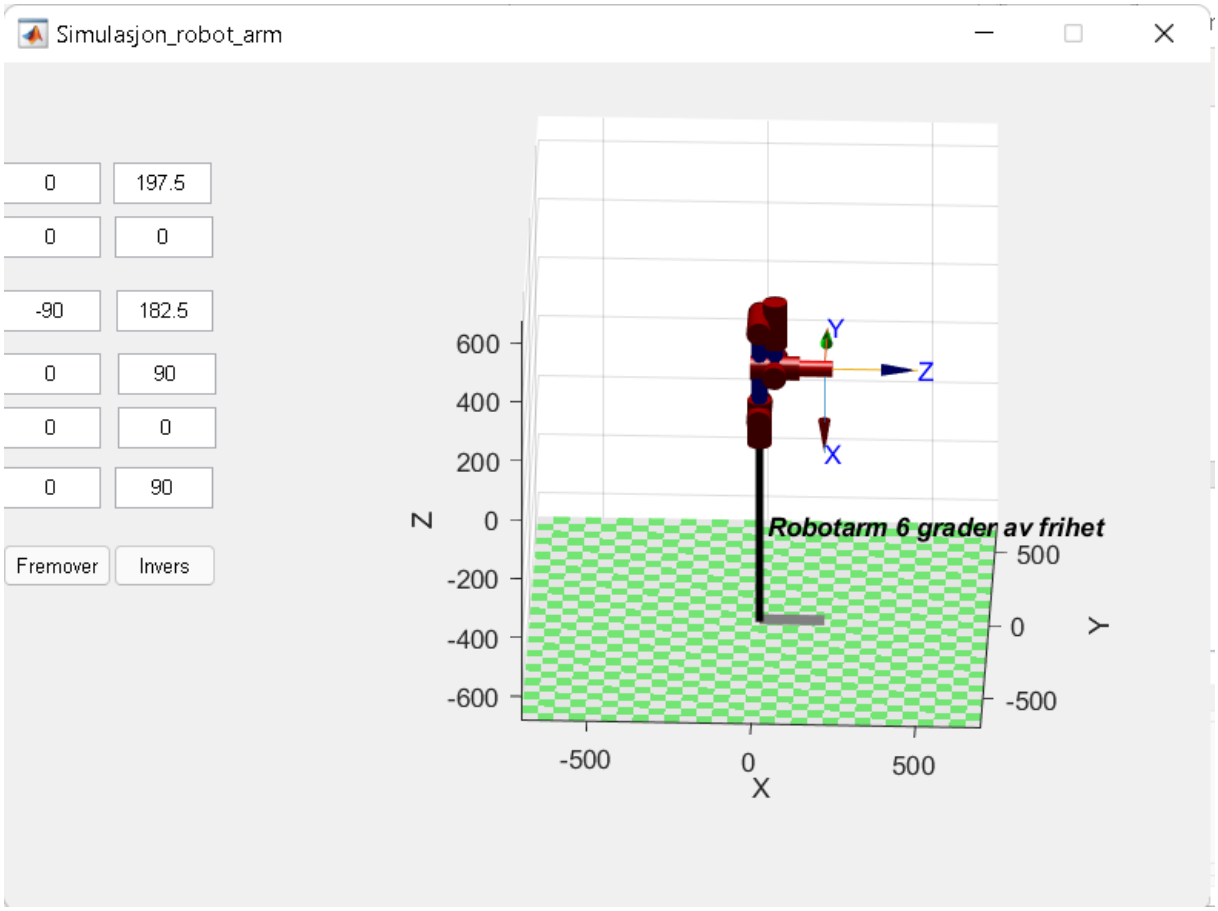
	Tid per revolusjon(s)	Maksimal hastighet (rad/s)	Maksimal akselerasjon(rad/s^2)	Girforhold	Pulser/revolusjon	Hastighet($\frac{pulser}{sekund}$)	Akselerasjon($\frac{pulser}{sekund^2}$)
Motor 1	8s per rev	$\frac{\pi}{2}$	$\frac{\pi}{4}$	4:1	6400	1600	800
Motor 2	4s per ¼ rev	$\frac{\pi}{4}$	$\frac{\pi}{8}$	47:1	18800	2350	1175
Motor 3	4s per ¼ rev	$\frac{\pi}{4}$	$\frac{\pi}{8}$	100:1	160000	20000	10000
Motor 4	4s per rev	π	$\frac{\pi}{2}$	100:1	160000	80000	40000
Motor 5	2s per ½ rev	π	π	100:1	160000	80000	80000

Motor 6	1 s per rev	4π	8π	1:1	1600	3200	6400

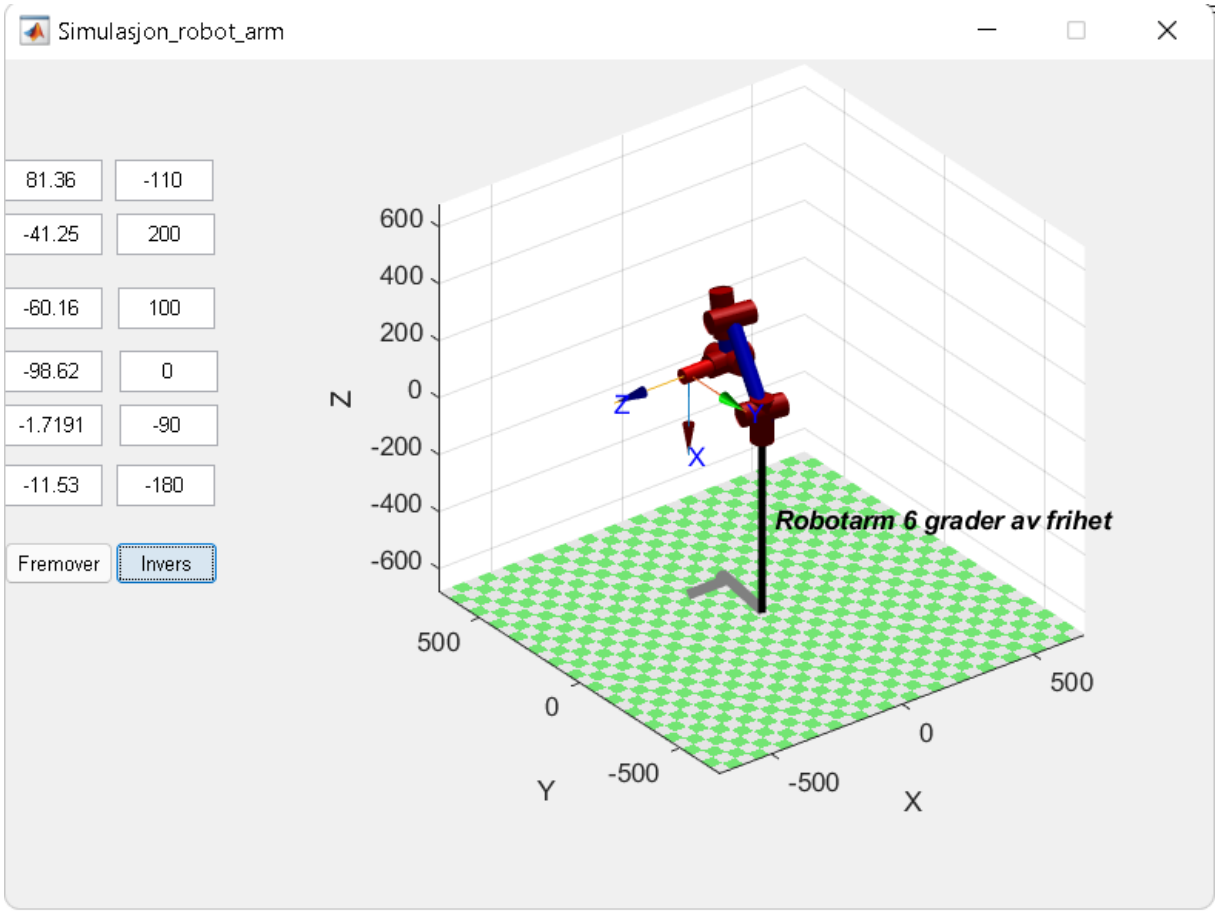
Fra tidligere hadde det blitt utarbeidet en invers-kinematikk-modell som fortalte hvilke vinkler hvert ledd måtte ha for å oppnå en bestemt posisjon. Under, i tabell 2.14.2 er disse vinklene omformet til pulser/steg, for å kunne styre motor-posisjonene med Arduino. Leddvinkler i første kolonne, antall pulser/steg for å oppnå ønsket leddvinkel, i nabokolonnen. Så måtte en også huske på å endre fortegnet på noen av stegrotasjonene, da vi tidligere hadde definert motsols rotasjons som positiv og medsols som negativ, hvor det for driverne til motor 1 og 2 var motsatt. For driverne til motor 3-6 gjaldt motsols rotasjon som positiv, men motor 3 og 5 var festet på et slik vis at motsols rotasjon for akslingen ga medsols rotasjon for leddet, så fortegnet på vinklene ble endret også her. Så kan det legges til at hvileposisjonen som er vist i figur 2.8.1 ikke var mulig å opprettholde, da motor 3 ikke hadde høyt nok hvilemoment (vrimoment uten strømtilførsel) til å holde oppe vekten av motor 4, 5 og 6. Den nye hvileposisjonen kan sees ved hjelp av simulasjonsverktøyet [vedlegg F] i figur 2.14.1 under. En måtte derfor ta høyde for at motor 3 roterte 90 grader mot klokken i tillegg til vinklene for motor 3 i figur 2.8.2 til 2.8.6 for å få robotarmen tilbake til den originale hvileposisjonen først, slik at den inverse kinematikken stemte. Så kan også leddvinkler ved påføring av røre til høyre side for robotarmen ses i figur 2.14.2 og 2.14.3 under der igjen.

Tabell 2.14.2-Leddvinkler/antall pulser(steg) for å oppnå de ønskede posisjonene

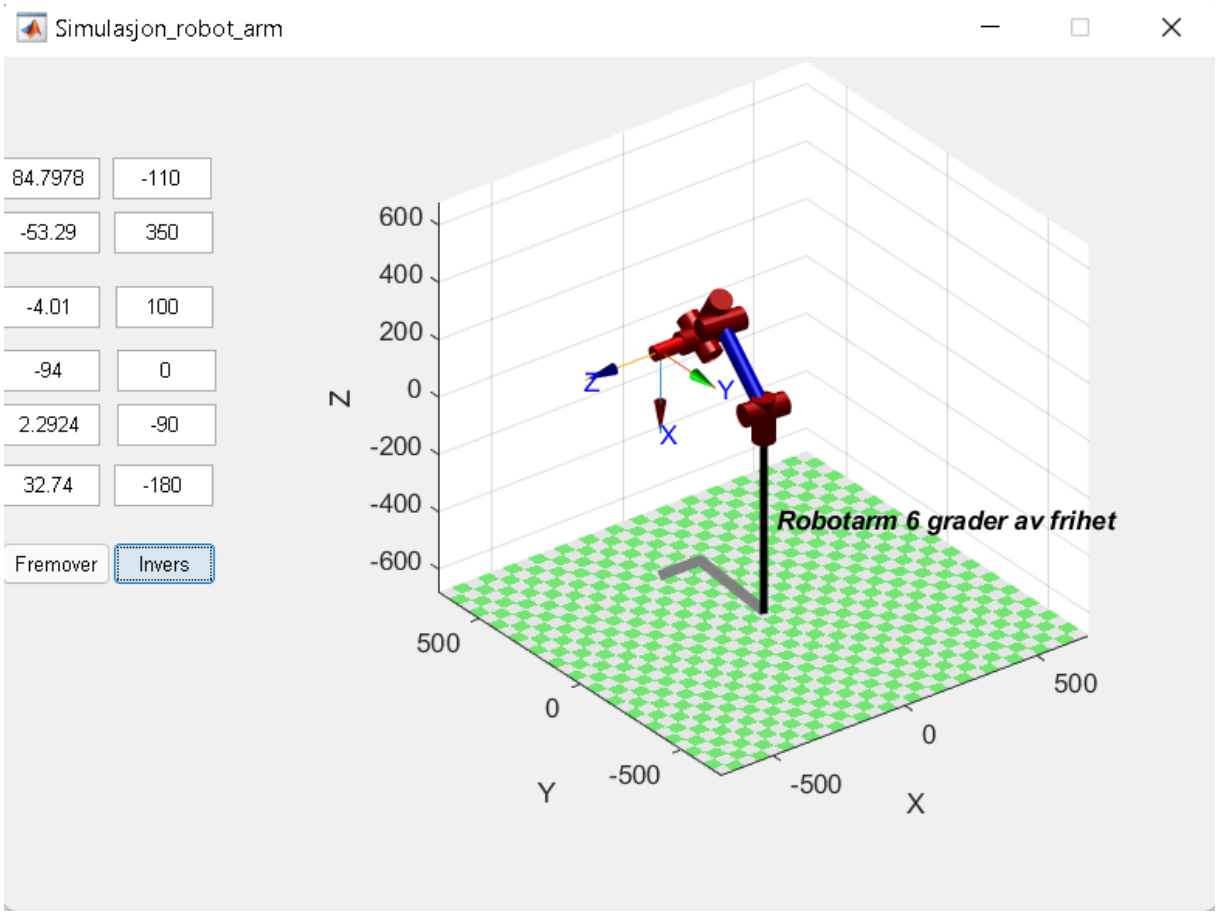
	Påføre røre vaffeljern venstre, nærmest.		Påføre røre vaffeljern venstre, lengst borte.		Påføre røre vaffeljern høyre, nærmest.		Påføre røre vaffeljern venstre, lengst borte.	
Ledd 1	-81°	1440	-85°	1511	81°	-1440	85°	-1511
Ledd 2	-41°	2141	-53°	2768	-41°	2141	-53°	2768
Ledd 3	90°-60°=30°	-13333	90°-4°=86°	-38222	90°-60°=30°	-13333	90°-4°=86°	-38222
Ledd 4	99°	44000	94°	41778	-99°	-44000	-94°	-41778
Ledd 5	-2°	889	2°	-889	-2°	889	2°	-889
Ledd 6	12°	53	-32°	-142	-12°	-53	32°	142



Figur 2.14.1-Ny hvileposisjon for robotarm, hvor ledd 3 er rotert 90 grader med klokken relativt til hvileposisjonen i figur 2.8.1.



Figur 2.14.2-Påføre røre senter vaffeljern høyre side, nærmest.



Figur 2.14.3-Påføre røre senter vaffeljern høyre side, lengst borte.

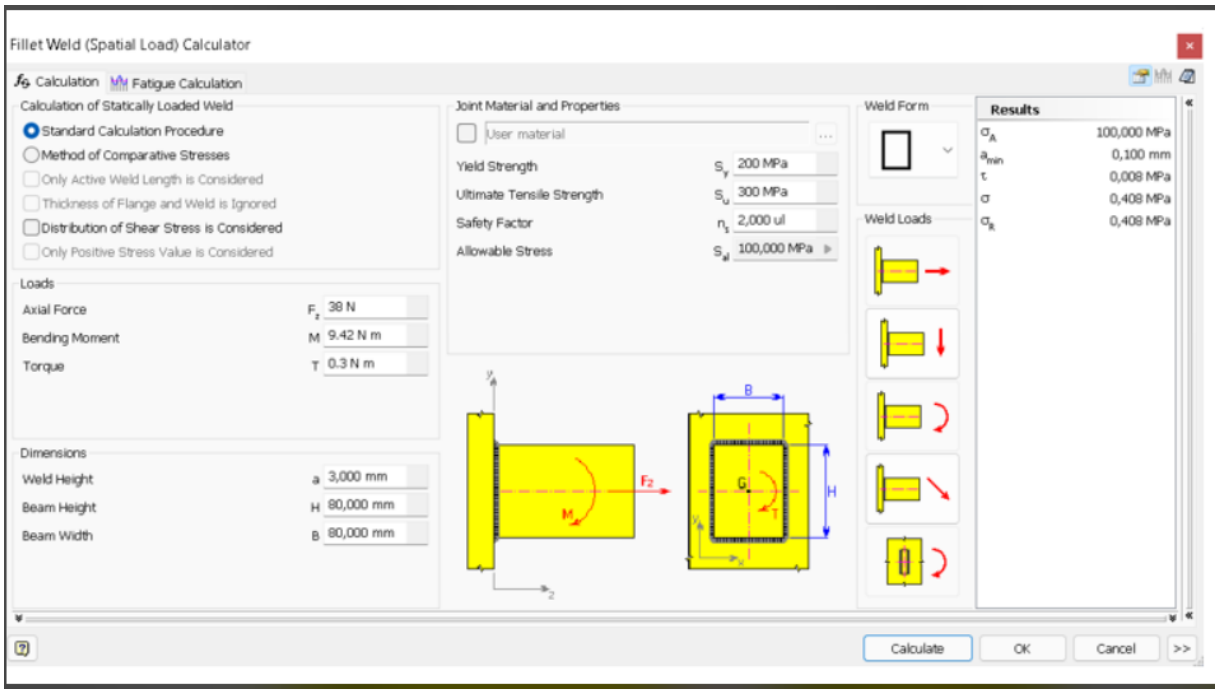
2.15 Dimensjonering med hensyn til flyting

Mesteparten av dimensjoneringen foregikk ved hjelp av kompendiet i faget maskinkonstruksjon, MSK210 [40], således også dimensjoneringen mht. flyting. All dimensjonering foregikk ved **kjente** dimensjoner. Det ble funnet deler i avfallshaugen på verkstedet, og så gjennomførte man beregninger, og fant ut om sikkerhetsfaktoren var tilstrekkelig. Krefter og momenter i tabell 3.3.3 ble brukt for å definere ytre belastninger ved de ulike leddene. Disse kreftene og momentene var allerede i lokale koordinater, så en trengte ikke å omforme dem før man gjennomførte beregningene. Det kan også legges til at man hele tiden så bort fra skjærspenning pga. bøyning (tverrgående skjærspenning). Dette fordi denne spenningen er lik 0 ved overflaten, hvor de andre spenningene er maksimale [41]. Det var flere ulike forbindelser/deler som ble dimensjonert mht. flyting. Her følger kort en liten beskrivelse rundt prosedyren av dimensjoneringen ved de ulike typene:

- Enkeltdeler: Fant først normalpenning pga. aksialkraft, skjærspenning pga. skjærkraft, normalspenning pga. bøyemoment og skjærspenning pga. torsjon ved hjelp av

likninger 2-1 og 2-2 i kompendiet [40, s. 14]. En beregnet deretter von Mises-spenning/ekvivalentspenningen med likning 2-17 [40, s. 24]. Eventuelle spenningskonsentrasjoner ble funnet ved hjelp av figur 2-25[40, s. 55]. Beregnet sikkerhetsfaktor ved å dele flytegrensen på ekvivalentspenningen. Minimum sikkerhetsfaktor ble satt til 3 for enkeltdeler.

- Skrueforbindelser: Fant ekvivalentspenning pga. ytre belastning på samme måte som ved enkeltdeler. Beregnet kraftfordelingsfaktoren ved hjelp av likning 8-3[40, s. 169], hvor skruen- og underlagets stivhet ble funnet med likning 8-4[40, s. 170] og 8-5[40, s. 171]. Fant ekvivalentspenning pga. forspenning med likning 8-13 [40, s. 176], hvor normalspenning -og skjærspenning pga. forspenning ble funnet med likning 8-11 og 8-12 [40, s. 176]. Beregnet sikkerhetsfaktor ved å dele flytegrensen på ekvivalentspenningen pga. forspenning, i tillegg til ekvivalentspenning pga. ytre belastning. Minimum sikkerhetsfaktor ble satt til 1.5 for skrueforbindelser.
- Sveiseforbindelser: Sveiseberegninger ble gjort i «Weld Calculator» i Autodesk Inventor [1]. Minimum sikkerhetsfaktor ble satt til 3 for sveiseforbindelser. Eksempel på beregning vist i figur 2.15.1 under. Man satte inn krefter og momenter fra tabell 3.3.3, definerte høyden på sveisen, samt formen på sveisen (lengde og bredde). Man fikk deretter opp en ekvivalentspenning. Man delte flytegrense på spenningen for å få en sikkerhetsfaktor. Var denne over 100, benevnte man den >100.
- Krympeforbindelser: For at brakettene som var festet på akslingene til motor 2, 3 og 5 skulle kunne rotere var det nødvendig å feste et kulelager på motsatt side av akslingen. På denne måten ville kulelageret ta halvparten av reaksjonskreftene, samtidig som rotasjon av braketten var mulig. Kulelageret ble festet til hver motorholder i en klemforbindelse mellom en skrue og en mutter. Kulelageret ble festet til braketten ved hjelp av en krympeforbindelse, der hullet var ørlite mindre enn den ytre diameteren til kulelageret. På grunn av dette ble det dannet et kontakttrykk på innsiden av hullet som holdt kulelageret på plass. På grunn av dette kontakttrykket, oppstår krympespenninger. Kontakttrykket måtte være stort nok til at det maksimale vrimomentet fra hver motor kunne bli overført uten at det førte til glidning, samtidig som spenningen i hullet ble for høy. Aksialkraft og vrimoment var allerede kjent, det samme var diameterne til kulelageret. Formler 8-19 til 8-30 i kompendiet [40, s. 191-193, 33] ble benyttet til å beregne en passende hullstørrelse. Minimum sikkerhetsfaktor ble satt til 3 for krympespenninger.



Figur 2.15.1-Eksempel på sveiseberegning i Autodesk Inventor [1].

2.16 Dimensjonering med hensyn til knekking

I utgangspunktet var det ikke planlagt dimensjonering mot knekking, men fordi lengden på brakettene ble så lange, bestemte man seg for å dimensjonere brakettene mht. knekking. Fra starten av hadde man f.eks. 2mm tykkelse på brakettene, fordi sikkerhetsfaktoren mot flyting var tilfredsstillende. Det viste seg dog at sikkerhetsfaktor mot knekking var under 1, så det ble derfor bestemt at tykkelsen måtte være 3mm. Prosedyren ved dimensjoneringen så slik ut:

- Euler-spenningen ble beregnet etter likning 2-29 [40, s. 33].
- Brakettene ble betraktet som Euler-tilfelle II [40, figur 2-10].
- Beregnet knekkespenning om den «kritiske» aksen (hvor tykkelsen blir betraktet som høyden ved beregning av treghetsmoment).
- Delte Euler-spenning på knekkespenning for å finne sikkerhetsfaktor.
- Minimum sikkerhetsfaktor mot knekking ble satt til 3.

2.17 Dimensjonering med hensyn til glidning

Ved dimensjonering mht. glidning ble to metoder brukt:

- Metode 1: Sikkerhetsfaktor forhåndsdefinert. Styrte sikkerhetsfaktoren med tilsetningsmomentet. Det vil si man beregnet hvor mange grader bolten skulle roteres

etter kontakt, for å få ønsket forspenning og friksjonskraft mellom skrue og del. Denne metoden ble benyttet ved settskrueforbindelser.

- Metode 2: Ved krympeforbindelser brukte man likninger 8-19 til 8-22 [40, s. 192]. Resulterende skjærspenning ble sammenlignet med kontakttrykket ganget med friksjonskoeffisienten. Dette forholdet ble definert som sikkerhetsfaktoren.

2.18 Dimensjonering med hensyn til deformasjoner

Konstruksjonen ble bygget opp av tynne plater, på 2- og 3mm. Det var derfor viktig at deformasjonene ikke ble for store, slik at skjevheter oppsto. Maksimal deformasjon ble satt til 3%. En studerte platene med minst tykkelse, og størst ytre belastning. Beregningene foregikk slik:

- Maksimal ytre normalkraft ble beregnet, ved å dele ytre ekvivalentspenning på spenningsarealet.
- Kraftfordelingsfaktor og stivhet til materiale ble beregnet.
- Total normalkraft: Adderte forspenningskraft og ytre kraft (multiplisert med (1-kraftfordelingsfaktor).
- Fant deformasjon ved å dele forspenningskraft på stivheten til materialet.
- Sikkerhetsfaktor ble funnet ved å dele tillatt deformasjon på faktisk deformasjon.

2.19 Materialvalg og konstruksjon av robotarmen på verkstedet

En kan først legge til at man ikke dimensjonerte noen av delene mht. utmatting da robotarmen maksimal skulle tas i bruk rundt en halvtime en gang i uken, og rotasjonshastighetene robotarmen skulle operere med var heller ikke særlig høye (60RPM og lavere).

Man startet i Autodesk Inventor [1] med å designe delene til roboten som skulle bære de 6 stegmotorene. Arbeidstegninger for samtlige deler finnes i vedlegg B. Planen var å bruke aluminium som hovedmateriale, siden det er relativt lett, billig, har god styrke og er lett tilgjengelig. Man prøvde å knekke noen plater av aluminium i plate-knekken. Dette resulterte i at materialet sprakk hvor knekken var. Man endte derfor opp med 316L-plater/rustfritt stål i stedet for. Dette medførte høyere vekt, men pga. økt seighet, strekkfasthet og flytegrense (200MPa) [42] kunne man gå ned i tykkelse på platene. Kostnadene gikk heller ikke opp, da dette materialet også kunne finnes i avfallshaugen.

Hydraulisk plateklippe-maskin og vinkelsliper ble brukt for å kappe til platene på mål, og for å bøye platene ble det brukt plate-knekke. For avrunding og finere detaljer brukte man båndsliper. Hull ble boret i søylebor. Akslinger ble dreiet ned i en dreiebenk. Gjenging gjorde man for hånd. Alt dette på UiS sitt verksted. Noen av brakettene/platene er sveiset sammen med MIG-sveising. De andre forbindelsene ble satt sammen med bolter og muttere, i fasthetsklasse 8.8 (flytegrense 640Mpa) [40, s. 168]. Sveiseforbindelse ble kun brukt på det som kunne betraktes som enkeltdeler, og hvor bolteforbindelser ikke var praktisk mulig. Bolteforbindelser ble brukt på resten, for å enkelt kunne demontere roboten. Eksternt så ble det kjøpt inn 3 kulelager:

- SKF 6000, 1stk, festet til brakett for motor 2 [43].
- SKF 626, 2stk, festet til brakett for motor 3 og 5 [44].

Hovedtanken bak kulelagrene var at de hadde en indre diameter som kunne tilpasses en metrisk skrue, og en ytre diameter som ikke var for stor i forhold til braketten den skulle krympes på. En så helt bort statisk- og dynamisk lagerbelastning, samt levetid, da kreftene vi opererte med maksimalt var rett i overkant av 40N (se tabell 3.3.3), og alle kulelagre vi så på opererte med kN. Levetiden så man bort fra, av samme grunn som man ikke dimensjonerte mht. utmatting (se første avsnitt i dette kapittelet).

2.20 Kobling av det elektriske

Det ble koblet et grenuttak med vippe-bryter til 230V-uttak i veggen. Spenningskilder ble koblet opp til grenuttak ved ledninger m. jord, nøytralleder og faseleder [45]. Spenningskilder ble koblet til drivere. Drivere ble koblet til de to motor-fasene. Drivere ble også koblet opp til kontrolleren. Det ble montert en startknapp, en stoppknapp, og en knapp som åpner ventilen på et koblingsbrett. Det ble montert tre LED-lamper, en for hver knapp, med hver sin motstand. Alt utenom koblingsbrettet med knappene og lampene ble fastmontert i et brukt sikringsskap. Det ble boret hull i sikringsskapet for ledninger mellom motorer/drivere og spenningskilde/pumpe og ventil. Vippe-bryter på grenuttaket fungerte som nødstopp. Selv om det tok stor plass så var det en bra løsning for å samle hardvare og ledninger, relativt ryddig også. Var for så vidt mulig å plassere skapet der du ville, en måtte bare ha lenger ledninger.

2.21 Dimensjonering av pumpe

En så tidlig bort fra det planlagte «slusesystemet» i forstudierapporten [vedlegg A], da dette systemet kun ville kunne påføre røre ved en posisjon. Man ville jo kunne påføre røre ved minimum 4 posisjoner. En bestemte seg for et system der en magnetventil [16] ble flyttet rundt av robotarmen, og åpnet seg ved de angitte posisjonene. Denne ventilen skulle på et vis bli koblet til en tank med vaffelrøre, hvor slanger, ved hjelp av en pumpe, transporterte røre opp til ventilen. Det ble oppgitt at den «normalt lukkede» magnetventilen fungerte ved trykk mellom 0.2 bar og 8 bar [16]. Så man måtte se til at trykket som nådde ventilen var en plass imellom dette. Pumpesystemet fungerte slik at en 10l-gryte over bakkeplan fungerte som en tank for vaffelrøren. En slange ble koblet til bunnen av gryten, og førte røren ned til en pumpe, på bakkeplan. Pumpe førte så røren opp til det høyeste punktet, som var høyere enn ventilen. Før røren gikk fra det høyeste punktet ned til ventilinngangen, hvor diameter for ventilutgangen var lavere enn diameteren til røret som gikk fra pumpen til ventilen. Ventilen ble festet til akslingen til motor 6, hvor den ble limt på sammenkoblingsdelen i figur B13. Slangen som var festet til ventilen hadde også en lavere diameter enn slangen som kom fra tanken. Systemet ble derfor delt opp i 5 punkt (se figur 1.1.1 under innledningen, og figur 2.21.1 under):

- 1: Oppi tanken. Åpen til atmosfæren.
- 2: Ved inngang til pumpe. Antok økning av trykk pga. pumpetrykk.
- 3: Høyeste punkt. Antok økning av trykk pga. pumpe. Antok reduksjon av trykk pga. høydeforskjell og friksjon.
- 4: Rett før inngang til ventilen. Antok økning i trykk pga. høydeforskjell.
- 5: Ventilåpning. Antok økning i hastighet, ved åpning av ventil, pga. diameterforskjell.

Figur 2.21.1-Tank til vaffelrøre, hvor slange fører røren fra tanken til pumpen, lokalisert rett til høyre for slangen i bildet.

Det kan legges til at før man endte med dette systemet så prøvde man en annen løsning. Dette systemet bestod også av en tank, pumpe, ventil, og slanger. Vi prøvde med en 2 liters piskebolle som tank, men siden det normalt blir laget 4l vaffelrøre under steke-prosessen, valgte vi å gå over til en 10l-gryten. Ikke bare for størrelsen, men for sterkere, tykkere materiale der slangekoblingene ble sveiset til. I bunnen ble den første slangekoblingen sveiset fast. Dette for å få den under det laveste punktet, hvor væsken normalt sett trekker mot. Videre gikk det en slange til sentrifugalpumpen. Slangen fra pumpen gikk så til et T-stykke (forgreining), hvor den ene slangen gikk tilbake til tanken og den andre gikk til ventilen. Hensikten med dette var å ha et system der røren gikk i sirkulasjon, for å alltid holde væske i slangene, og sørge for at det ikke bygget seg opp et for høyt trykk. Etter mange testforsøk fant vi ut at vi måtte droppe sirkulasjonsslangen. Dette fordi væskemengden til ventilen tilført ventilen var tilnærmet lik 0. Som for så vidt ga mening, da returpunktet på gryten var lokalisert lavere enn ventilåpningen (væskestrømmen følger minste motstands vei). Man

kunne sørget for en løsning der gryten var lokalisert høyere enn ventilåpningen, og slik sørge for at væsketrykket ved åpningen var høyt nok. Dette ville medført en del arbeid, så man bestemte seg heller for en annen og enklere løsning, nemlig løsningen beskrevet i det forrige avsnittet.

Før noen dimensjonering ble foretatt brukte man først en brukt sentrifugalpumpe for kjølevæske, som produserte et trykk på 0.5 bar [46]. Dette fordi man hadde den for hånden, fra en gammel bruktbil, som gjorde at kostnadene var lik null. Det kom røre ut av ventilåpningen, i konfigurasjonen nevnt i forrige avsnitt, men det var så vidt det dryppet fra åpningen. Med pumpen som man skulle kjøpe som et resultat av dimensjoneringen ønsket man altså et høyere trykk.

En beregnet først trykket ved alle de 5 punktene tidligere beskrevet, med et pumpetrykk på 0.5 bar. Med åpen ventil. En tok høyde for friksjon i røret, som var en forutsetning med en så viskøs væske som vaffelrøre, og beregnet trykkene ved de ulike punktene ved hjelp av Bernoullis ligning, inkludert friksjon (likning 2.21-2 under) [47]. Pga. diameterforskjeller fikk en også bruk for det faktum at tversnittarealet multiplisert væskehastighet er konstant (likning 2.21-1). Friksjonshøydefallet i likning 2.21-2 fant en med likning 2.21-3 under [48]. Hvor man videre fant friksjonsfaktoren med likning 2.21-4 under [49]. Reynolds tall ble funnet med likning 2.21-5 [50]. Følgende antagelser ble gjort:

- Man antok at væskehastigheten i tanken var 0. Den var åpenbart ikke helt 0, men sammenlignet med hastigheten i rørene, var påvirkningen på Bernoullis likning ubetydelig. En antagelse man gjorde, fordi likning 2.21-2 ville hatt fire ukjente hvis ikke, og man kunne derfor ikke løst den mellom de to første punktene.
- Man antok at trykket i tanken var 1 ATM, då det var åpent til atmosfæren.
- Nå stod man igjen med tre ukjente ved punkt 2: trykket, hastigheten og friksjonshøydefallet.
- Siden friksjonshøydefallet er en funksjon av hastigheten, måtte man delvis gjette seg til en verdi for friksjonshøydefallet mellom de første to punktene. Man fant et diagram for friksjonshøydefall [51], og så at friksjonshøydefallet ved den laveste strømningshastigheten var 2.94m pr. 100m rør. Vårt rør var 0.45m, som impliserte et friksjonshøydefall for vann på 0.013m. Som man senere kom fram til, så var viskositeten til vaffelrøre 2000 ganger høyere enn vann, så man kunne antatt at friksjonshøydefallet også var 2000 ganger større. Dette ville blitt feil, siden

strømningshastigheten og hastighet i systemet var mye lavere enn ved vann. Man satte et forhold man syntes virket logisk, og sa at friksjonshøydefallet for vaffelrøre var 10 ganger større enn vann, selv om det trolig var noe større. Man antok derfor at friksjonshøydefallet mellom punkt 1 og 2 var 0.13m=13cm.

- Væskehastigheten ved punkt 2 kunne nå beregnes, etter å ha omformet likning 2.21-3-2.21-5 til likning 2.21-7.
- Deretter kunne trykket ved punkt 2 beregnes.
- Strømningshastighet (konstant), og væskehastigheter ved de resterende punktene ble beregnet med likning 2.21-1. Strømningshastighet ble beregnet til 0.02l/s=0.2dl/s, som ga mening, basert på det man observerte ved ventilåpningen.
- Trykkene ved de resterende punktene kunne nå beregnes med likning 2.21-2, nå som hastighet, friksjonshøydefall, og trykk ved det foregående punktet var kjent.

$$\boldsymbol{Q = \frac{V}{t} = Av \;\rightarrow\; A_1 v_1 = A_2 v_2 \rightarrow v = \frac{Q}{A}} \qquad (\mathbf{2.21-1})$$

$$\boldsymbol{P_1 + \frac{1}{2}\rho v_1^2 + \rho g h_1 + P_P = P_2 + \frac{1}{2}\rho v_2^2 + \rho g h_2 + f_h \rho g} \qquad (\mathbf{2.21-2})$$

$$\boldsymbol{Hvor\ f_h = \frac{fLv^2}{2Dg}} \qquad (\mathbf{2.21-3})$$

$\boldsymbol{L}$ *er lengden på røret og* $\boldsymbol{D}$ *er diameteren.*

$$\boldsymbol{f = \frac{64}{Re}\ ved\ laminær\ flyt.} \qquad (\mathbf{2.21-4})$$

$$\boldsymbol{Reynolds\ tall, Re = \frac{\rho v D}{\mu}} \qquad (\mathbf{2.21-5})$$

$\boldsymbol{\mu}$ *er rørens viskositet.*

For å beregne viskositeten til røren ble det brukt en metode hvor en slapp en kule fra toppen av røren bunnen til røren, med en kjent høyde, hvor en tok tiden på fallet [52]. Som man ser i likning 2.21-6 under så er viskositeten en funksjon av massetettheten til kulen -og røren, samt radius til kulen. Massetettheten til røren ble enkelt funnet ved å veie en liter røre, som da viste 1002g på kjøkkenvekten.

$$\boldsymbol{\mu = \frac{(\rho_{kule} - \rho_{røre})}{v_t} g \frac{2}{9} r^2} \qquad (\mathbf{2.21-6})$$

$$Hvor\ \boldsymbol{v_t = \frac{h}{t}}$$

$$\boldsymbol{v = \frac{f_h \rho g D^2}{32 \mu L}} \qquad (\mathbf{2.21-7})$$

Kulen av stål (massetetthet 7900$\frac{kg}{m^3}$), med radius på 3mm brukte i snitt 1.18 sekund på å falle en distanse på 8cm. Viskositeten til en standard vaffelrøre ble derfor beregnet til å være 2.0 Pa*s. Interessant nok kunne en legge merke til at massetettheten til røren var det samme som vann, mens viskositeten var 2000 ganger større.

Trykkene i vaffelrøren ved de 5 ulike punktene ble beregnet med MATLAB-kode [vedlegg K], og er vist i tabell 2.21.1 under, sammen med andre viktige parametere.

Tabell 2.21.1- Trykk ved ulike punkter i pumpesystemet ved pumpetrykk på 0.5 bar og åpen ventil. Væske er vaffelrøre.

	Diameter, **D** (m)	Lengde på rør fra punkt til neste, **L** (m)	Høydeforskjell fra punkt til neste, **h** (m)	Hastighet, **v** (cm/s)	Strømningshastighet, **Q** (l/s)	Væsketrykk, **P** (bar)
Punkt 1-Oppi tank	0.26	0.45	-0.25	0	0.02	1.013
Punkt 2-Ved inngan	0.021	0.90	0.45	1.96	0.02	1.525

g til pumpe						
Punkt 3-Høyeste punkt	0.013	0.30	-0.20	16	0.02	0.933
Punkt 4-Rett før inngang til ventil	0.013	0.052	-0.052	16	0.02	0.771
Punkt 5-Utgang til ventil	0.008	X	X	42	0.02	0.555

Basert på tallene i tabell 2.21.1 kunne man konkludere med at trykket ved ventilåpningen var tilstrekkelig, selv med pumpen som kun produserte 0.5 bar, men det var bare så vidt, da trykket som nådde åpningen kun var 0.5 bar, som jo forklarte hvorfor det knapt dryppet ut av ventilåpningen. Man ville ha en høyere strømningshastighet enn denne pumpen kunne gi, så man kom over følgende pumpe, som kunne tilføre dette:

- Kayoba helautomatisk trykkvannspumpe, 7.6$\frac{l}{m}$, maks 3.8 bar (kunne justeres ned), opererte på 12V, 1199kr [53]. Se figur x under.

Trykket ved ventilåpningen ville åpenbart være høyere enn 0.2 bar, så spørsmålet man måtte stille seg var om trykket ble høyere enn 8 bar ved lukket ventil. Beregninger med likning 2.21-2 ble gjennomført med pumpetrykk på 3.8 bar, og lukket ventilåpning. Man så om trykket ved ventilåpningen var under 8 bar. Siden ventilen var lukket, var følgelig strømningshastigheten lik 0. Dette forenklet beregningene. Det var kun høyde og pumpetrykk som endret væsketrykket. Man fant ut at væsketrykket ved ventilåpningen, med lukket ventil, var på 4.82 bar [utregning i vedlegg K], altså godt under maksimal tillatt trykk

på 8 bar. Væsketrykk ved de ulike punktene kan ses i tabell 3.3.4 under resultater. Pumpen kunne altså stå på kontinuerlig under hele steke-prosessen. Man gikk derfor til innkjøp av pumpen. Systemet ble først testet med vann, deretter med røren oppi. Man kunne konkludere med at trykket i systemet med vann som væske var høyt nok til at rørene og ventilen ble tilstrekkelig rene. Vaskeprosessen foregikk ganske enkelt slik: Man sørget for at robotarmen var i hvileposisjon, og åpnet ventilen med en egen knapp [vedlegg J], og plasserte et kar eller en bøtte under. Ventilen ble lukket med samme stoppknappen som førte robotarmen tilbake til hvileposisjon. En skylte deretter igjennom nok ganger, slik at systemet ble tilstrekkelig rent.

Det siste man trengte å vite var hvor lenge ventilåpningen for å få ønsket mengde vaffelrøre på vaffeljernet, som var 0.5 dl [54]. Man måtte derfor finne strømningshastigheten i systemet. Man brukte samme prosedyre som ved 0.5-bar pumpen til å anslå et friksjonshøydefall [vedlegg K]. Bare at man antok minimum dobbel strømningshastighet pga. høyere pumpetrykk. Man fant da et anslått friksjonshøydefall på 0.44m mellom punkt 1 og punkt 2. Hastigheten i væsken ble deretter beregnet, før man fant strømningshastigheten med likning 2.21-1. Den ble beregnet til å være 0.072l/s=0.72dl/s. Som impliserte at ventilen måtte stå åpen 0.7s=700ms. Om man senere fant ut at denne verdien måtte endres pga. feil i antagelser, eller behov for større volum røre på jernet, kunne den enkelt endres i styreprogrammet [vedlegg J, nederste funksjon]. Trykkene ved punkt 1 til 5, med åpen ventil, og med pumpetrykk på 3.8 bar, finnes i tabell 3.3.4 under resultater. Man brukte nøyaktig samme metode som ved beregningene med 0.5-bar kjølevæske-pumpen [vedlegg K].

2.**22 Sammensetning av det mekaniske, det elektriske og programvaren.**

Motorer ble montert på motorholdere. Akslinger ble festet på motorakslingene. Braketter ble montert fast. Motorholder ble montert på hverandre. Programvare ble testet, hvor man gjerne endret på ledningskonfigurasjonen underveis, om man så de kom i klem. Pumpesystem ble testet. Robotarmen ble montert på en stålplate, og ikke fastmontert på pauserommet, slik at den kunne flyttes rundt etter behov. Man sørget også for at tidsintervallet etter lukking av ventil var tilstrekkelig til at ventilen fikk dryppet av seg (6s etter testing), før den gikk videre til neste posisjon [vedlegg J, siste funksjon]. Dette for å unngå søl. Brakettene og resten av delene vi hadde laget på verkstedet ble lakkert med sort og gull farge for å få en finere finish på robotarmen. Også rammen til tanken.

3

Resultater

3.1 Funksjonaliteten til robotarmen

Det første og viktigste resultatet som er relevant, er om robotarmen fungerte som den skulle. Oppnådde den ønskede posisjoner, kom det ut røre de plassene det skulle, og med riktig volum-mengde? Svaret på det er at det gjorde det. Ønskede posisjoner ble oppnådd, og ønsket/riktig mengde røre kom ut av ventilåpningen ved de nevnte posisjonene. Så kan det legges til man ikke hadde tatt høyde for motstanden ved å rotere slangen med røre i, for motor 6, kun rotasjon av ventilen. Det burde man gjort, for motor 6 klarte ikke å roteres når slangen var festet til ventilen. Hvilemomentet var heller ikke stort nok til at slangen holdt seg i ro. Man måtte derfor fast-montere slangen m. ventilen til motorholder 6, slik at rotasjonen til motor 6 ble låst. Løsningen ses i figur 3.1.1 under. Siden motor 6 sin rotasjon ble låst, endte man opp med å måtte kompensere litt med rotasjonen fra de andre motorene. Man endte i all hovedsak opp med å trekke i fra motor 6 sin rotasjonsvinkel fra motor 3 sin rotasjonsvinkel pga. konfigurasjonen tilsa at det var det enkleste. Slik oppnådde man likevel ønsket posisjon [vedlegg J].

Figur 3.1.1-Ny løsning for fastmontering av ventil.

3.2 Sikkerhetsfaktorer for bæredeler

Under, i tabell 3.2.1 til 3.2.4 følger oppnådde sikkerhetsfaktorer ved dimensjonering mht. flyting, knekking, glidning og deformasjoner. Man oppnådde minimumskravet for sikkerhetsfaktorer ved alle kritiske punkter, bortsett fra tre: Krympespenningene i krympeforbindelsene mellom kulelager og braketter for motor 3 og 5, samt den kritiske bøyespenningen i platen, som opprinnelig skulle festes til benken på pauserommet. Disse var alle hårfint under 3.

Tabell 3.2.1-Dimensjonering mht. flyting

	Forbindelse	Type spenning	Linjer med beregning i MATLAB-kode [vedlegg I]	Figur	Sikkerhetsfaktor mot flyting
Motorholder 1/bjelke	Enkeltdel	Ekvivalentspenning	9-28	D1	550
Plate med motorholder 1 til benk	Skrueforbindelse	Ekvivalentspenning	30-53	D2	1.65
Plate med motorholder 1	Enkeltdel	Kritisk bøyespenning	55-72	D1	963 om z-aksen og 2.58 om x-aksen.
Plate til motorholder 1	Sveiseforbindelse	Ekvivalentspenning	Sveisekalkulator Autodesk Inventor	D1	>100
Forbindelse på aksling til motor 1	Enkeltdel	Ekvivalentspenning med spenningskonsentrasjon	101-117	D4	11.3
Forbindelse mellom	Skrueforbindelse	Ekvivalentspenning	119-144	D5	1.51

aksling motor 1 og motorholder 2					
Tynneste del av forbindelse på aksling til motor 1	Enkeltdel	Kritisk bøyespenning	169-183	D4	385 om z-aksen og 6.7 om x-aksen.
Forbindelse på aksling til motor 2	Enkeltdel	Ekvivalentspenning med spenningskonsentrasjon	185-201	D9	9.35
Forbindelse mellom brakett-side m. kulelager og motorholder 2	Skrueforbindelse	Ekvivalentspenning	203-225	D7	1.63
Plate på motorholder med hull til skrue.	Enkeltdel	Ekvivalentspenning	246-261	D8	7.92
Forbindelse mellom brakett-side m. 4 skruehull og motorholder 2	Skrueforbindelse	Ekvivalentspenning	263-287	D9	1.55
Forbindelse mellom kulelager og	Krympeforbindelse	Krympespenning	302-311	D6	4.06

u-brakett festet til motor 2					
Forbindelse mellom aksling motor 2 og u-brakett	Enkeltdel	Kritisk bøyespenning	313-326	D9	14.6 om y-aksen og 20.9 om z-aksen
Forbindelse mellom 2 u-braketter.	Skrueforbindelse	Ekvivalentspenning	364-390	D10	1.52
Forbindelse mellom aksling motor 3 og u-brakett	Skrueforbindelse	Ekvivalentspenning	430-452	D11	1.57
Forbindelse mellom aksling motor 3 og u-brakett	Enkeltdel	Kritisk bøyespenning	454-467	D11	266 om z-aksen og 50.7 om x-aksen
Forbindelse mellom kulelager og motorholder 3	Skrueforbindelse	Ekvivalentspenning	469-492	D12	1.63
Forbindelse mellom kulelager og u-brakett festet til motor 3	Krympeforbindelse	Krympespenning	419-428	D12	2.86

Motorholder 3	Sveiseforbindelse	Ekvivalentspenning	Sveisekalkulator Autodesk Inventor	B8	>100
Forbindelse mellom motorholder 3 og motorholder 4	Skrueforbindelse	Ekvivalentspenning	494-515	D13	1.61
Forbindelse mellom aksling motor 4 og u-brakett	Skrueforbindelse	Ekvivalentspenning	517-539	D14	1.62
Motorholder 4	Sveiseforbindelse	Ekvivalentspenning	Sveisekalkulator Autodesk Inventor	B10	>100
Forbindelse mellom aksling motor 5 og u-brakett	Skrueforbindelse	Ekvivalentspenning	541-563	D15	1.65
Forbindelse mellom kulelager og u-brakett festet til motor 5	Krympeforbindelse	Krympespenning	603-612	D16	2.86
Forbindelse mellom motorholder 5 og	Skrueforbindelse	Ekvivalentspenning	614-635	D17	1.63

motorholder 6					
Motorholder 6	Sveiseforbindelse	Ekvivalentspenning	Sveisekalkulator Autodesk Inventor	B12	>100

Tabell 3.2.2-Dimensjonering mht. knekking

	Forbindelse	Type spenning	Linjer med beregning i MATLAB-kode [vedlegg I]	Figur	Sikkerhetsfaktor
Plate på motorholder 2, festet til brakett-side m. kulelager	Enkeltdel	Eulerspenning	227-244	D8	41.1
u-brakett festet til motor 2	Enkeltdel	Eulerspenning	328-349	B6	**3.90 (0.770 m. 2mm tykkelse)**
u-brakett festet til motor 4 og 5	Enkeltdel	Eulerspenning	566-588	B11	4320

Tabell 3.2.3-Dimensjonering mht. glidning.

	Forbindelse	Sikkerhetsfaktor definert før montering?	Linjer med beregning i MATLAB-kode	Figur	Sikkerhetsfaktor

			[vedlegg I]		
Motorholder 1 og motor 1	Skrueforbindelse	Ja	74-86	D3	**10.0**, ved rotasjonsvinkel på 0.1^{o} ved montering
Aksling motor 1 og forbindelse	Settskrueforbindelse	Ja	88-99	D5	**10.0**, ved rotasjonsvinkel på 0.3^{o} ved montering
Kulelager mellom motorholder 2 og u-brakett	Krympeforbindelse	Nei	289-300	D6	**14.3**, ved hulldiameter på 25.98mm
Aksling motor 2 og forbindelse	Settskrueforbindelse	Ja	351-362	D9	**10.0**, ved rotasjonsvinkel på 1.7^{o} ved montering
Aksling motor 3 og forbindelse	Settskrueforbindelse	Ja	392-404	D11	**10.0**, ved rotasjonsvinkel på 1.01^{o} ved montering
Kulelager mellom motorholder 3 og u-brakett	Krympeforbindelse	Nei	406-417	D12	**102**, ved hulldiameter på 18.98mm.
Kulelager mellom motorholde	Krympeforbindelse	Nei	590-601	D16	**102**, ved hulldiameter på 18.98mm.

r 5 og u-brakett					

Tabell 3.2.4-Dimensjonering mht. deformasjoner

	Forbindelse	Samme materiale?	Linjer med beregning i MATLAB-kode [vedlegg I]	Figur	Sikkerhetsfaktor mot uønsket deformasjon
Plate med motorholder 1 til benk	Skrueforbindelse	Ja (1 stålplate)	146-152	D2	59.5
Forbindelse mellom aksling motor 1 og motorholder motor 2	Skrueforbindelse	Ja (stålplate og stålplate)	154-168	D5	44.7

3.3 Vrimomenter, reaksjonskrefter -og momenter og trykk i væskesystemet.

Under, i tabell 3.3.1 og 3.3.2 følger vrimomenter ved utgangspunkt, og så etter modifiseringer av motor 6, med utgangskonfigurasjon, hvor alle motorene var de samme. Alt med utgangspunkt i Lagrange-mekanikken. I tabell 3.3.3 følger reaksjonskrefter -og momenter etter Newton-Euler-analysen, med de endelige motorene. Nederst, i tabell 3.3.4 finner man trykket ved ulike punkter i væskesystemet.

Tabell 3.3.1-Vrimomenter (Nm) hvor alle motorer er NEMA 23 57x57, ved ulike posisjoner. Alle motorer går samtidig, med maks hastighet og akselerasjon for alle motorer. Posisjon 1 i kolonne 1, posisjon 2 i kolonne 2 osv. Vrimoment for motor 1 i rad 1 osv.

	Posisjon(pos) 1	Pos 2	Pos 3	Pos 4	Pos 5
T1	-25.51	11.86	9.41	0.44	2.77

T2	-30.5	12.55	32.08	-20.82	-2.5
T3	-28.49	1.05	-0.74	-14.21	4.03
T4	18.55	10.66	10.38	16.78	12.71
T5	1	2.69	9.92	-8.39	-5.87
T6	0.02	-0.01	-0.03	0	0

Tabell 3.3.2-Maksimale dreiemomenter (Nm) etter modifisering av motor 6

	Posisjon(pos) 1	Pos 2	Pos 3	Pos 4	Pos 5
T1	-23.49	10.11	8.07	-1.61	1.32
T2	-25.92	10	26.74	-18.97	-2.82
T3	-24.51	-0.31	-1.57	-13.81	2.53
T4	15.7	8.76	9.07	14.57	11.18
T5	1.23	2.23	8.11	-6.54	-4.92
T6	-0.02	-0.01	-0.03	0	0

Tabell 3.3.3-Maksimale reaksjonskrefter (N) og momenter (Nm) for hvert ledd. Sammenligner alle relevante posisjoner. Endelige resultater etter andre modifisering av samtlige motorer

	Ledd 1	Ledd 2	Ledd 3	Ledd 4	Ledd 5	Ledd 6
F_X(Radiell)	5.90	-1.32	12.7	3.88	-5.65	-11.0
F_Y(Radiell)	1.50	23.9	19.1	-5.44	13.2	0.510
F_Z(Aksial)	40.6	-5.97	-5.71	-16.82	-1.82	-5.19
M_X(Bøyemoment)	-5.98	-0.700	0.640	0.73	0	0.0100
M_Y(Bøyemoment)	-4.60	1.37	-0.400	0.59	0.06	-0.0200
M_Z(Dreiemoment)	1.37	7.51	0.870	-0.06	0.100	0

Tabell 3.3.4- Trykk ved ulike punkter i pumpesystemet ved pumpetrykk på 0.5 bar og åpen ventil. Væske er vaffelrøre.

	Diameter, **D** (m)	Lengde på rør fra punkt til neste, **L** (m)	Høydeforskjell fra punkt til neste, **h** (m)	Hastighet, **v** (cm/s)	Strømningshastighet, **Q** (l/s)	Væsketrykk, **P** (bar)
Punkt 1-Oppi tank	0.26	0.45	-0.25	0	0.072	1.013
Punkt 2-Ved inngang til pumpe	0.021	0.90	0.45	6.62	0.072	4.79
Punkt 3-Høyeste punkt	0.013	0.30	-0.20	54.3	0.072	2.90
Punkt 4-Rett før inngang til ventil	0.013	0.052	-0.052	54.3	0.072	2.30
Punkt 5-Utgang til ventil	0.008	X	X	143	0.072	1.55

3.4 Totale kostnader

Man gikk på forhånd ut og sa at man ønsket å konkurrere med eksisterende produkter når det kom til pris. -Så hva er prisen på tilsvarende produkter? Man kom over en lettvektsrobot, lansert tilbake i 2017, som lå ute for 40000kr [55]. Det står i artikkelen at dette var en så revolusjonerende pris, at man så på det som «ultrabillig». Den kostet f.eks. en tredjedel av hva tilsvarende robotarmer hadde vært prissatt. Den veide 8kg, og kunne altså flyttes rundt i rommet. Den tålte en ytre belastning 1.25kg, står det. Vår robotarm er dimensjonert for å tåle minimum 1kg, og etter å ha prøvd selv, kunne man slå fast at den også, uten problem, kunne flyttes rundt i rommet, etter behov. Om en ser bort fra brukervennlighet, brukersikkerhet og estetikk kan man i alle slå fast at det er snakk om to relativt tilsvarende produkt. Så hva ble prisen på selve robotarmen, sett bort fra pumpesystem, og utstyr spesifikt til dette? Og kunne den konkurrere med prisen til robotarmen som kostet 40000kr? -Den totale kostnaden ved selve robotarmen kan ses under. Priser er pr. 09.03.2022, og man ser bort fra kostnader ved materiale, da dette ble betraktet som avfall. En ser også bort fra kostnader ved skruer, muttere, skiver, kulelager, og diverse ledninger/annet elektrisk tilleggsutstyr. Fraktkostnader er også sett bort fra.

• "Nema 23 Stepper Motor Bipolar L=56mm w/ gear-ratio 4:1 Planetary Gearbox", motor 1, antall: 1stk, pris: 54.77$ [28]

• "Nema 23 Stepper Motor Bipolar L=56mm w/ gear-ratio 47:1 Planetary Gearbox", motor 2, antall: 1stk, pris: 54.77$ [29]

• "Nema 14 Stepper Motor Bipolar L=33mm w/ gear-ratio 100:1 Planetary Gearbox", motor 3, motor 4 og motor 5, antall: 3stk, pris: 3x 32.44$=97.32$ [30]

• «Nema 17 Bipolar 1.8deg 26Ncm 0.4A 12V 42x42x34mm 4 Wires», motor 6, antall: 1stk, pris: 8.05$ [31]

• «Digital Stepper Driver 0.3-2.2A 10-30VDC for Nema 8,11,14,16,17 Stepper Motor», driver for motor 3,4,5, og 6, antall: 4stk, pris: 4x13.59$=54.36$ [32]

• «Digital Stepper Driver 1.8-5.6A 20-50VDC for Nema 23,24,34 Stepper Motor», driver for motor 1 og 2, antall: 2stk, pris: 2x22.66$=45.32$ [33]

• «LRS-35-24 MEAN WELL 35W 24VDC 1.5A 115/230VAC Enclosed Switching Power Supply", spenningskilde motor 6, antall: 1stk, pris: 8.15$ [34]

• «LRS-200-24 MEAN WELL 200W 24VDC 8.8A 115/230VAC Enclosed Switching Power Supply", spenningskilde motor 3, 4 og 5, antall: 1stk, pris: 19.36$ [35]

• «350W 24VDC 14.6A 115/230VAC Switching Power Supply for CNC Router Kits", spenningskilde motor 1 og 2, antall: 1stk, pris: 26.89$ [36]

• Sum motorer, drivere og spenningskilder på stepperonline.com: 368.99$=3310kr

• "Elegoo Arduino Mega R3 ATmega 2560+USB-Cable", kontroller, antall: 1stk, pris: 22.99$ [37]

• "360 Pieces Multicolored Breadboard Jumper Wire Dupont Wire Ribbon Cables Kit 30CM 20CM 10CM 40 Pin Male to Female, 40 Pin Male to Male, 40 Pin Female to Female, for Arduino and Raspberry Pi", ledninger, antall: 2stk, pris: 2x17.99$=35.98$ [56]

• Sum varer kjøpt på Amazon.com: 58.97$=**529kr**

• Sum kostnader: **3839kr.**

Anslår man kostnader knyttet til elektrisk tilleggsutstyr, ledninger, bolter, muttere, skiver, kulelager og fraktkostnader ender man nok opp på en plass mellom **5000-6000kr.** Altså omtrentlig en åttendedel av 40000kr. En kan derfor slå fast at UiS trolig har spart penger på å gå for denne løsningen, framfor å kjøpe inn en industriell robotarm. Selv om de da ville fått et mer brukervennlig, mer estetisk og mer sikkert produkt å håndtere.

Totale kostnader for væskesystem ble også beregnet:

- «Kayoba helautomatisk trykkvannspumpe 12V», Pumpe, antall: 1stk, pris: 1199 kr [53]
- «Slange miljøtex luft/kjem 3/8», Slange, antall: 1m, pris: 54,74 kr (kjøpt fysisk i butikk)
- «Slange miljøtex luft/kjem 3/4», Slange, antall: 3m, pris: 328,44 kr. (kjøpt fysisk i butikk)
- «Slangenippel 3/4 BSPT SL A31», Nippel, antall: 1stk, pris: 59,24 kr. (kjøpt fysisk i butikk)
- «Xtra gryte 10L rustfritt stål 26x20 cm», Gryte, antall: 1stk, pris: 499 kr. (kjøpt fysisk i butikk)
- «Solenoid magnetventil med slangetilkobling Ø 11mm», Ventil, 1stk, pris 249kr [16]
- Sum kostnader væskesystem: **2389kr.**

Slår man sammen kostnadene på robotarmen og pumpesystemet ender man derfor opp med total kostnad enn plass mellom **7000-8000kr**, om man tar med elektrisk tilleggsutstyr, diverse ledninger, bolter, muttere, skiver og kulelager med i beregningen.

4

Diskusjon

4.1 Konseptet

Konseptet som ble valgt var i hovedsak for å holde kostnadene så lave som mulig. Samtidig så ønsket man å produsere noe man aldri har sett før. Om ikke et unikt produkt, så i alle fall noe med en viss form for særegenhet. Det finnes sikkert der ute, men pumpesystemet, slik vi endte opp med å montere, har i alle fall aldri vi sett før. For å få et så unikt design har vi også konstruert delene selv, om enn med inspirasjon fra diverse standarddeler. Å lage u-braketter er ikke akkurat nytenkende, men når det gjelder noen av motor-holderne, har en i alle fall ikke sett noe tilsvarende ute på det åpne markedet. På UiS står f.eks. en «mars-rover» som også har en lignende robotarm, hvor det er tydelig bestilt deler fra fabrikant. Det var en tanke å bestille slike deler, og da ville roboten vår sett mer estetisk og mer profesjonell ut. Dette hadde kostet veldig mye mer, så vi bestemte oss for å konstruere delen selv, og ta den utfordringen. En kunne også 3D-printet deler av karbonfiber for å få et mer nøyaktig/mindre skjevt design, samt noenlunde den samme styrken som ved rustfritt stål [57]. Roboten ville også blitt betraktelig lettere, og man hadde trolig ikke trengt så kraftige motorer som de vi gikk til innkjøp av. Man valgte likevel å ikke gjøre dette, da det ville medført høyere kostnader, samt så var det ofte stor kø/pågang rundt 3D-printeren. Men begrunnet også valget av miljøhensyn, da karbonfiber, i motsetning til rustfritt stål, er vanskelig å resirkulere [57]. Når det kommer til det elektriske så er det også blitt gjort på egenhånd selv om ingen av oss har noen omfattende bakgrunn fra faget. Kanskje dette er noe som kunne vært gjort av data/elektroingeniør -studenter. Det var snakk om at vi skulle få med en i fra elektro på prosjektet, så det er også med på å fortelle hvor omfattende denne delen av prosjektet var. Det har vært en utfordring, og det gjør det bare oppmuntrende da en har klart dette selv.

4.2 Tilstrekkelige sikkerhetsfaktorer?

Som nevnt under kapittel 3.2, under resultater så man at sikkerhetsfaktorene var tilstrekkelige, bortsett fra ved 3 punkter. Hva kunne en gjort for å forbedre disse sikkerhetsfaktorene? -Vel, når det kommer krympespenningen pga. krympeforbindelsen for

kulelageret til motor 3 og 5 så kunne en enkelt og greit gjort denne mindre ved å slipe hullet ørlite større. Dette kunne man gjort uten at det ville gått på bekostning av sikkerhetsfaktoren mot glidning, da denne i utgangspunktet var skyhøy (tabell 3.2.3). Når det kommer til den kritiske bøyespenningen i platen som opprinnelig skulle festes til benken på pauserommet, kunne man gjort denne mindre ved å velge en plate som var tykkere enn 2mm, som f.eks. 3mm. Dette problemet løste seg etter hvert på egenhånd, då man heller gikk for en løsning der man monterte platen fast på en annen 2mm-plate, av karbonstål, slik at klemforbindelsen mellom skruene og boltene ble på 4mm. Til slutt så kan det nevnes at sikkerhetsfaktoren mht. knekking av braketter og andre deler som er utsatt for dette kunne blitt gjort høyere ved å sveise på noen forsterkninger i hjørnene. Forsterkningene ville gjort at treghetsmomentet om den svakeste aksen ville blitt langt høyere, og slik ville Euler-spenningen også blitt høyere. På den måten ville man fått en økt sikkerhetsfaktor. Man valgte til slutt å la vær å gjøre dette, da sikkerhetsfaktoren mht. knekking var over 3 ved alle kritiske punkter, og sveisingen ville kunne gjøre at materialet krympet seg. På grunn av dette kunne det oppstått enda flere skjevheter enn det man så i det endelige produktet, og det var ikke sikkert man ville klart å sette delene sammen etterpå. Så kan det også sies at man ved alle beregninger så bort fra vekten av materialet, som stort sett var rustfritt stål. En vekt som på ingen måte er neglisjerbar, men det var også grunnen til at man opererte med såpass høye sikkerhetsfaktorer som man gjorde. Sikkerhetsfaktorene gjør det kan oppstå større ytre belastninger enn det robotarmen er dimensjonert for, og svakheter i materialet, uten at man trenger å se svikt i funksjonen av den grunn.

4.3 Er funksjonaliteten god nok til at roboten kan monteres permanent på pauserommet?

I slutten av prosjektet testet vi og finjusterte på koordinasjonen til robotarmen, og tidsintervallet hvor magnetventilen skulle være åpen og lukket. Dette fikk vi finjustert slik vi ønsket. Vi ba på et tidspunkt veileder, Yaaseen A. Amith om å komme ned og se på produktet/prosessen. Det var da vi diskuterte om robotarmen skulle stå permanent på pauserommet. Den fungerte jo som den skulle, og gjorde de arbeidsoppgavene den var tiltenkt. Så hvorfor skulle den evt. ikke monteres permanent på pauserommet, og tas i bruk hver fredag? Roboten i seg selv krevde etter å ha kjørt gjentatte ganger tilstramminger, spesielt hvor motorakslingene var festet til delene. Dette pga. skjevheter, som et resultat av at delene var laget for hånd, av et ikke altfor kvalifisert personell. Tank, pumpe, slange, og ventil behøver rikelig rengjøring etter bruk, dette med tanke på at vaffelrøre går gjennom og

uten rengjøring vil det ligge igjen i disse komponentene. Gryten må også demonteres fra pumpen, for å kunne bli rengjort tilstrekkelig. Ventilen må også demonteres for å bli tilstrekkelig ren. Dette medfører en økt arbeidsmengde når det kommer til vaskeprosessen. Roboten skulle jo forenkle arbeidsmengden til de ansatte, ikke pålegge dem ekstra arbeid, Så kan det også sies at hvis dette ikke blir gjort godt nok kan det oppstå mugg og det er ikke bra helsemessig, og medfører en stor risiko m.tp. matsikkerhet. Vi vil f.eks. ikke stå ansvarlige om noen av de ansatte skulle bli matforgiftet etter den ukentlige sammenkomsten. Brukervennligheten p.dd. er heller ikke helt optimalisert. For å starte systemet så må en trykke på en liten knapp, og det er ikke sikkert at samtlige klarer å se forskjellen på de ulike knappene. Å åpne ventilen når den ikke skal åpnes er ikke akkurat gunstig. Skulle det oppstå strømbrudd underveis i prosessen, må robotarmen manuelt flyttes tilbake til startposisjon. Hvis ikke vil den ved oppstart neste gang gå til helt feil posisjoner. Dette fordi programmet er laget slik at stegmotorene ved start på strømtilførsel oppfatter den posisjonen som null-posisjonen. Ved strømbrudd vil en ny nullposisjon, ved feil posisjon bli definert, om man ikke flytter den manuelt tilbake slik den skal stå. Dette er noe personellet som betjener roboten hele veien måtte hatt i bakhodet.

Ellers så er det ikke etter Helse, Miljø og Sikkerhet, forsvarlig å la hvem som helst bruke den. Roboten består av motorer som kan yte høyt dreiemoment, og skulle man få en finger i klem, så er det ikke noen mekanisme som gjør at motorene automatisk stopper. På grunn av dette kreves det avstand og våkenhet under operasjonen. På bakgrunn av utfordringene knyttet til funksjonaliteten, ble vi enige om at roboten skulle kunne brukes til framvisning, og ved spesielle anledninger. Med andre ord at den ikke skal brukes hver fredag, og dermed stå permanent montert på pauserommet, og heller tas fram når det ønskes. Vel å merke av opplært/kvalifisert personell. Hvilke tiltak kan bli iverksatt for å bedre brukersikkerhet og brukeropplevelse rundt prosessen. Her er noen tiltak, gjerne implementert av framtidige studenter fra elektro- og data.

- Installere en skikkelig nødstopp. En så lenge så fungere vippebryteren på grenuttaket som nødstopp. Den kutter strømmen til samtlige motorer, i tillegg til pumpe og ventil. Problemet er bare at grenuttaket ikke alltid er like iøynefallende, og lett å få kontroll på hvor. Samt så kan avstanden fra robotarm til grenuttak være for stor i enkelte tilfeller. En nødstopp må derfor installeres slik at den kan nås ved alle ulike tilfeller rundt betjening av robotarmen.

- Installere et bedre vaskeprogram. Trykket fra pumpen er tilstrekkelig til å rengjøre rørene, men demonteringsprosessen av gryte og ventil medfører som nevnt mye ekstra-arbeid. Det hadde derfor vært en god løsning å installere et vaskeprogram som gjorde at man slapp dette.
- Styre motor-posisjoner på et annet vis, slik at nullposisjonen ikke blir påvirket av strømbrudd. F.eks. med en roterende «koder» [58]. Dette gjør at koderne sender signal til kontrolleren hvor de befinner, en posisjon som ikke blir påvirket av strømbrudd. Ved å feste disse koderne på akslingene til motorene, eller deler som roterer likt som en motoraksling, kan man lage et langt mer pålitelig styreprogram. Man fikk kjøpt inn 10 stk roterende kodere [59], men de kom så sent fram, at man aldri rakk å implementere en god løsning. De kan uansett brukes av noen andre studenter ved et senere tidspunkt i en eventuell forbedringsprosess.
- Montere på en gripearm, og sørge for at roboten løfter opp lokket på vaffeljernet, tar ut vafler, og muligens også sprayer på fett før steking. Roboten er som tidligere nevnt, dimensjonert for dette. Målet med robotarmen var jo hele tiden å lette på arbeidsmengden til de ansatte, og klarer man å iverksette dette tiltaket nærmer denne arbeidsmengden seg nullnivå. Det ble også kjøpt i en gripearm på et tidligere stadium [60], men tiden strakk aldri til når det kom til å utarbeide en god løsning. Også en oppgave for framtidige elektro- og datastudenter.
- Justere skjevheter. Lage nye deler, etter arbeidstegninger i vedlegg B, for å justere skjevheter. Dette for å en enda mer nøyaktig prosess og et produkt som i større grad tåler tidens tann. Muligens en oppgave for framtidige maskiningeniørstudenter.

5

Konklusjon

Det å konstruere, programmere og ferdig-stille en robot av en slik komplisert skala tar lang tid. Som nevnt har vi laget delene selv. Dette har tatt veldig lang tid, og timene på verkstedet har vært mange. Heldigvis fant en fort ut at aluminium ikke var et alternativ til de bærende delene. Ellers så laget vi mange deler på nytt, pga. liten feilmargin når det kom monteringen. Hadde vi bestilt disse delene så kunne vi fått mer tid til andre ting, som for eksempel andre funksjoner, som f.eks. robotarmen kunne løfte lokket på vaffeljernene, og ta vafler av

vaffeljernene. I tillegg kunne det frigjort tid til å forbedre brukersikkerheten og brukeropplevelsen rundt håndteringen av robotarmen.

I begynnelsen så lager man seg et bilde av hvordan ting kommer til å se ut, men så har du også uforutsette ting som oppstår. Vi hadde f.eks. ikke regnet med et relativt stor el-skap når det kom til monteringen av det elektriske. En kan til slutt konkludere med at robotarmen klarte å gjennomføre de oppgavene den var tiltenkt. Den oppnådde ønskede posisjoner, og rett mengde røre ble tilført vaffeljernet ved disse posisjonene. Man var likevel ikke hundre prosent tilfreds med brukersikkerheten og brukeropplevelsen knyttet til håndteringen av produktet. Så man bestemte seg for, enn så lenge, at robotarmen kun skal tas fram ved spesielle anledninger, og kun tas i bruk av kvalifisert opplært personell. Man ser også at robotarmen er i stand til å gjennomføre flere oppgaver enn det den kan i dag, som f.eks. å løfte lokkene til vaffeljernene, og ta ut vafler av vaffeljernene. Tiltakene som blir beskrevet kan på et framtidig tidspunkt iverksettes av andre studenter på fakultetet.

Litteraturliste

[1] Autodesk. “Buy Inventor”. autodesk.com. Hentet fra: Inventor Software | Get Prices & Buy Official Inventor 2023 | Autodesk (sist besøkt 14.05.2022)

[2] Amazon. “diymore Silver ROT3U 6DOF Aluminum Robot Arm Mechanical Robotic Clamp Claw Kits(Unassemble Parts Without Servos)”. Amazon.com. Hentet fra: Amazon.com: diymore Silver ROT3U 6DOF Aluminium Robot Arm Mechanical Robotic Clamp Claw Kits(Unassembled Parts Without Servos) : Toys & Games (sist besøkt 14.05.2022)

[3] automate. “Servo Motors vs. Stepper Motors in Motion Control: How to Choose the Right One for Your Appclication”. automate.org- Hentet fra: Servo Motors vs. Stepper Motors in Motion Control: How to Choose the Right One for Your Application (automate.org) (sist besøkt 14.05.2022)

[4] Wikipedia. “Stepper motor”. Wikipedia.org. Hentet fra: Stepper motor – Wikipedia (sist besøkt 14.05.2022)

[5] Dronebot Workshop. “Big Stepper Motors with Arduino”(25.05.2019. Sist sett: 14.05.2022[Online Video]. Hentet fra: Big Stepper Motors with Arduino - YouTube

[6] Hentet fra papirleksikon «dynamikk (mekanikk)», i Snl, 2019[Online] Hentet frå: dynamikk – mekanikk – Store norske leksikon (snl.no), (sist besøkt 14.05.2022)

[7] profoundphysics. "Lagrangian Mechanics for Dummies: An Intuitive Introduction". profoundphysics.com. Hente fra: Lagrangian Mechanics For Dummies: An Intuitive Introduction – Profound Physics, (sist besøkt 14.05.2022)

[8] Engineering Educator Academy. "Equations of Motion for an N-link Robot in the Joint Space using Lagrangian Formulation"(21.11.2021). Siste sett: 14.05.2022[Online Video]. Hentet fra: Equations of Motion for an N-link Robot in the Joint Space using the Lagrangian Formulation - YouTube

[9] Reality Bytes. "Forward and Inverse Kinematics" realitybytes.blog. Hentet fra: Forward and Inverse Kinematics: Jacobians and Differential Motion | Reality Bytes (sist besøkt 14.05.2022)

[10] Wikipedia. "List of moments of inertia". Wikipedia.org. Hentet fra: List of moments of inertia - Wikipedia (sist besøkt 14.05.2022)

[11] Wikipedia. "Forward kinematics". Wikipedia.org. Hentet fra: Forward kinematics - Wikipedia, (sist besøkt 14.05.2022)

[12] stepperonline. «Nema 23-57x57mm: P Series Nema 23 Bipolar 1.8deg 1.26Nm». stepperonline.com. Hentet frå: Nema 23 - 57 x 57mm : P Series Nema 23 Bipolar 1.8deg 1.26Nm ... (omc-stepperonline.com), (sist besøkt 14.05.2022)

[13] Wikipedia. "Denavit-Hartenberg parameters». Wikipedia.org. Hentet frå: Denavit–Hartenberg parameters - Wikipedia, (sist besøkt 14.05.2022)

[14] Automatic Addison, "How to Assign Denavit-Hartenberg Frames to Robotic Arms". Automaticaddison.com. Hentet fra: How to Assign Denavit-Hartenberg Frames to Robotic Arms – Automatic Addison (sist besøkt 14.05.2022)

[15] Kinematic diagram - Wikipedia

[16] Solenoid magnetventil 12V med slangetilkoblinger Ø 11 mm (supermagneter.no)

[17] Profound Physics. "Christoffel Symbols: A Complete Guide With Examples". Profoundphysics.com Hentet fra: Christoffel Symbols: A Complete Guide With Examples – Profound Physics, (sist besøkt 14.05.2022)

[18] Ø. Grøn. «coriolisakselerasjon», i Snl, 2021[Online]. Hentet fra: coriolisakselerasjonen – Store norske leksikon (snl.no), (sist besøkt 14.05.2022)

[19]Ali Raza. “Dynamics of the n-Link Robotic Manipulators (Multi-body Mechanical Systems): General Formulation”(26.01.2021). Sist sett: 14.05.2022[Online Video]. Hentet fra: Dynamics of the n-Link Robotic Manipulators (Multi-body Mechanical Systems): General Formulation - YouTube

[20] RoboDK. “Inverse Kinematics in Robotics: What You Need to Know”. Robodk.com. Hentet fra: Inverse Kinematics in Robotics: What You Need to Know - RoboDK blog, (sist besøkt 14.05.2022)

[21] Peter Corke. “Robotics Toolbox”. Petercorke.com. Hentet fra: Robotics Toolbox - Peter Corke, (sist besøkt 14.05.2022)

[22] Mathworks. “MATLAB GUI”. Mathworks.com. Hentet fra: MATLAB GUI - MATLAB & Simulink (mathworks.com), (sist besøkt 14.05.2022)

[23] Stepperonline. “Nema 11 Bipolar 1.8deg 12Ncm (170z.in) 0.67A 6.2V 28x28x51mm 4 Wires-11HS29-0674S”. stepperonline.com Hentet fra: Nema 11 Bipolar 1.8deg 12Ncm (17oz.in) 0.67A 6.2V 28x28x51mm 4 Wires - 11HS20-0674S by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[24] Wikipedia. “Newton-Euler equations”. Wikipedia.org. Hentet fra: Newton–Euler equations - Wikipedia, (sist besøkt 14.05.2022)

[25] Engineering Educator Academy. “Equations of motion for robotic manipulators using Newton Euler Formulation”(01.12.2021). Sist sett: 14.05.2022[Online Video]- Hentet fra: https://www.youtube.com/watch?v=sOj2Y_5S9-k

[26] I-programmer. “Recorsion”. I-programmer.info. Hentet fra: Recursion (i-programmer.info), (sist besøkt 14.05.2022)

[27] Brown. “EN4 Notes: Kinematics of Rigid Bodies”. Brown.edu, Hentet fra: EN4 Notes: Kinematics of Rigid Bodies (brown.edu), (sist besøkt 14.05.2022)

[28] stepperonline. ”Nema 23 Stepper Motor Bipolar L=56mm w/ Gear Ratio 4:1 Planetary Gearbox-23HS22-2804S-PG4». stepperonline.com. Hentet fra: https://www.omc-stepperonline.com/economy-planetary-gearbox/nema-23-stepper-motor-bipolar-l-56mm-w-gear-ratio-4-1-planetary-gearbox-23hs22-2804s-pg4?mfp=184-frame-size-mm%5BNema%2023%20%2857%20x%2057%29%5D, (sist besøkt 14.05.2022)

[29] stepperonline. ”Nema 23 Stepper Motor Bipolar L=56mm w/ Gear Ratio 47:1 Planetary Gearbox-23HS22-2804S-PG47». stepperonline.com. Hentet fra: Nema 23 Stepper Motor Bipolar L=56mm w/ Gear Ratio 47:1 Planetary Gearbox - 23HS22-2804S-PG47 by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[30] stepperonline. ”Nema 14 Stepper Motor Bipolar L=33mm w/ Gear Ratio 100:1 Planetary Gearbox-14HS13-0804S-PG100». stepperonline.com. Hentet fra: Nema 14 Stepper Motor Bipolar L=33mm w/ Gear Ratio 100:1 Planetary Gearbox - 14HS13-0804S-PG100 by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[31] Stepperonline. “Nema 17 Bipolar 1.8deg 26Ncm (36.8oz.in) 0.4A 12V 42x42x34mm 4 Wires-17HS13-0401S1”. stepperonline.com Hentet fra: Nema 17 Bipolar 1.8deg 26Ncm (36.8oz.in) 0.4A 12V 42x42x34mm 4 Wires - 17HS13-0404S1 by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[32] stepperonline. “Digital Stepper Driver 0.3-2.2A 10-30 VDC for Nema 8,11,14,16,17 Stepper Motor”. Stepperonline.com. Hentet fra: Digital Stepper Driver 0.3-2.2A 10-30VDC for Nema 8, 11, ... (omc-stepperonline.com), (sist besøkt 14.05.2022)

[33] stepperonline. “Digital Stepper Driver 1.8-5.6A 20-50 VDC for Nema 23,24 Stepper Motor”. Stepperonline.com. Hentet fra: Digital Stepper Driver 1.8~5.6A 20-50VDC for Nema 23, 24, 34 Stepper Motor - DM556T by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[34]stepperonline. “LRS-35-24 MEAN WELL 35W 24VDC 1.5A 115/230VAC Enclosed Switching Power Supply.stepperonline.com Hentet fra: LRS-35-24 MEAN WELL 35W 24VDC 1.5A 115/230VAC Enclosed Switching Power Supply - LRS-35-24 by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[35] stepperonline. “LRS-200-24 MEAN WELL 200W 24VDC 8.8A 115/230VAC Enclosed Switching Power Supply.stepperonline.com Hentet fra: LRS-200-24 MEAN WELL 200W 24VDC 8.8A 115/230VAC Enclosed Switching Power Supply - LRS-200-24 by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[36] stepperonline. “350W 24V 14.6A 115/230V Switching Power Supply Stepper Motr CNC Router Kits-S-350-24”.stepperonline.com Hentet fra: 350W 24V 14.6A 115/230V Switching Power Supply Stepper Motor CNC Router Kits - S-350-24 by STEPPERONLINE (omc-stepperonline.com), (sist besøkt 14.05.2022)

[37] Amazon. “ELEGOO MEGA R3 Board ATmega 2560+USB Cable Compatible with Arduino IDE Projects RoHS Compliant”. Amazon.com. Hentet fra: Amazon.com: ELEGOO MEGA R3 Board ATmega 2560 + USB Cable Compatible with Arduino IDE Projects RoHS Compliant : Electronics, (sist besøkt 14.05.2022)

[38]Arduino. “AccelStepper”. Arduino.cc. Hentet fra: AccelStepper - Arduino Reference, (sist besøkt 14.05.2022)

[39]B.B. Larsen. “MOSFET-transistor”, i Snl, 2021[Online]. Hentet fra: MOSFET-transistor – Store norske leksikon (snl.no), (sist besøkt 14.05.2022)

[40] Hirpa G. Lemu: «Dimensjonering av maskinelementer. Kompendium i fag MSK210 Maskinkonstruksjon», (sist endret av forfatter 06.01.2020).

[41] Sbainvent. ”Bending(Transverse Shear Stress)”. Sbainvent.com. Hentet fra: Bending (Transverse Shear Stress) - S.B.A. Invent (sbainvent.com), (sist besøkt 14.05.2022)

[42] Astrup. «Sammenligningstabell». Astrup.no. Hentet fra: Rustfritt+Sammenligningstabell (1).pdf, (sist besøkt 14.05.2022)

[43] SKF. “626”. SKF.com. Hentet fra: 626 - Deep groove ball bearings | SKF, (sist besøkt 14.05.2022)

[44] SKF. “6000”. SKF.com. Hentet fra: 6000 - Deep groove ball bearings | SKF, (sist besøkt 14.05.2022)

[15]Lunelamper. «Hva betyr fargen på ledningene?». Lunelamper.no. Henter fra: Hva betyr fargen på ledningene? – Lunelamper.no, (sist besøkt 14.05.2022)

[46] Topsflo. “TL-B10 Brushless DC Pump”. Topsflo.com. Hentet fra: China 24V micro water pump suppliers,12v hot water circulation pump,12v mini dc water pump Topsflo brushless dc pump manufacturers, (sist besøkt 14.05.2022)

[47]: Itacanet. “Part 7: Pumps and Turbines-The Bernoulli Equatin”. Itacanet.org. Hentet fra: Part 7: Pumps and Turbines – The Bernoulli Equation | ITACA (itacanet.org), (sist besøkt 14.05.2022)

[48] Pipeflow. “Pipe Friction Loss Calculations”. Pipeflow.org. Hentet fra: Pipe Friction Loss Calculations (pipeflow.com), (sist besøkt 14.05.2022)

[49]: Pipeflow. “Pipe Friction Factor Calculation”. Pipeflow.com. Hente fra: Pipe Friction Factor Calculation (pipeflow.com), (sist besøkt 14.05.2022)

[50] Engineeringtoolbox. “Reynolds Number”. Engineeringtoolbox.com. Hente fra: Reynolds Number (engineeringtoolbox.com), (sist besøkt 14.05.2022)

[51] Engineeringtoolbox. “Plastic Pipes-Friction Head Loss vs. Water Flow”. Engineeringtoolbox.com. Hentet fra: Plastic Pipes - Friction Head Loss vs. Water Flow (engineeringtoolbox.com), (sist besøkt 14.05.2022)

[52] Ese.iitb.ac. “Falling Ball Viscometer”. Ese.iitb.ac.in. Hentet fra: Falling Ball Viscometer.pdf (iitb.ac.in), (sist besøkt 14.05.2022)
[53] Jula. «Helautomatisk trykkvannspumpe». Jula.no. Hentet fra: Helautomatisk trykkvannspumpe | Til ferskvann | KAYOBA (jula.no), (sist besøkt 14.05.2022).
[54] TORO. ”TORO Vafler familiepakning». Toro.no. Hentet fra: TORO Vafler familiepakning 591 g - TORO, (sist besøkt 14.05.2022)
[55] Teknisk ukeblad. «Lettvektsrobot til 40.000kr begeistrer robotforsker». Tu.no. Hente fra: Lettvektsrobot til 40.000 kroner begeistrer robotforsker - Tu.no, (sist besøkt 14.05.2022)
[56] Amazon. “360 Pieces Multicolored Breadboard Jumper Wire Dupojt Wire Ribbon Cables Kit 30CM 20CM 10CM 40 Pin Male to Female. 40 Pin Male to Male, 40 Pin Female to Female, for Arduino and Raspberry Pi”. Amazon.com. Hentet fra: Amazon.com: 360 Pieces Multicolored Breadboard Jumper Wire Dupont Wire Ribbon Cables Kit 30CM 20CM 10CM 40 Pin Male to Female, 40 Pin Male to Male, 40 Pin Female to Female, for Arduino and Raspberry Pi : Electronics, (sist besøkt 14.05.2022)

[57] L. E. Helseth. «karbonfiber», i Snl, 2021[Online]. Hentet fra: karbonfiber – Store norske leksikon (snl.no), (sist besøkt 14.05.2022)
[58] Dronebot Workshop. “How Rotary Encoder Works and How to Use It with Arduino”(25.07.2016). Sist sett: 14.05.2022[Online Video]. Hentet fra: How Rotary Encoder Works and How To Use It with Arduino - YouTube

[59] Amazon. “DaFuRui 10Pack KY-040 Rotary Encoder Module 360 Degree Rotary Encoder Module with Knob Cap for Arduino.” Amazon.com. Hentet fra: DaFuRui 10Pack KY-040 Rotary Encoder Module 360 Degree Rotary Encoder Module with Knob Cap for Arduino: Amazon.com: Industrial & Scientific, (sist besøkt 14.05.2022)
[60] Amazon. “Premium Metal Rbot Mechanical Claw/Clamp Arm/Gripper/ with High Torque Servo, Robotics Part/ Accessory Model for Arduino/Raspberry pi/Microbit to Clip/Carry, DIY AI ROS STEAM Education". Amazon.com. Hentet fra: Amazon.com: Premium Metal Robot Mechanical Claw / Clamp Arm/ Gripper / with High Torque Servo, Robotics Part / Accessory Model for Arduino / Raspberry pi / Microbit to Clip / Carry, DIY AI ROS STEAM Education : Toys & Games, (sist besøkt 14.05.2022)

Vedlegg A-Forstudierapport

Ruben Edland og Narve Gilje Nordås

Forstudierapport bacheloroppgave våren 2022

«Design og konstruksjon av vaffelrobot»

Universitetet i Stavanger

Innholdsfortegnelse

1. Introduksjon

Målet med denne forstudierapporten er å introdusere tanker og ideer rundt design og konstruksjon av en vaffellagende robot. Roboten skal designes og konstrueres våren 2022, og være ferdigstilt innen starten av mai. Rapporten inneholder også en tidsplan/gjennomføringsplan for prosjektet, blant annet i form av en Ganttdiagram.

2. Oppgavens bakgrunn

Å samles rundt et måltid er sosialiserende. Man bygger lettere relasjoner, og om det er noe ekstra godt å bite i, øker gjerne stemningen i rommet noen hakk ekstra. Vafler er en av disse rettene. Å lage vafler for et menneske er ikke en svært krevende oppgave, ei heller tar det særlig mye tid. Men se for deg følgende scenario: Alle ansatte på fakultet er opptatt med undervisning eller andre viktige oppgaver i timen før lunsjpause. Likevel ville man gjerne kunne se fram til et godt, felles måltid i lunsjen. Og det er nettopp her vaffelroboten kommer inn i bildet. Før man går til forelesning/undervisning skal man i teorien kunne trykke på en knapp, og når man kommer tilbake skal man bli møtt av et fat som bugner av vafler.

2.1 Beskrivelse av oppgave

Vaffeljern og vaffelrøre er to elementer for å lage vafler. Samt så trenger man gjerne et menneske for å helle smøre jernet, helle røre på jernet, lukke- og åpne det, samt ta vaffelen av jernet. Vi planlegger å bygge en solid, robust seksakset robotarm, som skal være montert fast til benken hvor vaffeljernet står. Robotarmen skal kunne gjennomføre alle oppgaver et menneske ville gjort, bortsett fra å lage røren, samt fylle røre på beholderen. I teorien skal også robotarmen kunne brukes til en hvilken som helst presisjonsavhengig oppgave, men i dette tilfellet skal den programmeres til å betjene vaffeljernet.

Vi tar utgangspunkt i eksisterende teknologi når det kommer til konstruksjon/programmering av robotarmen(som vist i f. eks figur 2.1), selv om den selvfølgelig skal ha et unikt design/program som er utformet av oss. Når det kommer til prosessen med å helle røre på jernet, tenker vi å utvikle en helt unik metode. Vi har i alle fall ikke sett noe tilsvarende på markedet i løpet av våre forstudier. Kort fortalt så har vi to beholdere med vaffelrøre, som er styrt av et slusesystem med to servoer, samt en fototransistor i den nederste beholderen. Detaljene i dette systemet vil bli nøye beskrevet i bachelor-rapporten.

Figur 2.1- Bilde 6 akset robot arm. https://www.mecademic.com/en/meca500-robot-arm

2.2 Forventninger

Det forventes å konstruere, og å designe en robot som klarer å gjøre de oppgavene den er tiltenkt. Samt å lage en robot som har unike løsninger og som holder seg innenfor lavest mulig kostnad.

Det forventes å gjennomføre de forskjellige oppgaver innenfor våre egne gitte tidsrammer(vist i Ganttdiagrammet i figur 3.3.1 og 3.3.2) , og få et godt, bredt læreutbytte av prosjektet.

3. Prosjektmål

- Velge materiale for konstruksjon av roboten. Her har vi allerede bestemt oss for en form for aluminiumlegering. Dette fordi det har en tredjedel av massetettheten til stål, mens det i tillegg er mer enn sterkt/seigt/strekkfast nok for vårt formål. Den eneste vektbærende oppgaven til robotarmen er tross alt bare å lukke/åpne lokket til vaffeljernet.
- Gjennomføre tilstrekkelig testing av roboten, før den evt. kan kommersialiseres i form av at lærerpersonalet kan ta den i bruk.
- Holde oss godt innenfor en fornuftig prisramme. Vi vet ikke nøyaktig hvor mye roboten kommer til å koste før den er ferdigstilt, og et fornuftig prisoverslag kan ikke lages før vi har bestemt oss for hvilke motorer(med påfølgende drivere og spenningskilde) vi går for. Men nye tyder likevel på at prisen for roboten, i tillegg til vaffeljern og redskaper, kommer til å havne på en plass imellom 3000-5000kr(eksklusiv aluminium).
- Velge PLS-verktøy å programmere i. Vi tenkte lenge å programmere PLS ved hjelp av Omron-PLS/HMI på elektrolab, men fant forut at kostnadene ville bli for høye, samt så ville det å betjene 230 volt medført en for stor risiko for to studenter som ikke har noen bred erfaring med kobling.
- En langt billigere løsning vil være å programmere PLSen ved hjelp av Arduino, i tillegg til en ekstern spenningskilde på trolig 24 volt, for å betjene motorene. Ved hjelp av et par motordrivere skal vi kunne finne motorer som er kraftige nok til å utføre de gitte oppgavene.
- Bestemme oss for et system for påføring av røre. Som nevnt i kapittel 2.1 har vi allerede bestemt oss for et system(gitt at det ikke feiler). Et alternativ til dette var at robotarmen betjente en øse som påførte vaffeljernet røre med. Men vi kom fort til en konklusjon at dette kom til å bli mye unødvendig søl. Og då måtte vi fort designet en vaskerobot også. Med den valgte løsningen håper vi å unngå alle former for søl.

3.1 Gjøremål

- Gjøre nødvendige forstudier. Dette fant vi fort ut at ville ta tid, og det ble derfor påbegynt allerede i uke 51. Tenker da spesielt på områder som vi ikke har beveget

oss så mye inn i under studiet. Dette gjelder f.eks Arduino, kobling, stegmotorer, servoer, drivere, kinematikk, med vekt på invers, i tillegg til å friske opp kunnskapene våre rundt dimensjonering av selve armen.

- Gjennomføre beregninger for å finne ut hvor kraftige motorer vi trenger.
- Finne ut hvilke typer motorer vi skal bruke til de forskjellige oppgavene. For presisjonens skyld tyder det meste på at det blir en kombinasjon av stegmotorer og servoer. Finne passende drivere, samt en fornuftig spenningskilde.
- Sette oss inn i og forstå oppkoblinger av motor til drivere(om nødvendig), og deretter til kontroller. Teste at alle koblinger fungerer som de skal.
- Velge kontroller. F.eks Arduino Uno- eller Mega.
- Før vi kan begynne på dimensjoneringen er vi nødt til å avgjøre hvor roboten skal befinne seg. På pauserommet eller et annet egnet sted? Dette må vi evt. avklare med fagansvarlig/veileder. Når vi har funnet et egnet sted, er vi nødt til å finne ut hva de maksimale ytre målene kan være. Då må vi ta med robotarm, fastmontert vaffeljern, slusesystem og et evt. sikringsskap/koblebrett med i beregningene.
- Etter vi har tatt de ytre maksimale målene, må vi avgjøre hvor mange grader av frihet robotarmen skal ha(antall akser, og ikke grad av frihet man tenker på i f.eks FEM). Vi må derfor avgjøre hvilke betingelser som er nødvendige for funksjonaliteten til robotarmen.

  Robotarmen skal kunne bevege seg rettlinjet i rommet, samt kunne rotere om x-, y- og z-aksen. En seksakset robotarm innfrir alle disse betingelsene. Den kan nesten oppføre seg som en menneskehånd, bevege seg fritt i rommet, samt utføre det som på engelsk heter «yaw»-, «pitch»- og «roll»-bevegelser[1]. Som er nøyaktig det samme som rotasjon om aksene i rommet. Det er i tillegg er denne type robotarmen som er vanligst i industrien[1]. Etter samtaler med veileder/fagansvarlig kan det godt være vi ombestemmer oss, dersom argumentene for flere eller mindre grader av frihet er gode.
- Dimensjonere robotarmen for å tilpasse den til motorene, samt å tåle spenningene, med nødvendig sikkerhetsfaktor.
- Designe robotarmen i Autodesk Inventor.
- Konstruere robotarmen på verkstedet.
- Sette sammen robotarmen med motorer og koblinger. Her må vi nok ta et koblingsskap i bruk.

- Teste robotarmen med litt enkel kinematikk.
- Dimensjonere motorer/servoer til slusesystemet utifra trykket vaffelrøren vil påføre.
- Finne riktig sensor til slusesystemet, og teste denne. Sjekke analoge verdier med Serial-monitor i Arduino.
- Designe slusesystemet i Autodesk Inventor.
- Konstruere slusesystem på verksted, og sette det sammen med servoer/sensor og koblinger.
- Utarbeide en nøyaktig modell for invers kinematikk i Excel. ¨-Hvilke vinkler må de forskjellige leddene ha for at gripearmen skal ha en gitt posisjon?
- Teste kinematikken.
- Fullføre programmet i Arduino, og teste roboten.
- Om tiden strekker til: Tilføre eventuelle sideeffekter som LCD-skjerm, dyse til robotarm for å smøre vaffeljernet med fett, LED-lys m.m.
- Dokumentere hele prosessen i dokumentet for Bachelor-prosjektet, og fullføre dette før 15.05.2022!
- I tillegg til mange flere utfordringer som enn måtte oppstå foran PC-skjermen eller på verkstedet.

3.2 WBS-aktivitetsdiagram

3.3 Ganttdiagram

Gant chart

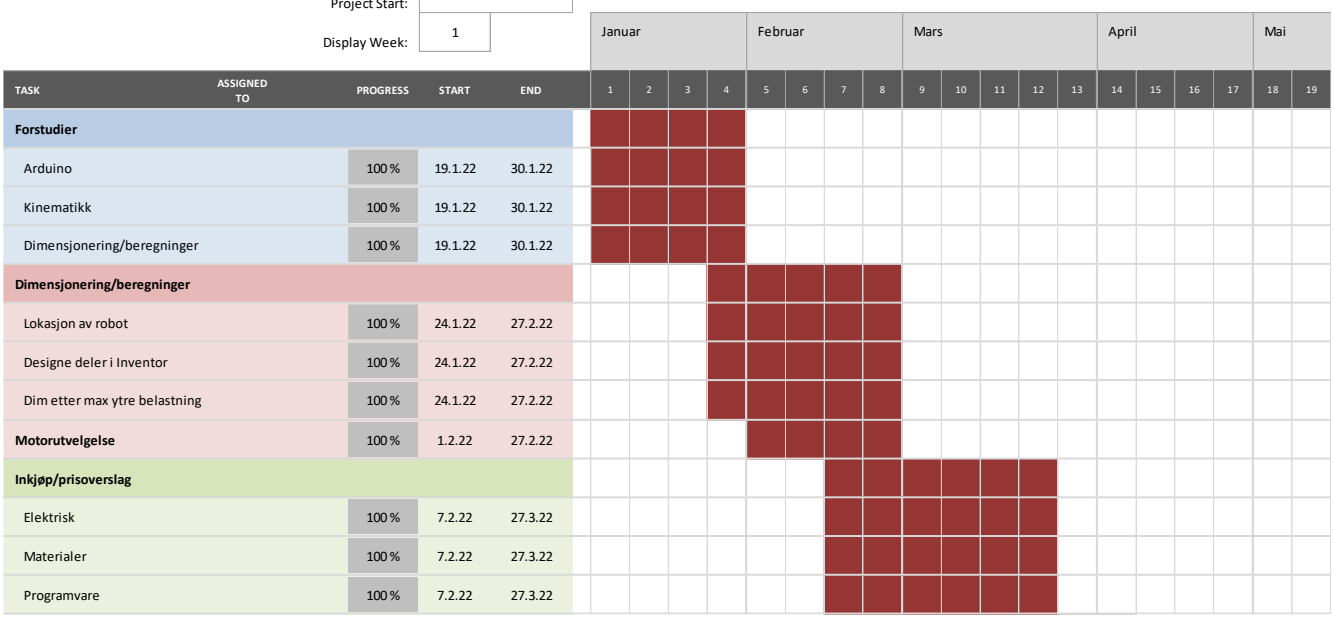
SIMPLE GANTT CHART by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Design og konstruksjon av vaffelrobot

Project Start: ons, 19.1.22

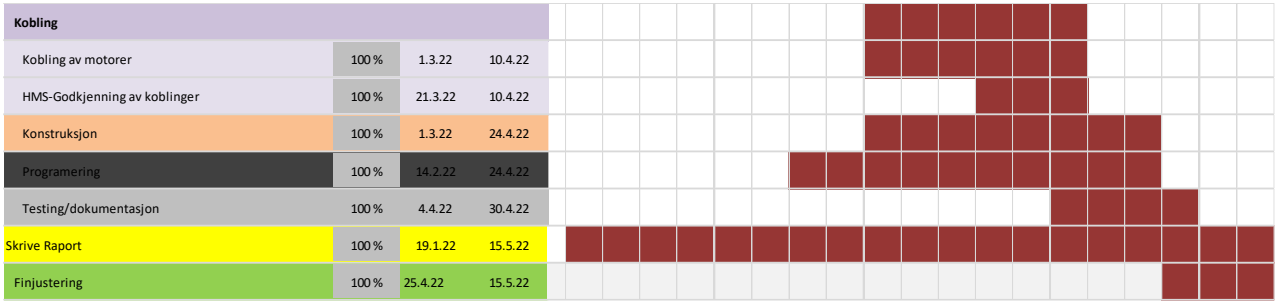
Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END
Forstudier				
Arduino		100 %	19.1.22	30.1.22
Kinematikk		100 %	19.1.22	30.1.22
Dimensjonering/beregninger		100 %	19.1.22	30.1.22
Dimensjonering/beregninger				
Lokasjon av robot		100 %	24.1.22	27.2.22
Designe deler i Inventor		100 %	24.1.22	27.2.22
Dim etter max ytre belastning		100 %	24.1.22	27.2.22
Motorutvelgelse		100 %	1.2.22	27.2.22
Inkjøp/prisoverslag				
Elektrisk		100 %	7.2.22	27.3.22
Materialer		100 %	7.2.22	27.3.22
Programvare		100 %	7.2.22	27.3.22



Figur 3.3.1-Første del av Ganttdiagram

Kobling			
Kobling av motorer	100 %	1.3.22	10.4.22
HMS-Godkjenning av koblinger	100 %	21.3.22	10.4.22
Konstruksjon	100 %	1.3.22	24.4.22
Programering	100 %	14.2.22	24.4.22
Testing/dokumentasjon	100 %	4.4.22	30.4.22
Skrive Raport	100 %	19.1.22	15.5.22
Finjustering	100 %	25.4.22	15.5.22



Figur 3.3.2-Andre del av Ganttdiagram

4. Eksisterende teknologi og teori

Det første vi gjorde da vi ble tildelt bacheloroppgaven var selvsagt å utforske hvilke tilsvarende løsninger som allerede eksisterte. Noe av det første vi kom over var to Facebook-videoer. Den ene viser en vaffelrobot utviklet av bachelorstudenter på Oslo Met[2]. Her ser vi en tilsynelatende overdimensjonert robotarm som betjener diverse redskaper. Men den trenger menneskelig assistanse, og påføringen av røre med øse medfører mye søl.

I den andre videoen ser vi en robotarm som automasjons-vg2-elever på Nord-Østerdal vgs har designet[3]. Også her blir røren påført med en øse, fra en bolle. Denne roboten innehar, iøynefallende, bedre kvalitet enn den som de nevnte bachelorstudentene konstruerte[2]. Det er langt mindre søl, armen er designet mer minimalistisk, og presisjonen virker å være enda bedre. Likevel: Også her må robotarmen hjelpes ved at et menneske tar vaffelen av jernet.

I en tredje video, som ligger ute på YouTube, har to mekatronikk-studenter konstruert sin versjon av en vaffelrobot[4]. Det som er unikt her er at gripearmen fungerer med 3 ulike funksjoner: En øse, en løftekrok og en gaffel til å ta vaffelen av jernet. Designet av selve robotarmen virker å være enkelt, men samtidig robust. Men også her har robotarmen problemer med å få vaffelen av jernet, og et forsøk på å klippe det bort fra videoen skjuler ikke det faktum. Det er med andre ord mye som tyder på at en gaffel ikke er løsningen når det kommer til å få vaffelen av.

I en siste video dukker det opp en løsning som virker å fungere mye bedre enn de tidligere nevnte. En YouTube-bruker med navnet «Bård» har designet en robotarm hvor han løfter vaffelen av med en skivemekanisme som har pigger/kroker under[5]. Robotarmen blir rotert et visst antall grader for å feste skiven til vaffelen, før den roterer tilbake den samme vinkelen for at vaffelen skal slippe taket.

For vår del blir det å ta det beste fra alle verdener, samtidig som vi har vårt eget unike design. Vi vil designe en robotarm etter en mal/standard beskrevet hos f.eks. robotsdoneright.com [1]. Som tidligere nevnt går vi for vårt eget unike slusesystem, og ikke øse-versjonen som vi har sett i de tidligere videoene. Unikt i den grad at vi ikke har sett tilsvarende løsninger på markedet. At det finnes et hundretalls tilsvarende system på diverse fabrikker i Kina kan vi ikke utelukke. Vi vil også gå for et så enkelt design som mulig, da det skal være prisbesparende, og det er funksjonalitet som er viktigst. Gjerne inspirert av mekatronikk-studentene [4], men selvfølgelig utviklet 100% av oss selv. Når det kommer til en mekanisme som løfter vaffelen kommer vi nok til å la oss inspirere av «Bård»[5]. Men vi kommer nok til å gå for et litt mer profesjonelt design(om tiden strekker til), i tillegg til at vi trolig kommer til å lage den av noe annet enn tre. Hovedmålet med robotarmen blir følgelig at den skal være inspirert av eksisterende teknologi, men hundre prosent unik, og forhåpentligvis et bedre og mer selvgående produkt.

5. Kilder

[1]What is a Six-Axis Robot? (robotsdoneright.com) , Robots Done Right: “What is a Six-Axis Robot?”, lastet ned 24.01.2022

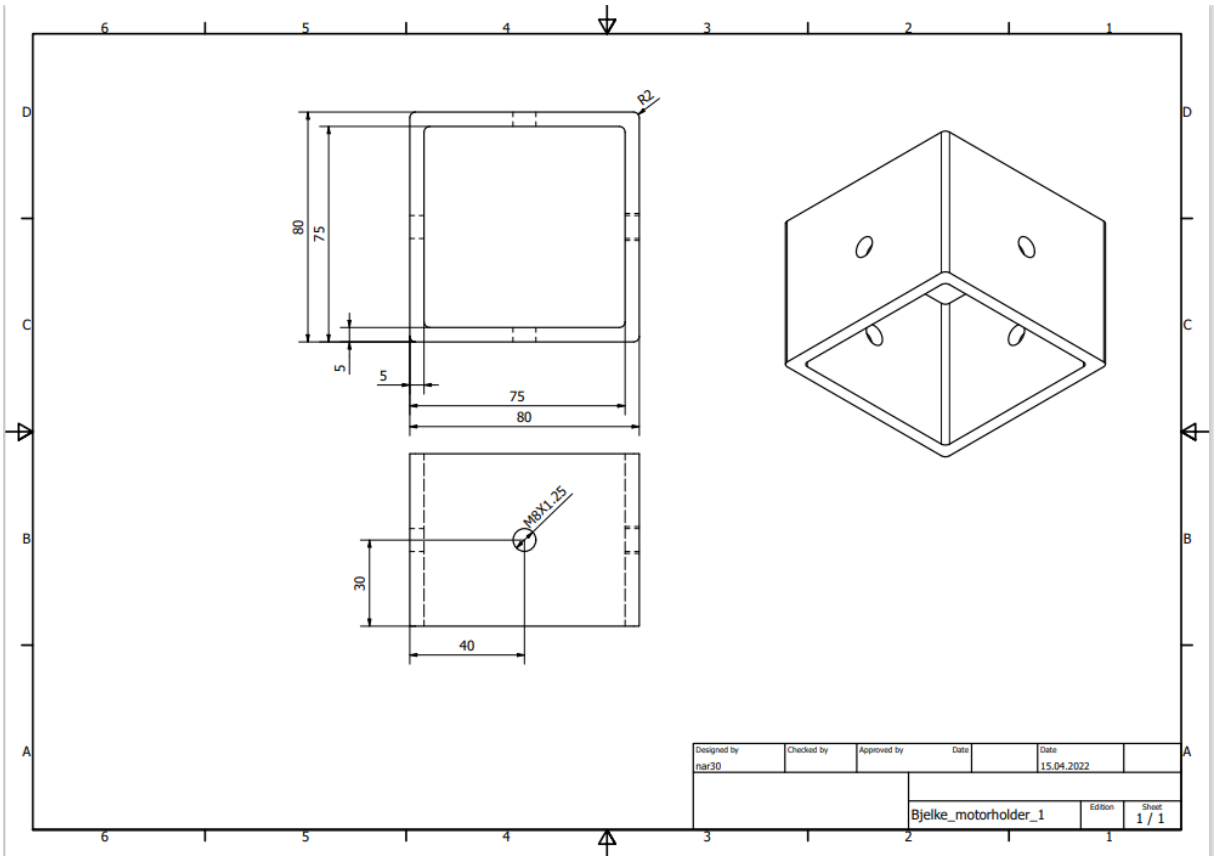
[2] Facebook , Ingeniør- og datafag, Oslo MET Facebook-side, ”Robotteknikk - vaffelsteking» , lastet opp 12.11.2015, lastet ned 24.01.2022

[3] Watch | Facebook , Nord-Østerdal videregående skole-NØVGS Facebook-side, «VG2 automasjonselever har laget vaffelrobot», lastet opp 28.11.2019, lastet ned 24.01.2022

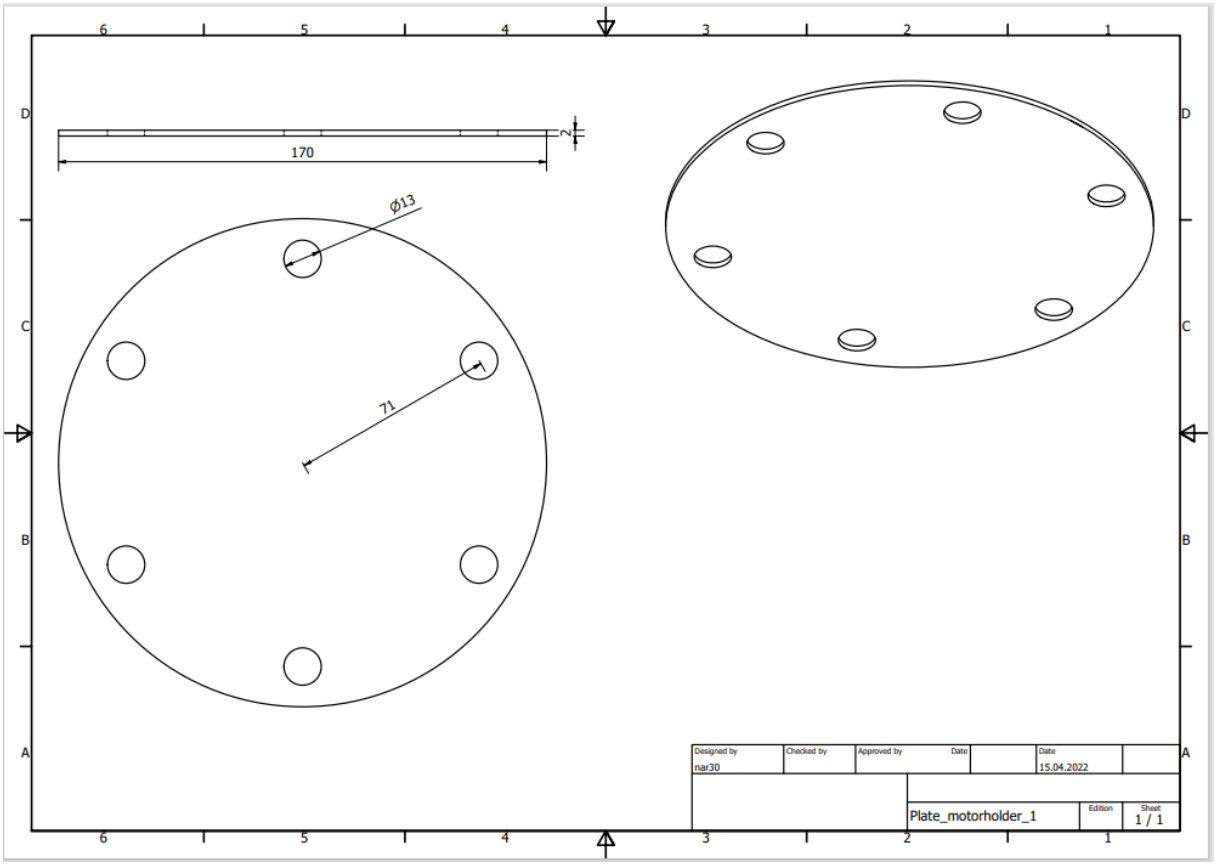
[4]Waffle making robot - Vaffelstekende robot - YouTube , Annam Rai, «Waffle making robot-Vaffelstekende robot», lastet opp «over 6 år siden(YouTube oppgir ikke eksakt dato)», lastet ned 24.01.2022

[5] UR10 Robot Computer Vision for making Waffles - YouTube , Bård, “UR10 Robot Computer Vision for making Waffles”, lastet opp 23.10.2017, lastet ned 24.01.2022

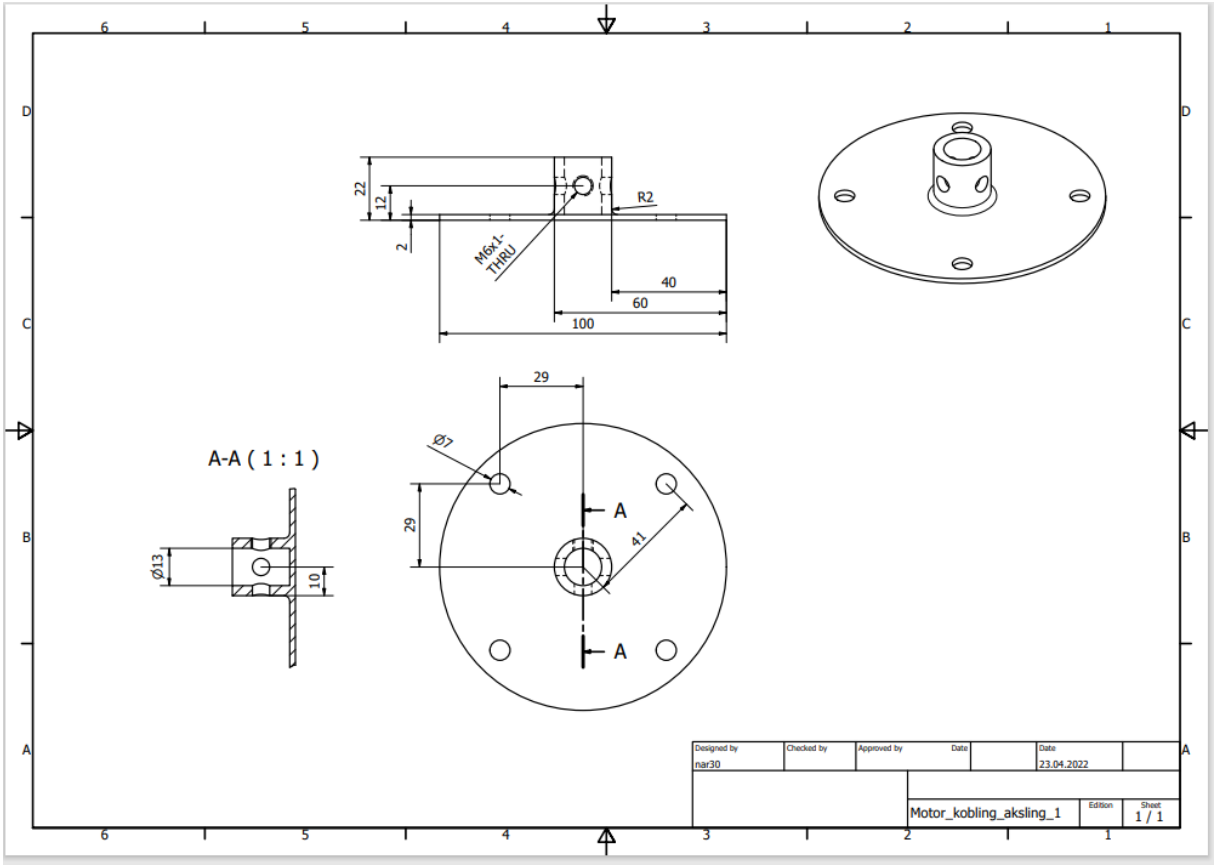
Vedlegg B-Arbeidstegninger



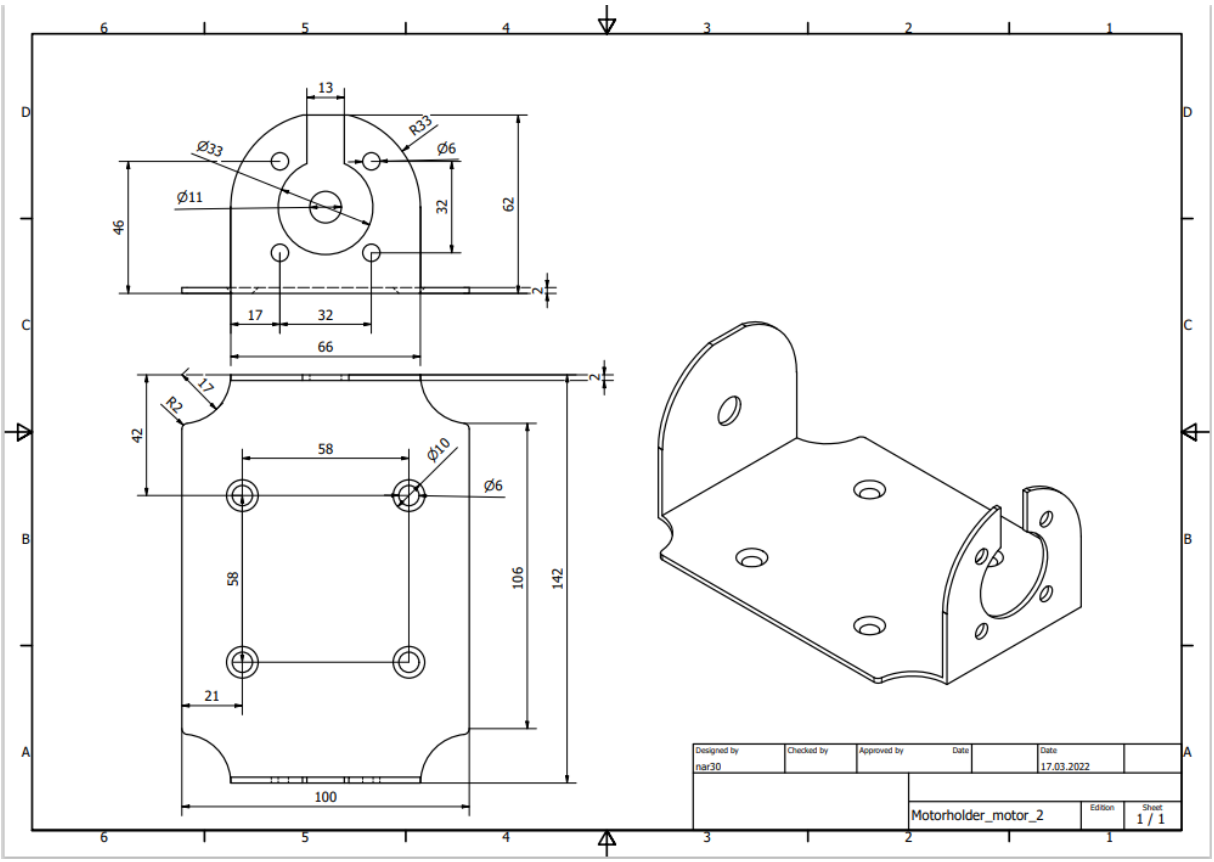
Figur B1-Bjelke til motorholder 1.



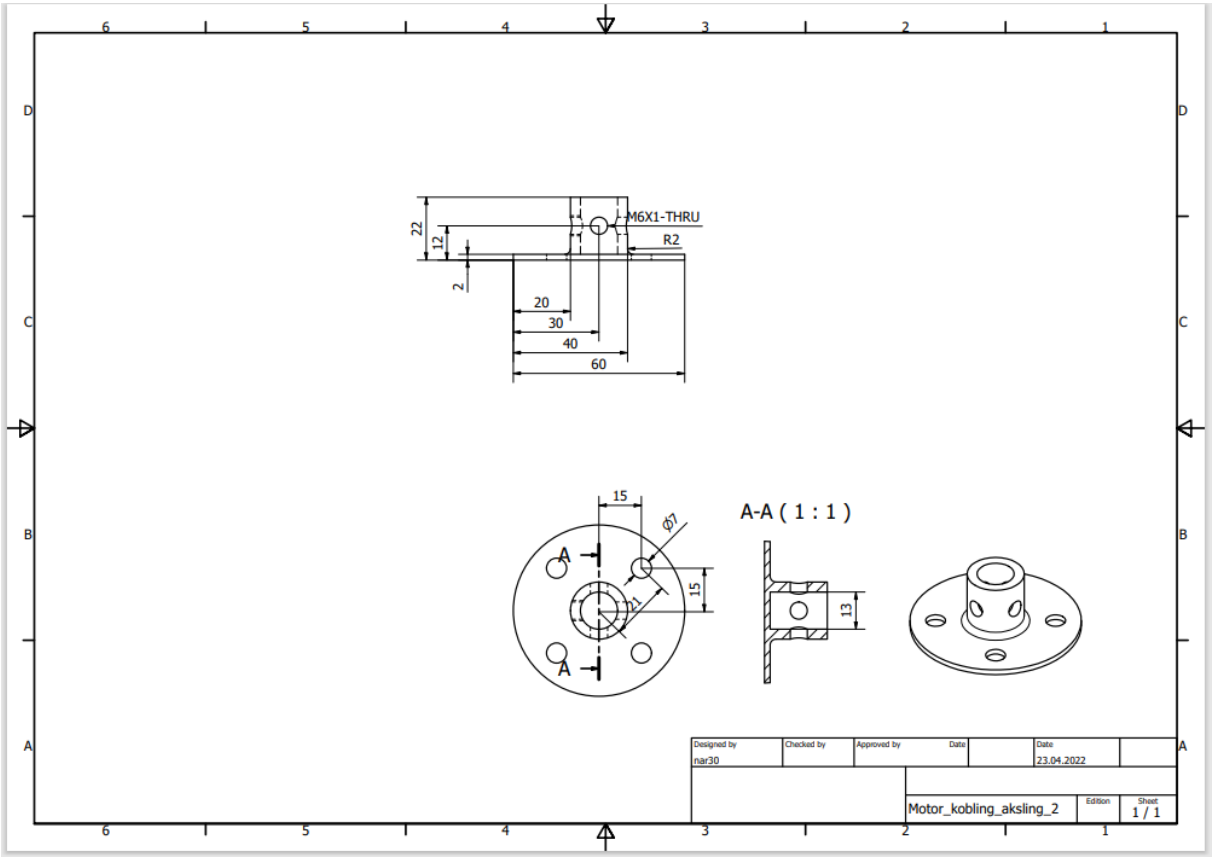
Figur B2-Plate til motorholder 1.



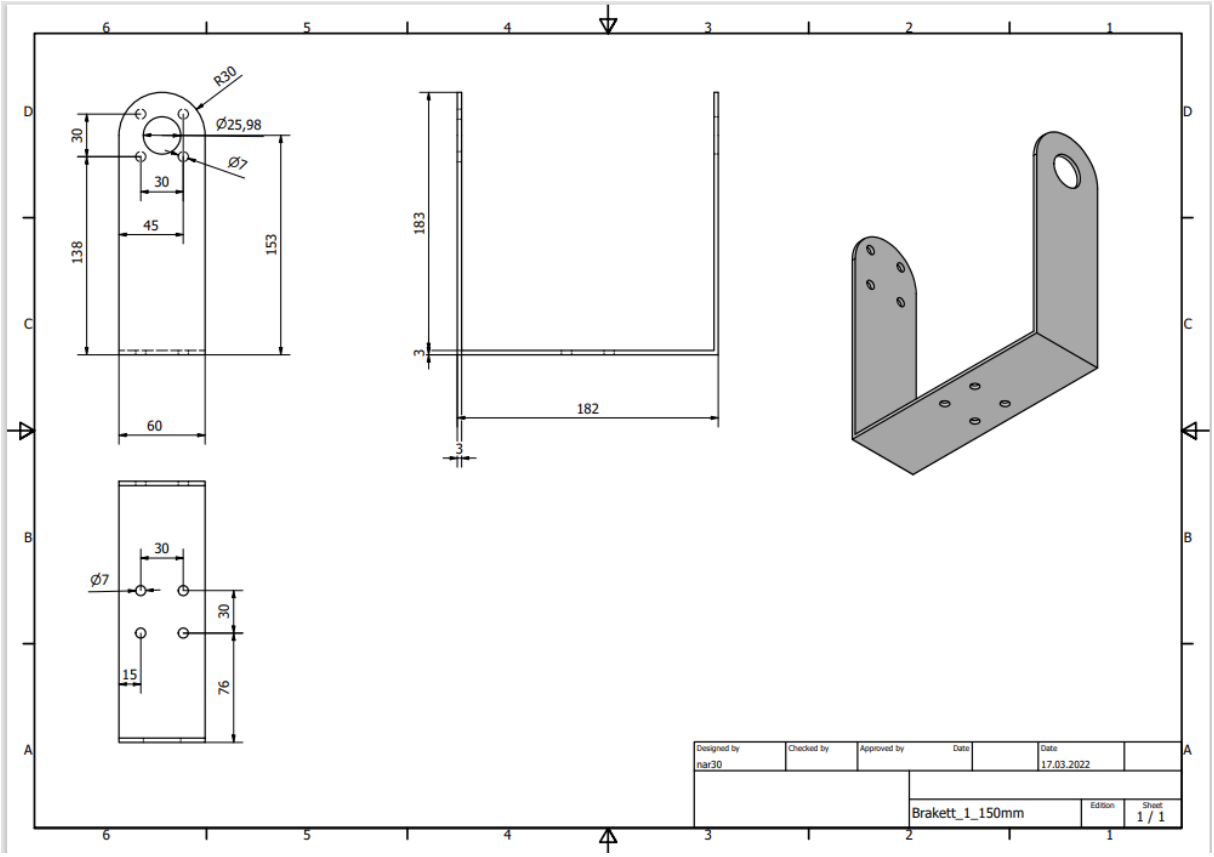
Figur B3-Sammenkoblingsdel for aksling til motor 1.



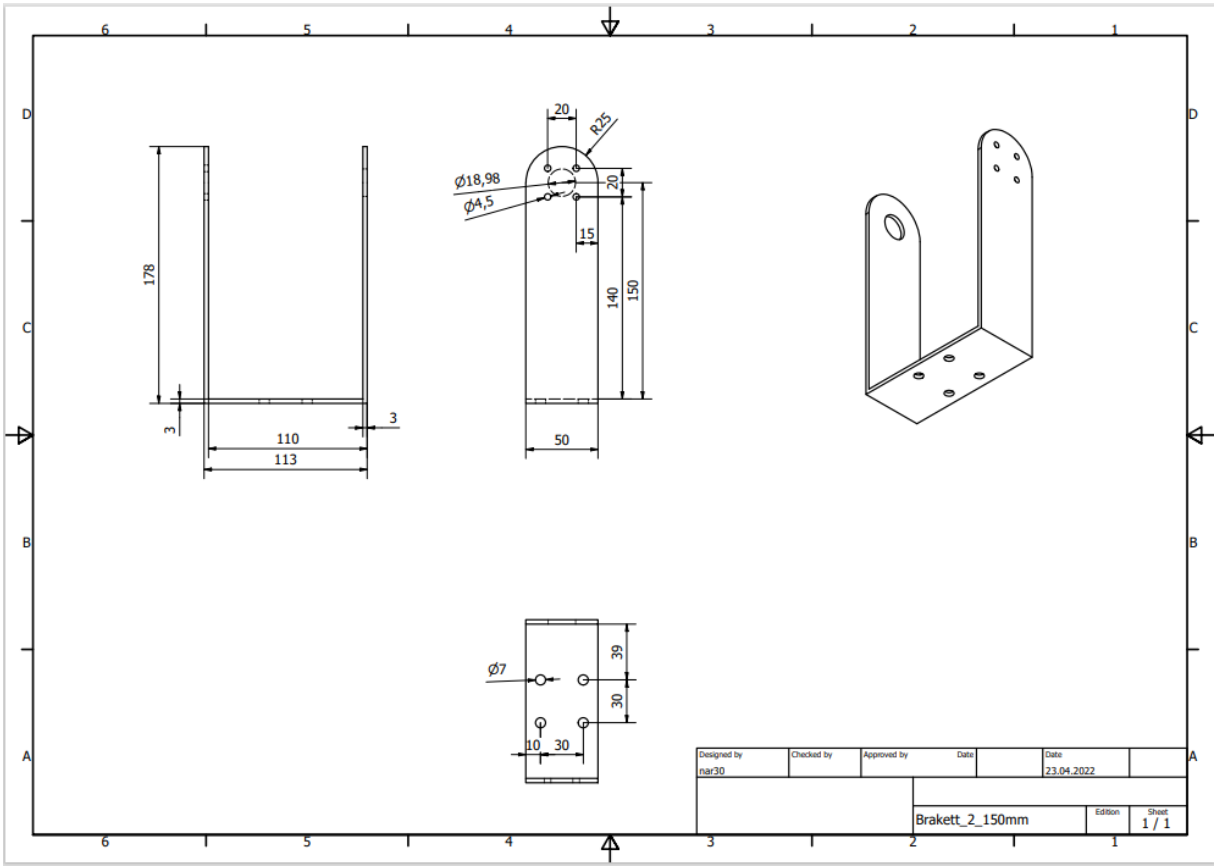
Figur B4-Motorholder til motor 2.



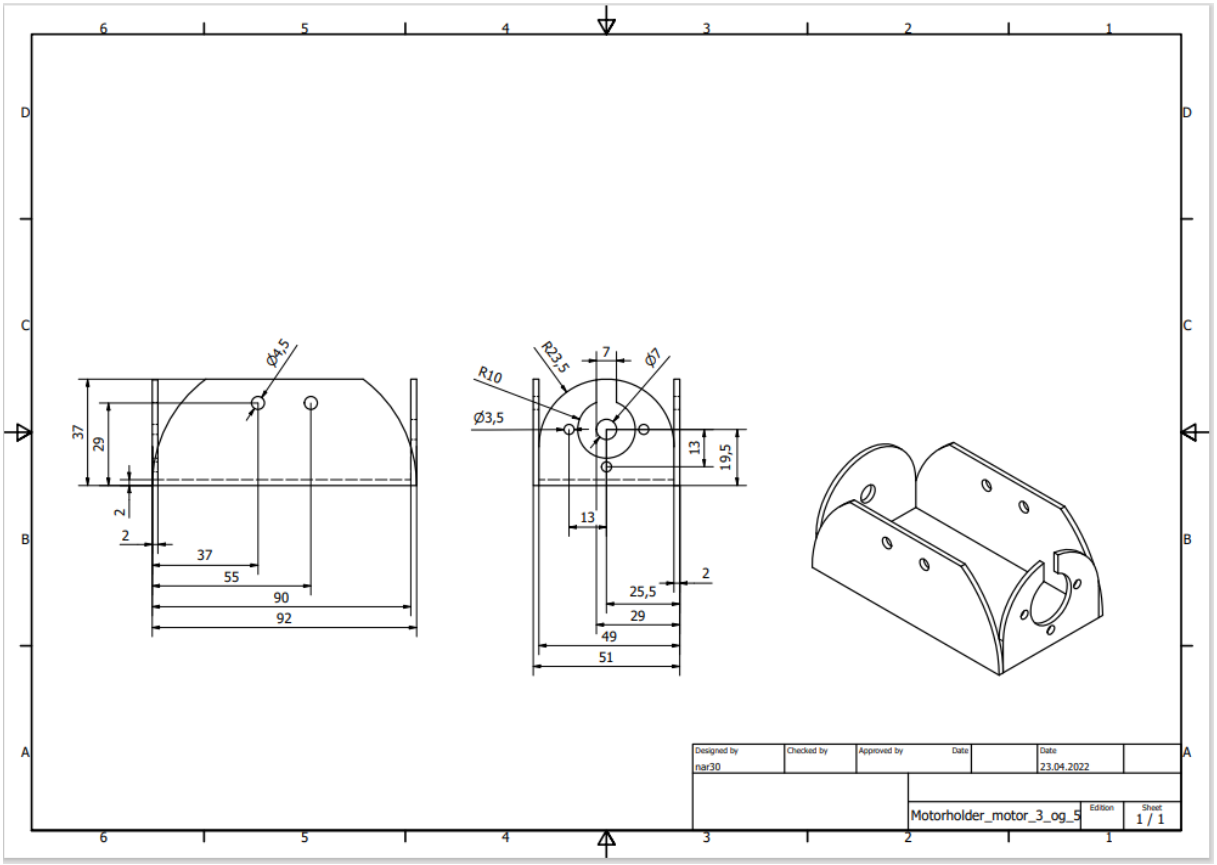
Figur B5- Sammenkoblingsdel for aksling til motor 1.



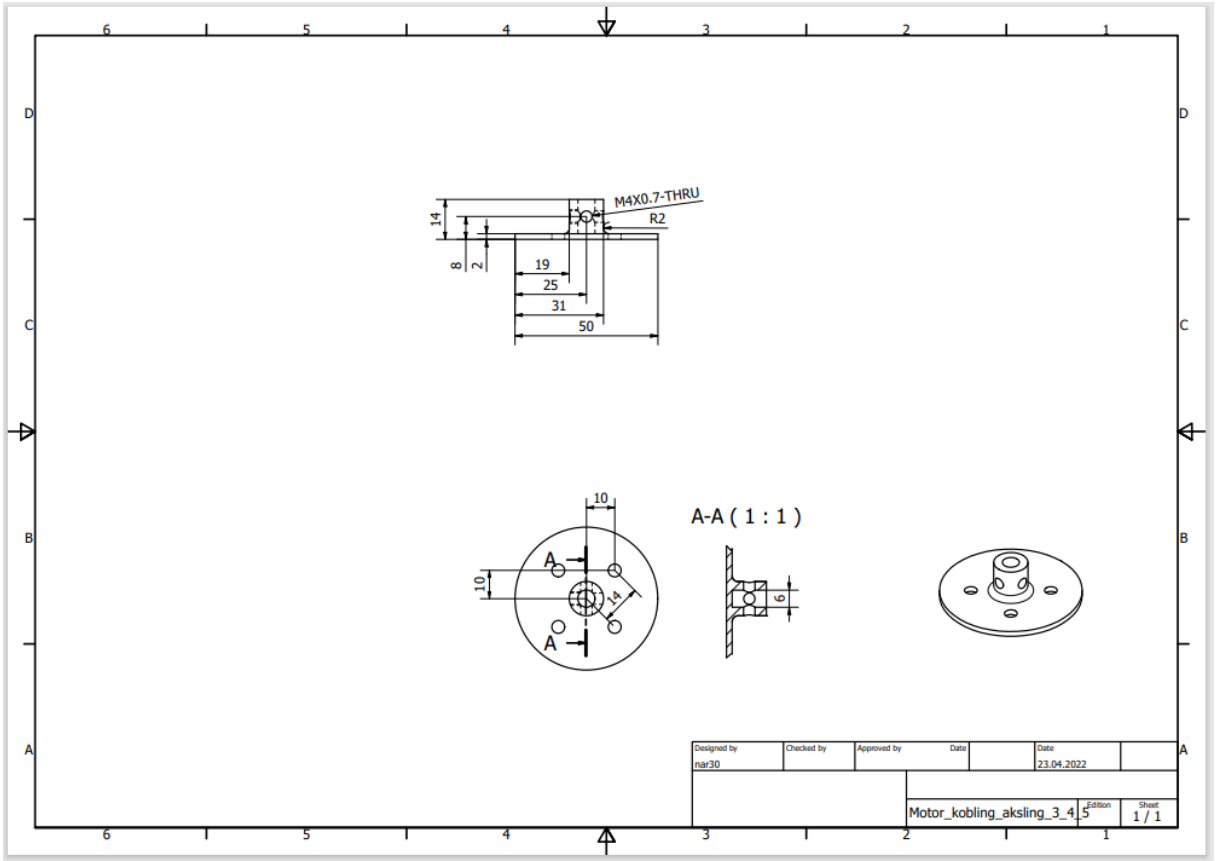
Figur B6-U-brakett 1, festet til sammenkoblingsdel for aksling på motor 2.



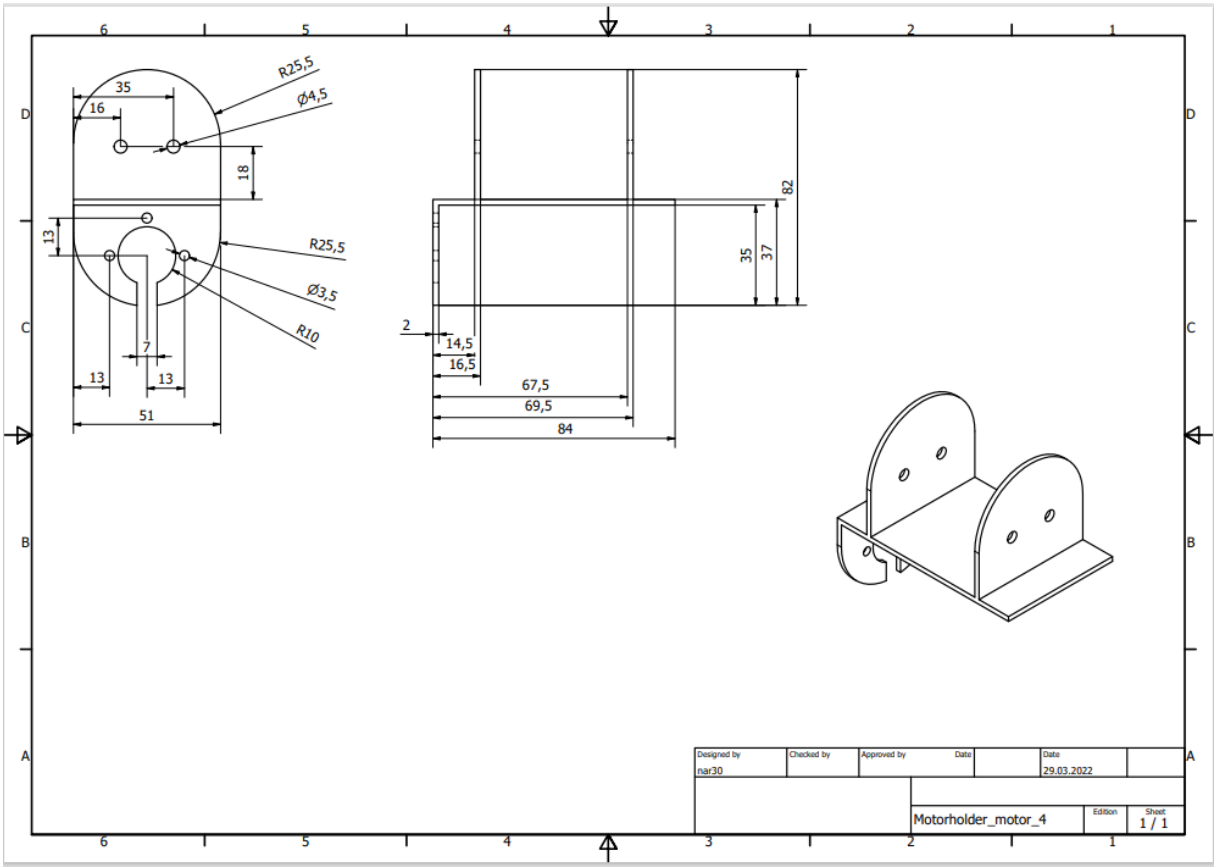
Figur B7- U-brakett 2, festet til sammenkoblingsdel for aksling på motor 3.



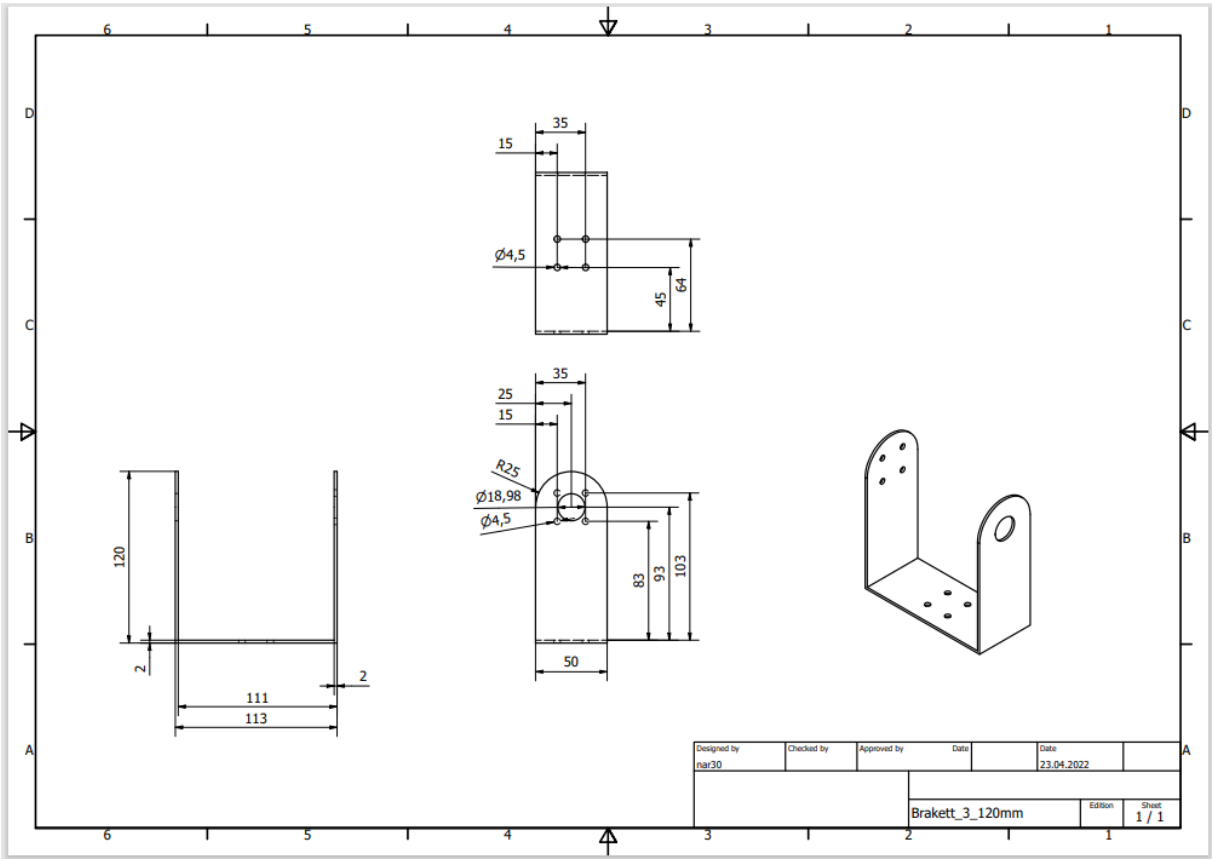
Figur B8-Motorholder for motor 3 og 5.



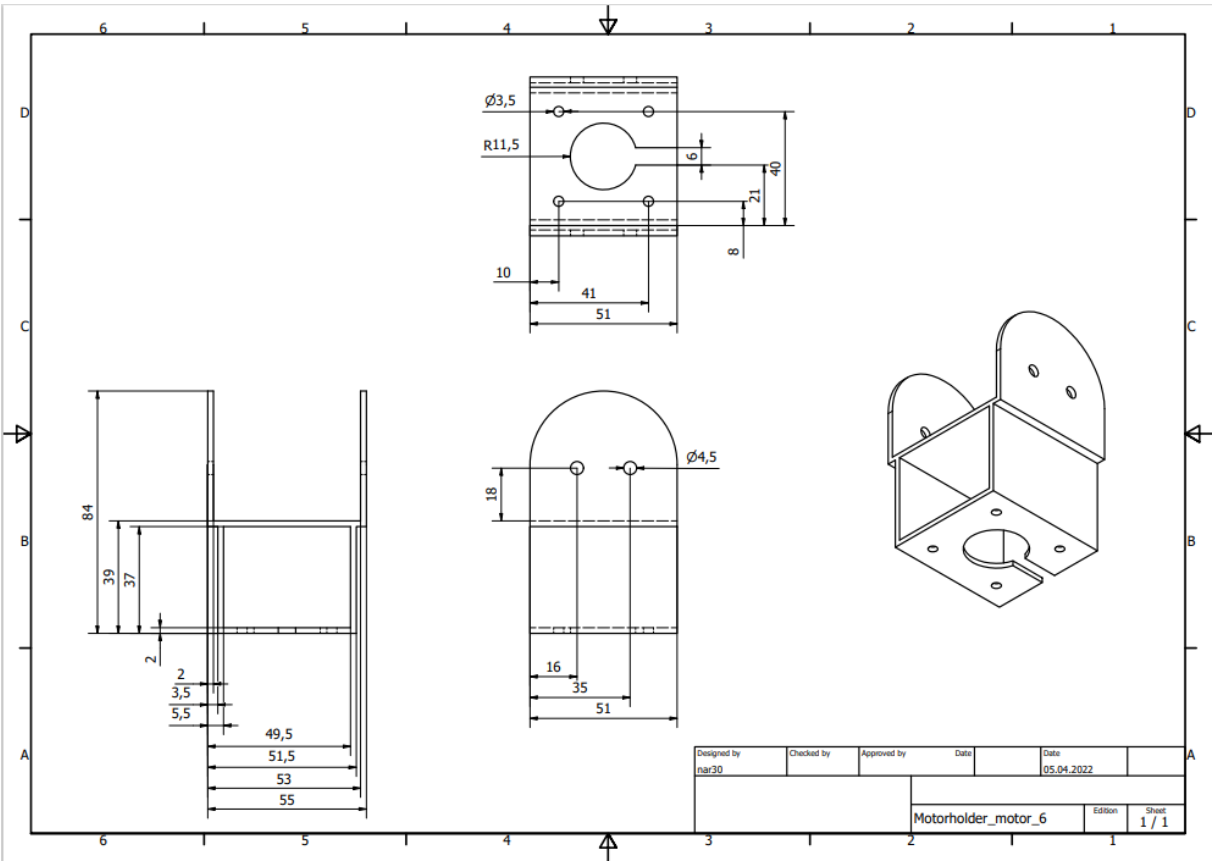
Figur B9-Sammenkoblingsdel for akslinger til motor 3,4 og 5.



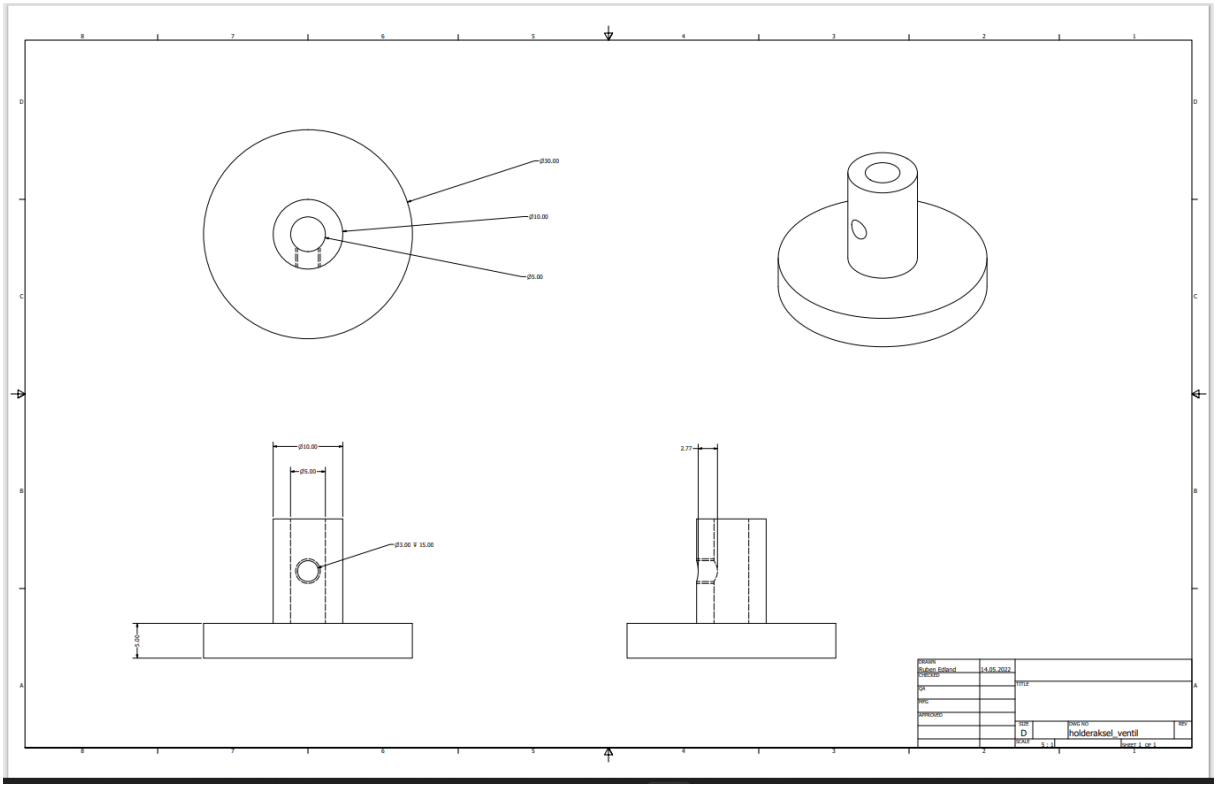
Figur B10-Motorholder 4.



Figur B11- U-brakett 3, festet til sammenkoblingsdel for aksling på motor 4.



Figur B12-Motorholder 6.



Figur B13-Sammenkoblingsdel for å feste ventil på aksling til motor 6.

Vedlegg C-Diverse tabeller

Tabell C1-Dreiemomenter(Nm) etter modifisering av motor 5

	Posisjon(pos) 1	Pos 2	Pos 3	Pos 4	Pos 5
T1	-2.62	0.32	-0.21	-1.75	-0.47
T2	-1.5	3.97	6.07	3.2	7.37
T3	-1.14	0.31	-0.02	0.27	2.17
T4	1.62	0.61	0.58	1.27	1.24
T5	0.13	0.84	1.35	0.19	0.03
T6	0	0	0	0	0

Tabell C2-Dreiemomenter(Nm) etter modifisering av motor 4

	Posisjon(pos) 1	Pos 2	Pos 3	Pos 4	Pos 5
T1	-2.26	0.43	-0.01	-1.39	-0.29
T2	-1.38	3.48	5.34	2.78	6.45
T3	-1.01	0.38	0.1	0.35	2.22
T4	1.53	0.6	0.58	1.2	1.15
T5	0.08	0.74	1.22	0.11	-0.02
T6	0	0	0	0	0

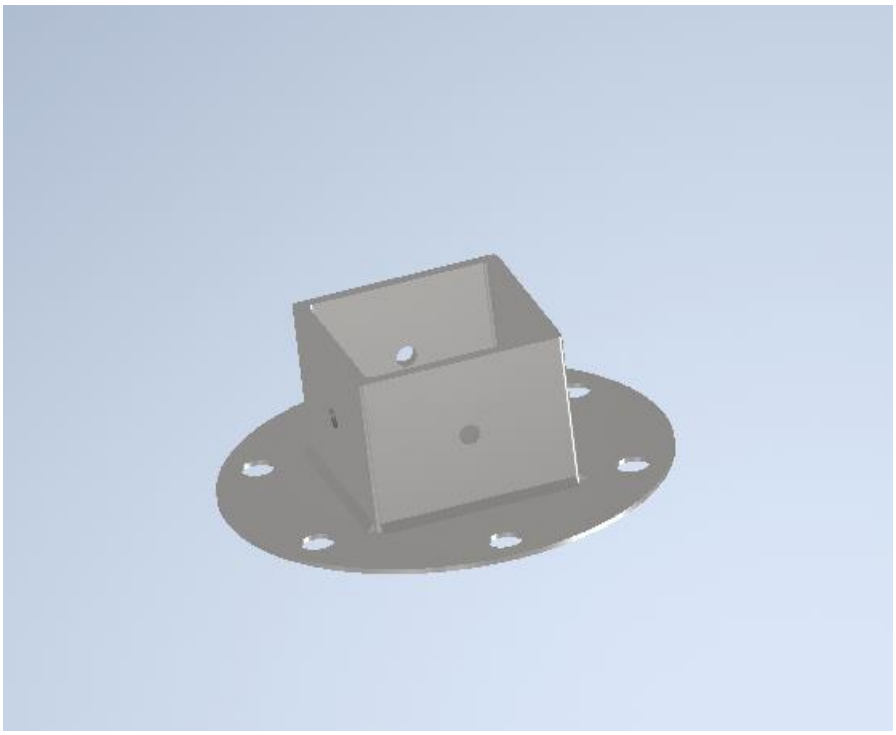
Tabell C3-Dreiemomenter(Nm) etter modifisering av motor 3

	Posisjon(pos) 1	Pos 2	Pos 3	Pos 4	Pos 5
T1	-2.38	0.28	-0.43	-1.5	-0.49
T2	-1.43	3.5	5.79	2.78	6.85
T3	-0.99	0.28	-0.09	0.26	2.21
T4	1.54	0.61	0.43	1.19	1.09
T5	0.1	0.78	1.38	0.18	0.09
T6	0	0	0	0	0

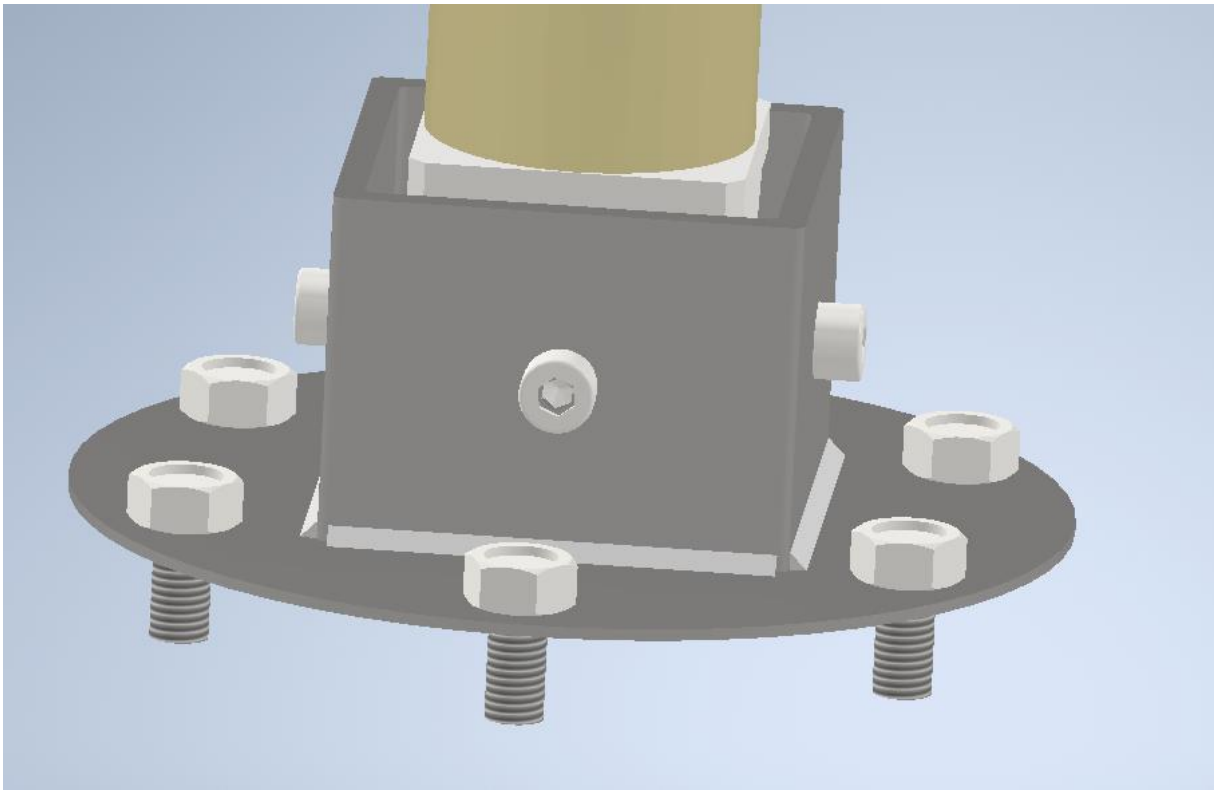
Tabell C4-Dreiemoment (Nm) etter modifisering av motor 2

	Posisjon(pos) 1	Pos 2	Pos 3	Pos 4	Pos 5
T1	-2.48	0.16	-0.63	-0.46	-0.6
T2	-1.4	3.45	5.8	3.21	6.86
T3	-0.98	0.26	-0.13	-0.58	2.22
T4	1.54	0.62	0.38	0.84	1.08
T5	0.11	0.82	1.44	-0.3	0.11
T6	0	0	0	0	0

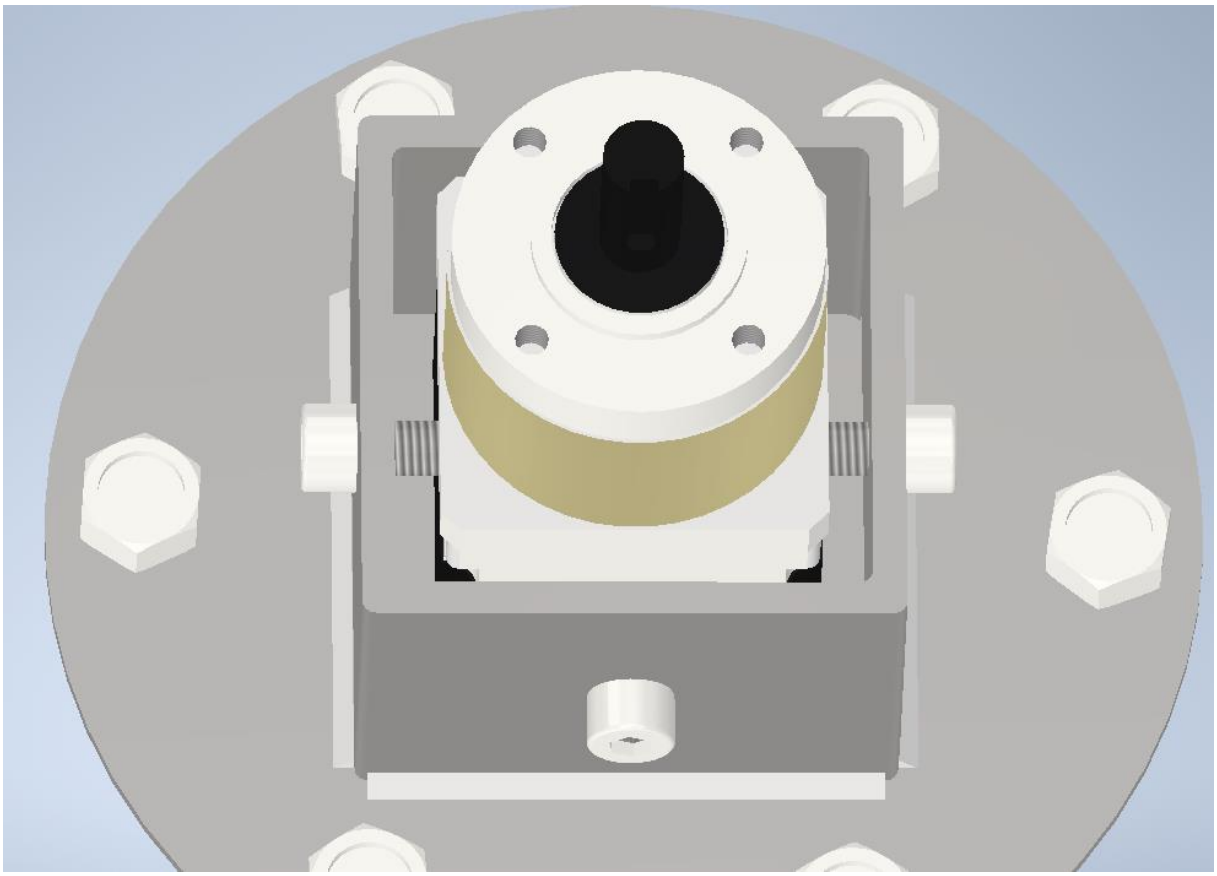
Vedlegg D-Diverse figurer



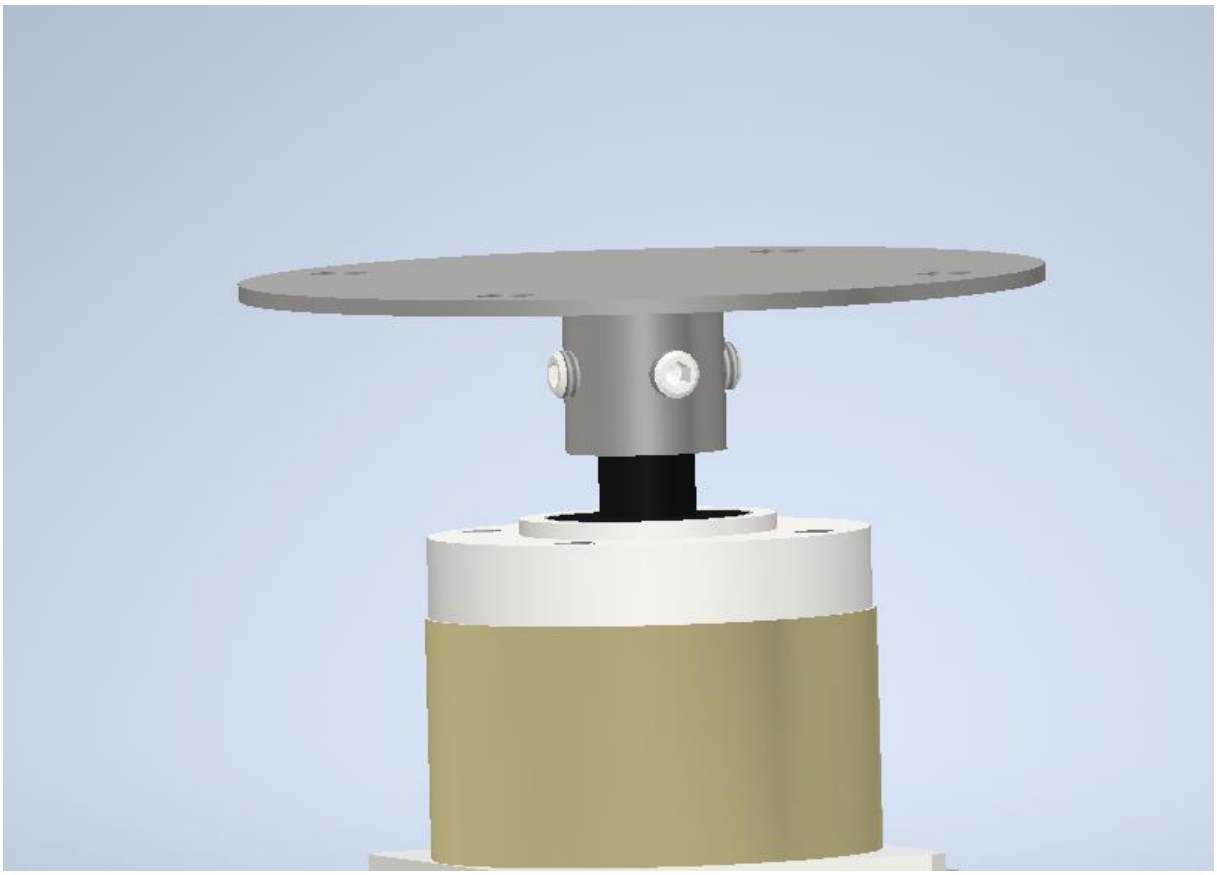
Figur D1-Motorholder 1, bjelke sveiset fast på plate.



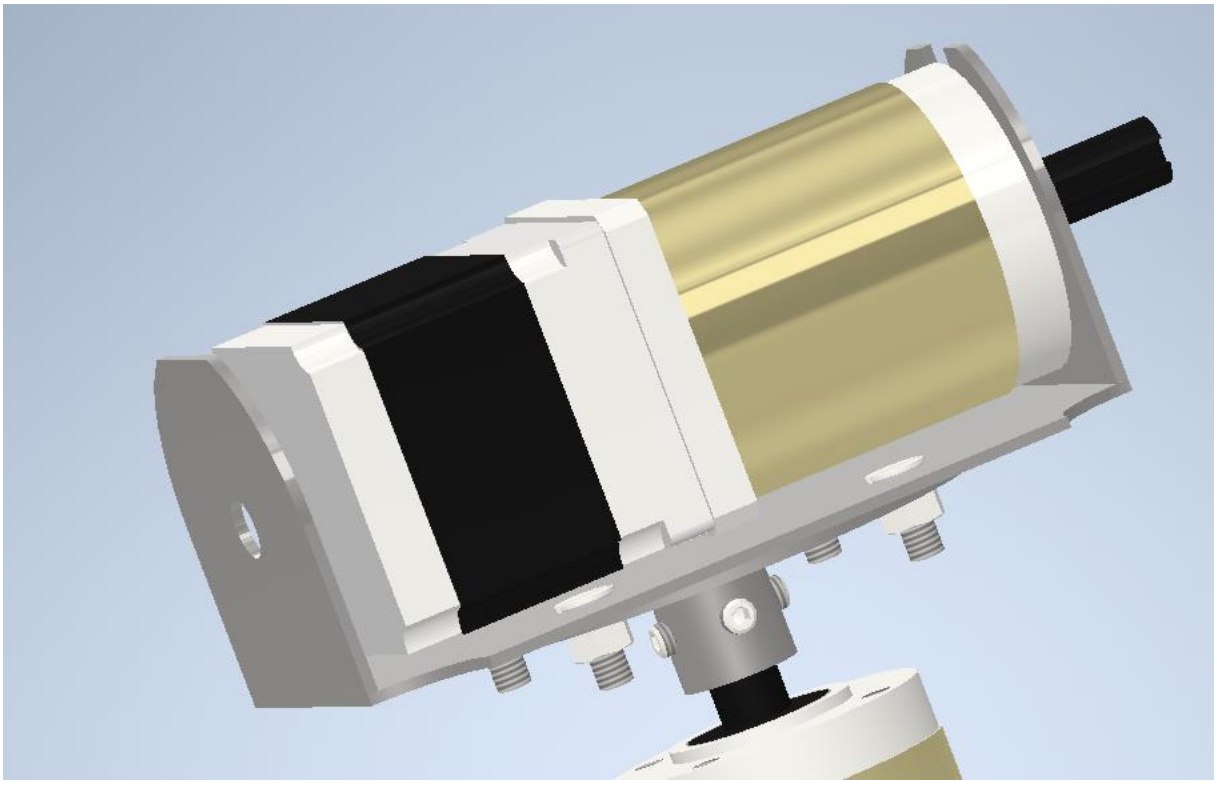
Figur D2-Skrueforbindelse til benk



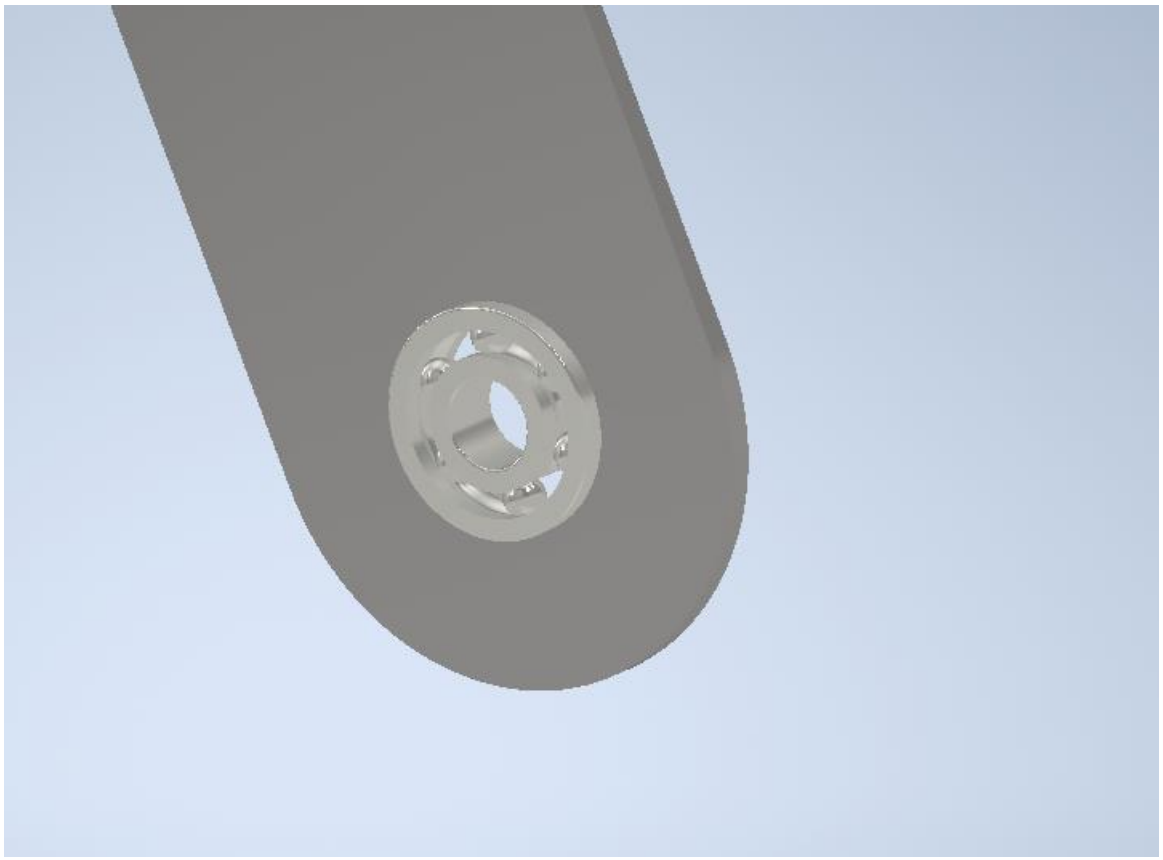
Figur D3-Skrueforbindelse som festet motor 1 til motorholder 1.



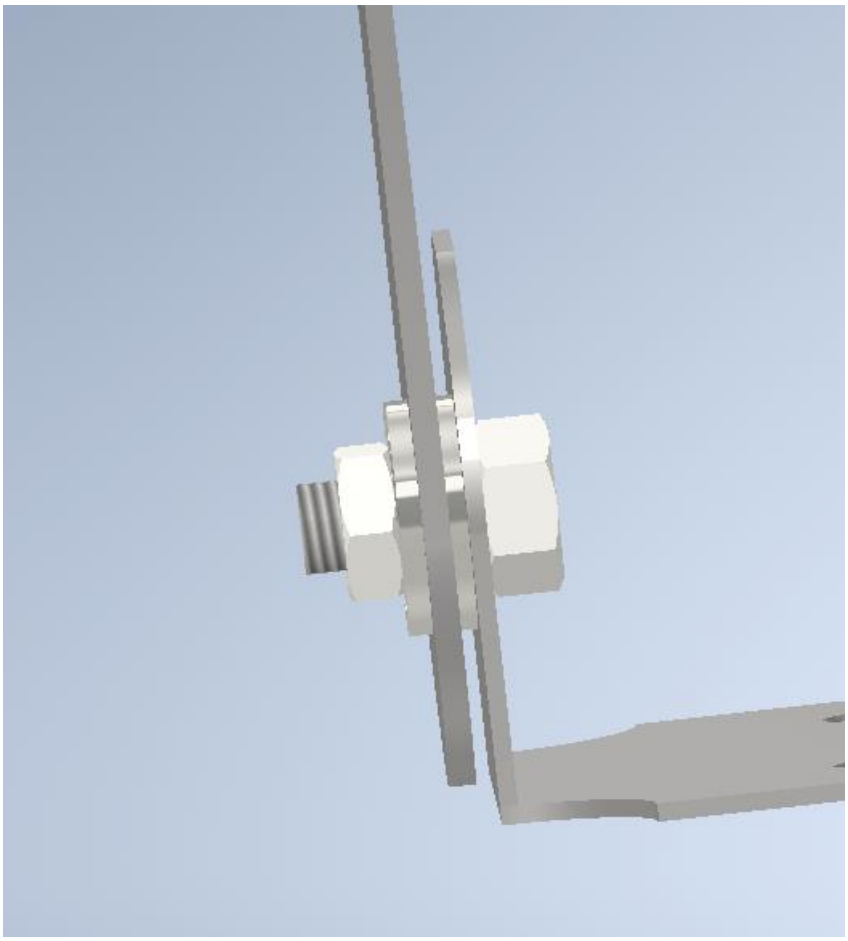
Figur D4-Settskrueforbindelse til aksling for motor 1



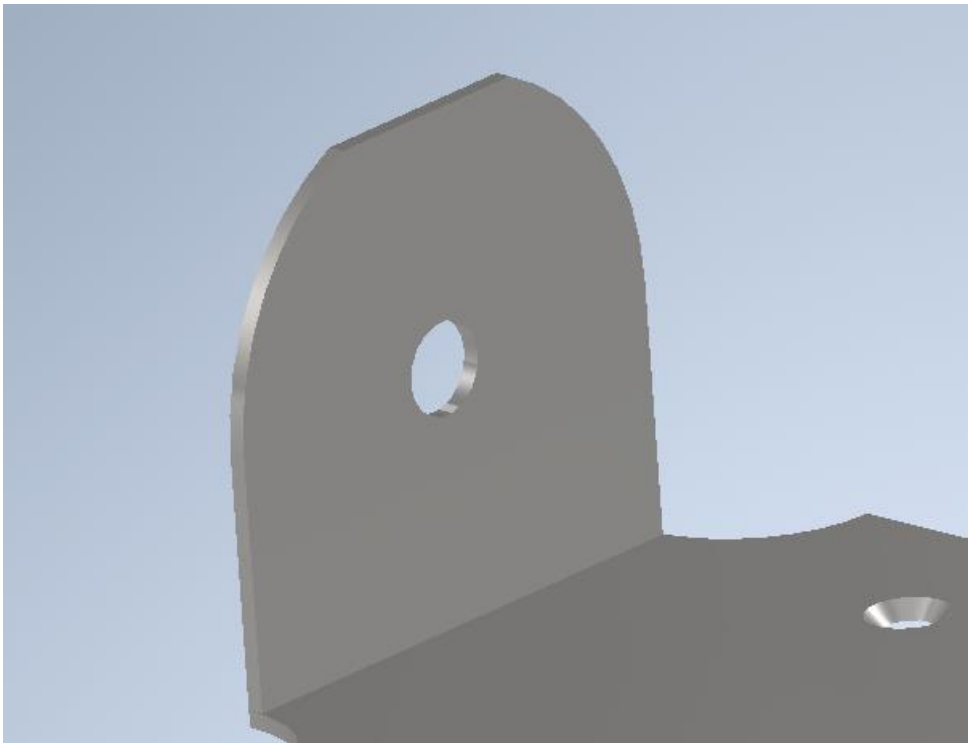
Figur D5-Skrueforbindelse mellom aksling motor 1 og motorholder 2.



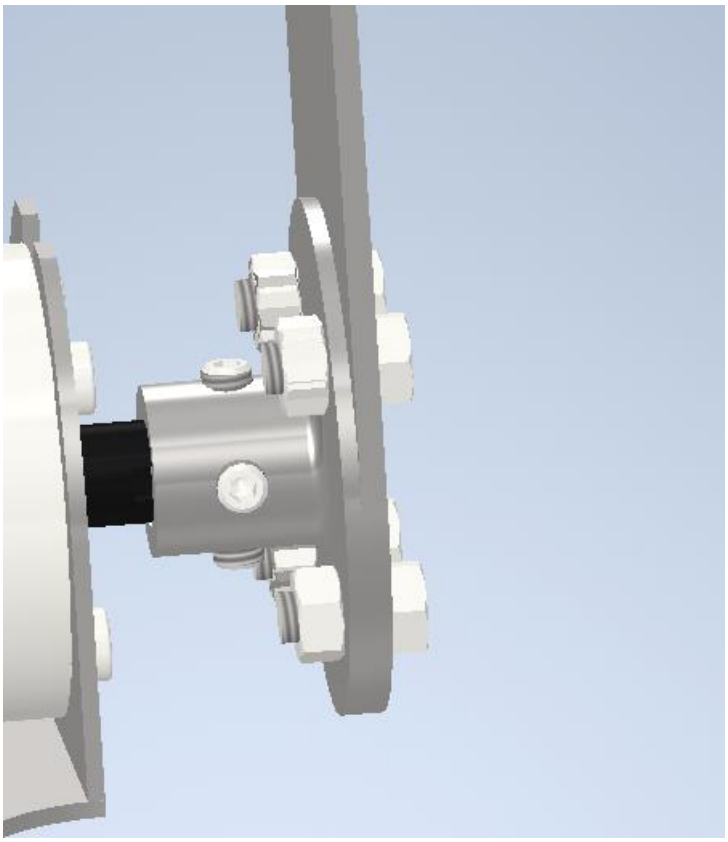
Figur D6-Krympeforbindelse av lager på brakett.



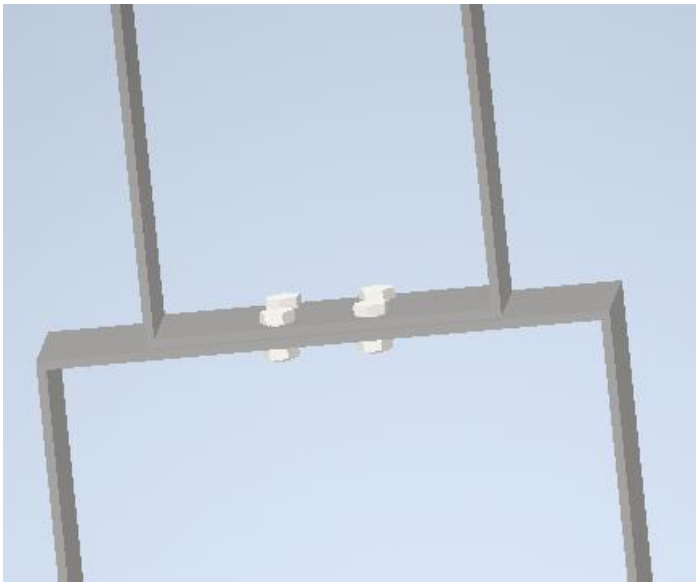
Figur D7-Skrueforbindelse mellom brakett 1 og motorholder 2.



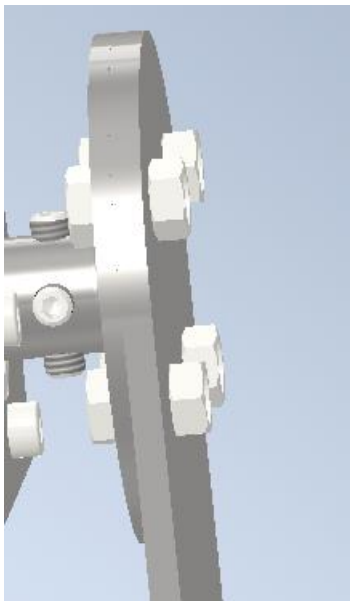
Figur D8-Hull på motorholder 2, og del utsatt for knekking.



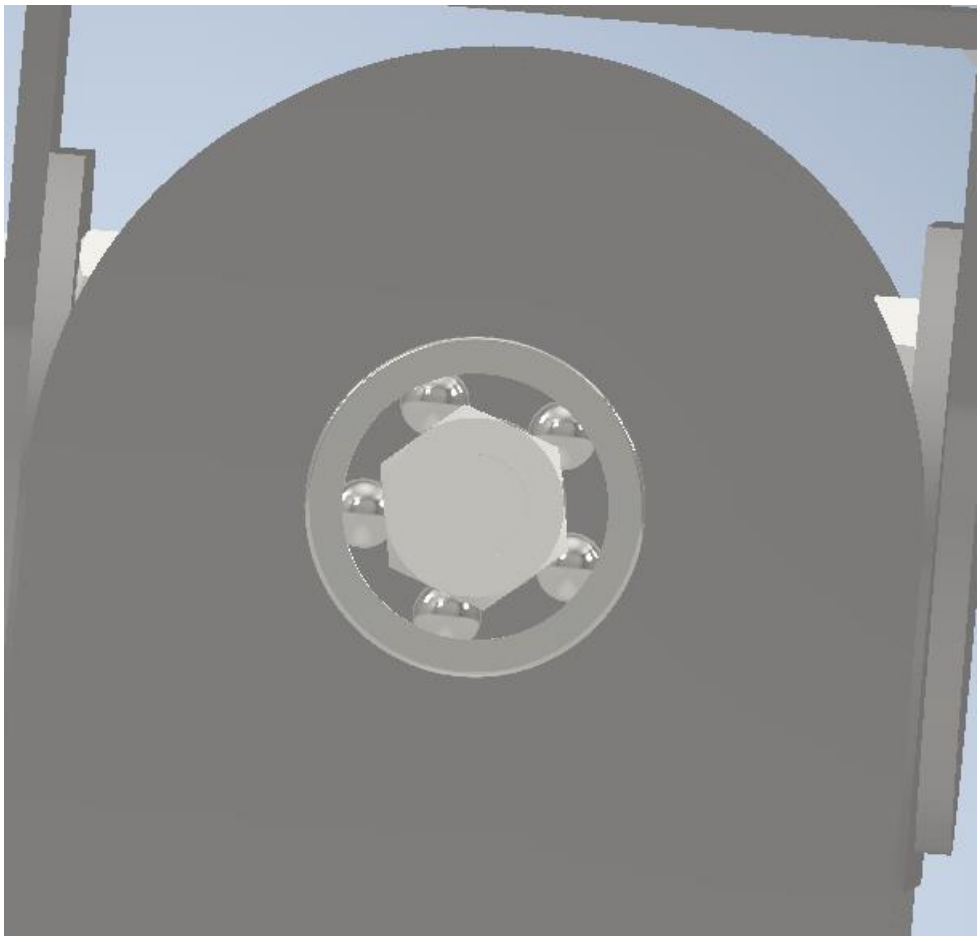
Figur D9-Skrueforbindelse mellom aksling på motor 2 og brakett 1, samt settskrueforbindelse for aksling til motor 2.



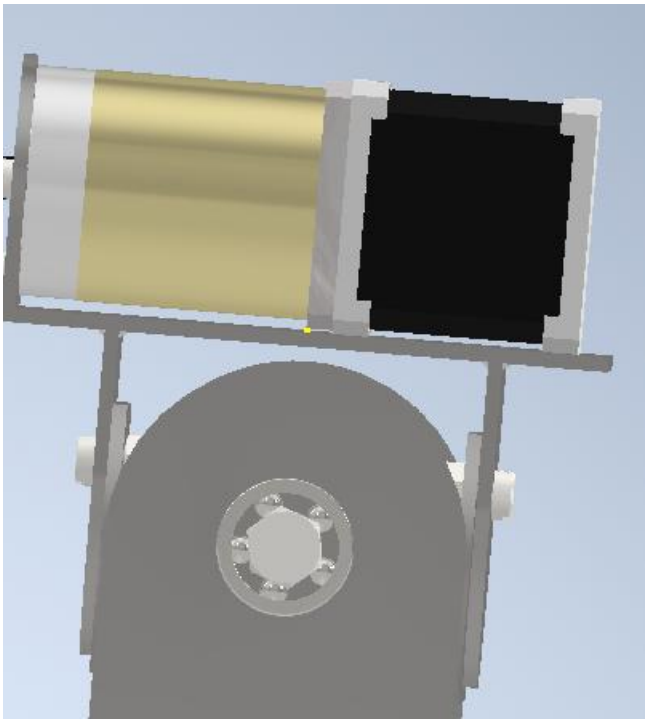
Figur D10-Skrueforbindelse mellom brakett 1 og brakett 2.



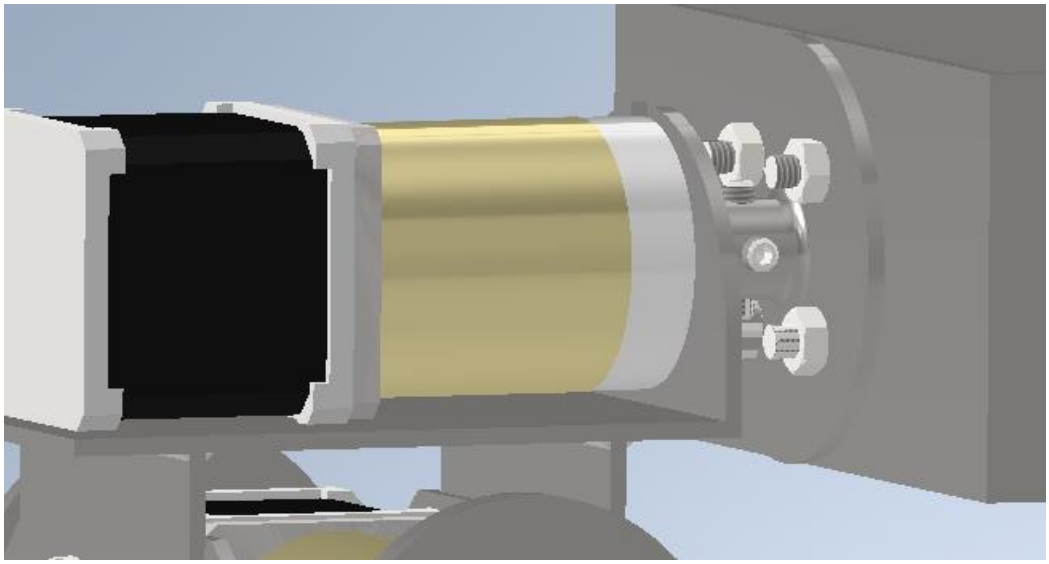
Figur D11-Skrueforbindelse mellom aksling på motor 3 og brakett 2, samt settskrueforbindelse på aksling til motor 3.



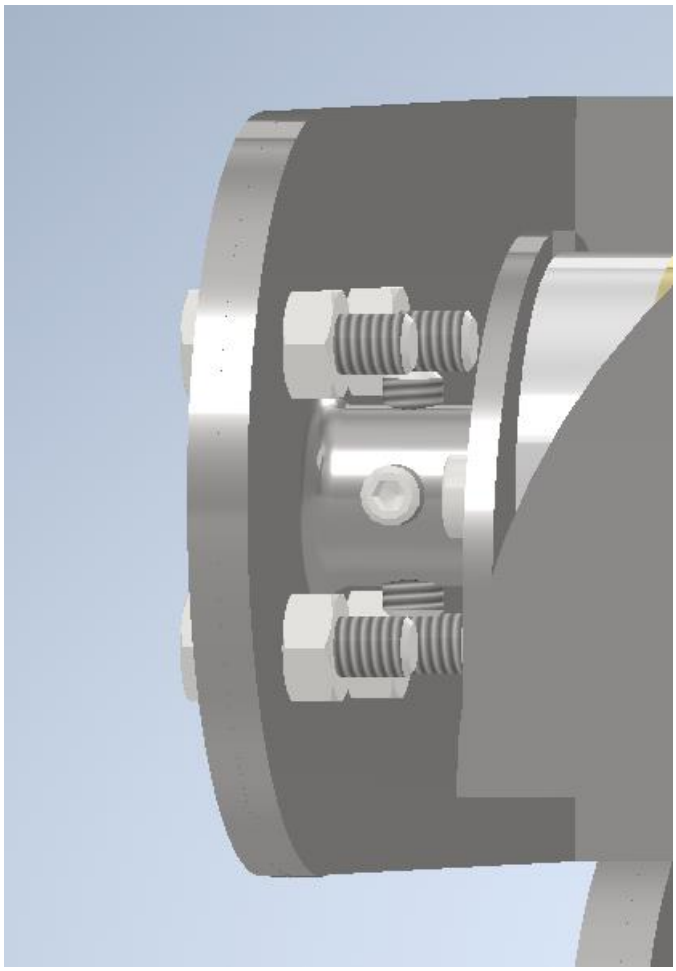
Figur D12-Krympeforbindelse kulelager på brakett 2, og skrueforbindelse mellom brakett m. kulelager-side og motorholder 3.



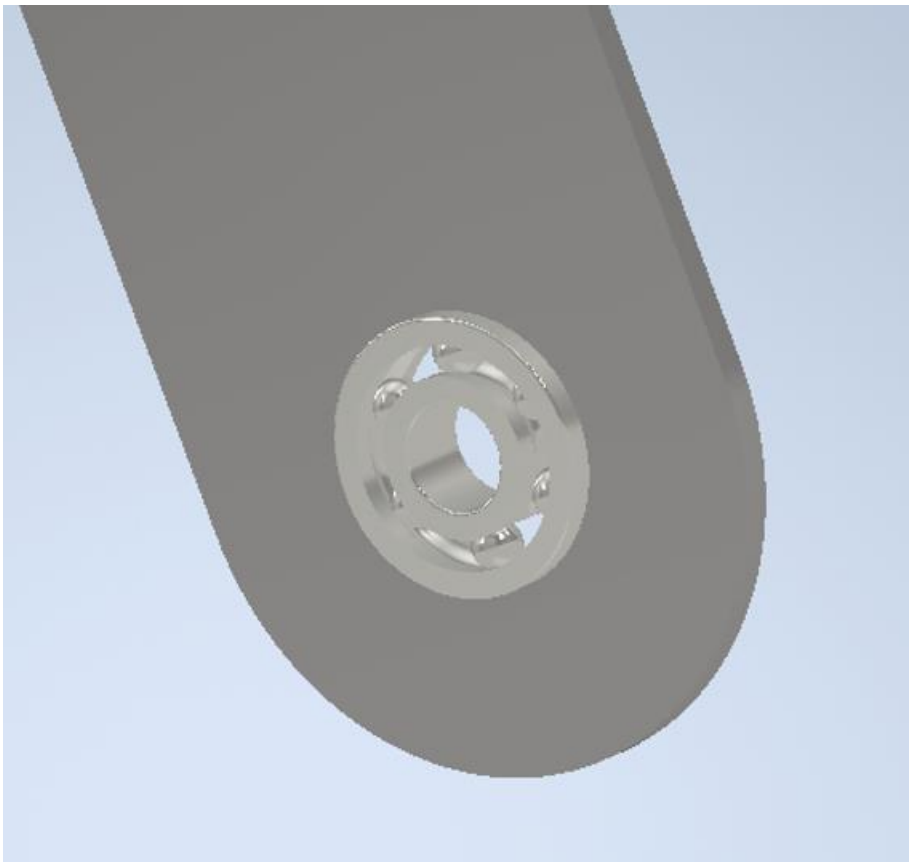
Figur D13-Skrueforbindelse mellom motorholder 3 og motorholder 4.



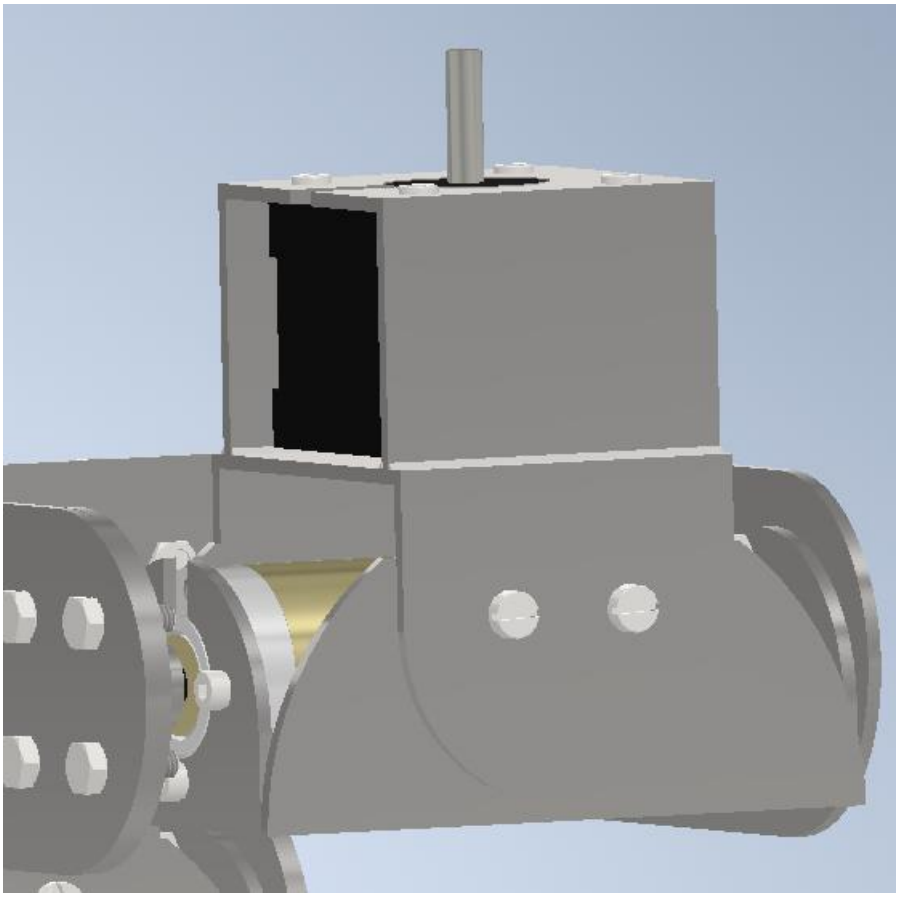
Figur D14-Skrueforbindelse mellom sammenkoblingsdel på aksling til motor 4 og u-brakett 3.



Figur D15-Skrueforbindelse mellom sammenkoblingsdel på aksling til motor 5 og u-brakett 3.



Figur D16-Krympeforbindelse mellom kulelager og u-brakett 3 (festet på motorholder 5).



Figur D17-Skrueforbindelse mellom motorholder 5 og motorholder 6.

Vedlegg E-Kode for fremover kinematikk

```
%%%Narve Gilje Nordås

close all
clc

%%Fremover kinematikk


%Denavit-Hartenberg-parametre: Fra 6 til 4 parametre.


%Definerte grad av frihet, lagde en symmetrisk tabell for parametrene(DH),
%identitetsmatrise(T) som ble brukt til å ganges med
%transformasjonsmatrisene, rammen for en transformasjonsmatrise(T_H),
%dimensjonerog leddvinklene(q1-->q6) i form av variabler(syms).

DOF=6;
DH= sym('DH', [DOF, 4]);
T=eye(4);
L1=0.3; %Lengden på forbindelsen mellom z-aksene til motor2 og motor3.
L2=0.15; %Lengden på forbindelsen mellom x-aksene til motor4 og motor5.
Lm2=(0.072+0.056); Lm3=(0.034+0.0436); Lm4=(0.034+0.0436); Lm5=(0.034+0.0436);
Lm6=0.042; %Motorlengder(Lm).
Hm2=0.057; Hm3=0.035; Hm4=0.035; Hm5=0.035; Hm6=0.042; %Motorhøyder(Hm).
Hv=0.03;Lv=0.052;Bv=0.03;                                    %Definerte total
høyde, lengde og bredde for ventil(Hv, Lv og Bv)
Hm=[Hm2;Hm3;Hm4;Hm5;Hm6;Hv]; %Samlet høyder i en vektor.
Bm2=Hm2; Bm3=Hm3; Bm4=Hm4; Bm5=Hm5; Bm6=Hm6; %Motorbredder(Bm), som i vårt
tilfelle er lik høyden.
Bm=[Bm2; Bm3; Bm4; Bm5; Bm6; Bv]; %Samlet bredder i en vektor.
Lm=[Lm2;Lm3;Lm4;Lm5;Lm6;Lv]; %Samler lengder i en vektor
Lr6=0.02; %Lengden på rotorer(Lr).
syms q1 q2 q3 q4 q5 q6 %Leddposisjoner(q1-->q6)

%Definerte DH-parametre[r, alfa, d, theta] for hvert ledd. Lagde deretter
%en for-loop for å sette parametrene inn i DH-tabellen, samt finne
%transformasjons(T_H1-->T_H6)- rotasjons- og translasjonsmatriser for hvert
%ledd(q1-->q6). Fant
%også transformasjonsmatriser(T_0_1-->T_0_6) for hvert ledd relatitvt til det
globale
%koordinatsystemet ved å gange dem med transformasjonsmatrisen til det
%forrige leddet. Endte til slutt opp med transformasjonsmatrisen fra
%globalt koordinat-system til tuppen av robotarmen(T_0_6).

parametre1=[0;pi/2;Hm2/2;q1]; %q1
parametre2=[L1;0;0;q2+pi/2]; %q2
parametre3=[(Hm3/2)+(Hm4/2);pi/2;0;q3]; %q3
parametre4=[0;-pi/2;L2;q4]; %q4
parametre5=[0;-pi/2;0;q5-pi/2]; %q5
```

```
parametre6=[0;0;Lm6+Lr6+(Hm5/2)+Lv;q6]; %q6
for i=1:DOF
    if i==1
        parametre=parametre1;
        R_0_0=T(1:3,1:3);
        d_0_0=T(1:3,4);
    elseif i==2
        parametre=parametre2;
        R_0_1=T(1:3,1:3);
        d_0_1=T(1:3,4);
    elseif i==3
        parametre=parametre3;
        R_0_2=T(1:3,1:3);
        d_0_2=T(1:3,4);
    elseif i==4
        parametre=parametre4;
        R_0_3=T(1:3,1:3);
        d_0_3=T(1:3,4);
    elseif i==5
        parametre=parametre5;
        R_0_4=T(1:3,1:3);
        d_0_4=T(1:3,4);
    else
        parametre=parametre6;
        R_0_5=T(1:3,1:3);
        d_0_5=T(1:3,4);
    end
    DH(i,1)=parametre(1,1);
    DH(i,2)=parametre(2,1);
    DH(i,3)=parametre(3,1);
    DH(i,4)=parametre(4,1);
    T_H=[cos(DH(i,4)) -sin(DH(i,4))*cos(DH(i,2)) sin(DH(i,4))*sin(DH(i,2))
DH(i,1)*cos(DH(i,4));
             sin(DH(i,4)) cos(DH(i,4))*cos(DH(i,2)) -cos(DH(i,4))*sin(DH(i,2))
DH(i,1)*sin(DH(i,4));
             0 sin(DH(i,2)) cos(DH(i,2)) DH(i,3);
             0 0 0 1];
    T=T*T_H;
end

%Det som var mest relevant for oss var posisjonen til tuppen i forhold til
%basen, rotasjonsmatrisen for tuppen i forhold basen, translasjon i x-, y-
%og z-retning relativt til, og rotasjonen om de ulike aksene i det globale
%koordinatsystemet.

R_0_6=T(1:3,1:3); %Rotasjonsmatrise fra base til tupp. Koordinatsystem 6 til
globale koordinater.
d_0_6=T(1:3,4);   %x-, y- og z-koordinater for tuppen.
```

Vedlegg F-Kode for simulasjonsverktøy

```
function varargout = Simulasjon_robot_arm(varargin)
% SIMULASJON_ROBOT_ARM MATLAB code for Simulasjon_robot_arm.fig
%      SIMULASJON_ROBOT_ARM, by itself, creates a new SIMULASJON_ROBOT_ARM or
raises the existing
%      singleton*.
%
%      H = SIMULASJON_ROBOT_ARM returns the handle to a new SIMULASJON_ROBOT_ARM
or the handle to
```

```
%      the existing singleton*.
%
%      SIMULASJON_ROBOT_ARM('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in SIMULASJON_ROBOT_ARM.M with the given input
arguments.
%
%      SIMULASJON_ROBOT_ARM('Property','Value',...) creates a new
SIMULASJON_ROBOT_ARM or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Simulasjon_robot_arm_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Simulasjon_robot_arm_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Simulasjon_robot_arm

% Last Modified by GUIDE v2.5 16-Feb-2022 22:56:35

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Simulasjon_robot_arm_OpeningFcn, ...
                   'gui_OutputFcn',  @Simulasjon_robot_arm_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Simulasjon_robot_arm is made visible.
function Simulasjon_robot_arm_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Simulasjon_robot_arm (see VARARGIN)

% Choose default command line output for Simulasjon_robot_arm
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Simulasjon_robot_arm wait for user response (see UIRESUME)
```

```
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Simulasjon_robot_arm_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function Theta_1_Callback(hObject, eventdata, handles)
% hObject    handle to Theta_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Theta_1 as text
%        str2double(get(hObject,'String')) returns contents of Theta_1 as a double


% --- Executes during object creation, after setting all properties.
function Theta_1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Theta_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Theta_2_Callback(hObject, eventdata, handles)
% hObject    handle to Theta_2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Theta_2 as text
%        str2double(get(hObject,'String')) returns contents of Theta_2 as a double


% --- Executes during object creation, after setting all properties.
function Theta_2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Theta_2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end



function Theta_3_Callback(hObject, eventdata, handles)
% hObject    handle to Theta_3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Theta_3 as text
%        str2double(get(hObject,'String')) returns contents of Theta_3 as a double


% --- Executes during object creation, after setting all properties.
function Theta_3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Theta_3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Theta_4_Callback(hObject, eventdata, handles)
% hObject    handle to Theta_4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Theta_4 as text
%        str2double(get(hObject,'String')) returns contents of Theta_4 as a double


% --- Executes during object creation, after setting all properties.
function Theta_4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Theta_4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Theta_5_Callback(hObject, eventdata, handles)
% hObject    handle to Theta_5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Theta_5 as text
%        str2double(get(hObject,'String')) returns contents of Theta_5 as a double
```

```
% --- Executes during object creation, after setting all properties.
function Theta_5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Theta_5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Theta_6_Callback(hObject, eventdata, handles)
% hObject    handle to Theta_6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Theta_6 as text
%        str2double(get(hObject,'String')) returns contents of Theta_6 as a double


% --- Executes during object creation, after setting all properties.
function Theta_6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Theta_6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function X_Callback(hObject, eventdata, handles)
% hObject    handle to X (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of X as text
%        str2double(get(hObject,'String')) returns contents of X as a double


% --- Executes during object creation, after setting all properties.
function X_CreateFcn(hObject, eventdata, handles)
% hObject    handle to X (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Y_Callback(hObject, eventdata, handles)
% hObject    handle to Y (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Y as text
%        str2double(get(hObject,'String')) returns contents of Y as a double


% --- Executes during object creation, after setting all properties.
function Y_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Y (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Z_Callback(hObject, eventdata, handles)
% hObject    handle to Z (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Z as text
%        str2double(get(hObject,'String')) returns contents of Z as a double


% --- Executes during object creation, after setting all properties.
function Z_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Z (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function RX_Callback(hObject, eventdata, handles)
% hObject    handle to RX (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of RX as text
%        str2double(get(hObject,'String')) returns contents of RX as a double


% --- Executes during object creation, after setting all properties.
function RX_CreateFcn(hObject, eventdata, handles)
% hObject    handle to RX (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function RY_Callback(hObject, eventdata, handles)
% hObject    handle to RY (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of RY as text
%        str2double(get(hObject,'String')) returns contents of RY as a double


% --- Executes during object creation, after setting all properties.
function RY_CreateFcn(hObject, eventdata, handles)
% hObject    handle to RY (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function RZ_Callback(hObject, eventdata, handles)
% hObject    handle to RZ (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of RZ as text
%        str2double(get(hObject,'String')) returns contents of RZ as a double


% --- Executes during object creation, after setting all properties.
function RZ_CreateFcn(hObject, eventdata, handles)
% hObject    handle to RZ (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in fremover_knapp.
function fremover_knapp_Callback(hObject, eventdata, handles)
% hObject    handle to fremover_knapp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Theta_1=deg2rad(str2double(handles.Theta_1.String));
Theta_2=deg2rad(str2double(handles.Theta_2.String))+pi/2;
Theta_3=deg2rad(str2double(handles.Theta_3.String));
Theta_4=deg2rad(str2double(handles.Theta_4.String));
Theta_5=deg2rad(str2double(handles.Theta_5.String))-pi/2;
Theta_6=deg2rad(str2double(handles.Theta_6.String));

L1=300; L2=150; Hm2=57; Hm3=35; Hm4=35; Hm5=35; Lm6=51; Lr6=20;Lv=52; %Lengder.

L(1)=Link([0;(Hm2/2)+4;0;pi/2]); %q1
L(2)=Link([0;0;L1;0]); %q2
L(3)=Link([0;0;(Hm3/2)+(Hm4/2)+12;pi/2]); %q3
L(4)=Link([0;L2;0;-pi/2]); %q4
L(5)=Link([0;0;0;-pi/2]); %q5
L(6)=[0;Lm6+Lr6+(Hm5/2)+Lv+10;0;0]; %q6

robotarm=SerialLink(L);
robotarm.name='Robotarm 6 grader av frihet';
robotarm.plot([Theta_1 Theta_2 Theta_3 Theta_4 Theta_5 Theta_6]);

T=robotarm.fkine([Theta_1 Theta_2 Theta_3 Theta_4 Theta_5 Theta_6]);
Posisjon=transl(T);
Rotasjon=t2r(T);
handles.X.String=num2str(round(Posisjon(1),1));
handles.Y.String=num2str(round(Posisjon(2),1));
handles.Z.String=num2str(round(Posisjon(3),1));
handles.RX.String=num2str(rad2deg(acos(round(Rotasjon(1,1),2))));
handles.RY.String=num2str(rad2deg(acos(round(Rotasjon(2,2),2))));
handles.RZ.String=num2str(rad2deg(acos(round(Rotasjon(3,3),2))));
% --- Executes on button press in invers_knapp.
function invers_knapp_Callback(hObject, eventdata, handles)
% hObject    handle to invers_knapp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
x=str2double(handles.X.String);
y=str2double(handles.Y.String);
z=str2double(handles.Z.String);
rx=deg2rad(str2double(handles.RX.String));
ry=deg2rad(str2double(handles.RY.String));
rz=deg2rad(str2double(handles.RZ.String));

L1=300; L2=150; Hm2=57; Hm3=35; Hm4=35; Hm5=35; Lm6=51; Lr6=20;Lv=52; %Lengder.

L(1)=Link([0;(Hm2/2)+4;0;pi/2]); %q1
L(2)=Link([0;0;L1;0]); %q2
L(3)=Link([0;0;(Hm3/2)+(Hm4/2)+12;pi/2]); %q3
```

```
L(4)=Link([0;L2;0;-pi/2]); %q4
L(5)=Link([0;0;0;-pi/2]); %q5
L(6)=[0;Lm6+Lr6+(Hm5/2)+Lv+10;0;0]; %q6

robotarm=SerialLink(L); %Definerte robotarmen med DH-parametre
robotarm.name='Robotarm 6 grader av frihet';

P=[x;y;z]; Phl=[0;0;0]; %Punkt tupp(P), og punkt håndledd(Phl)
RX=[1 0 0; 0 cos(rx) -sin(rx); 0 sin(rx) cos(rx)];
RY=[cos(ry) 0 sin(ry); 0 1 0; -sin(ry) 0 cos(ry)];
RZ=[cos(rz) -sin(rz) 0; sin(rz) cos(rz) 0; 0 0 1]; %rotasjonsmatriser
R_0_6=RX*RY*RZ;
k=[0; 0; 1];
k=R_0_6*k; k=((Hm5/2)+Lm6+Lr6+Lv)*k;%Finner z-komponentet til rotasjonsmatrisen
for i=1:3
    Phl(i,1)=P(i,1)-k(i,1); %Definerer punktet til håndleddet i forhold til tuppen
end
DOF=6; %Grader av frihet
theta=sym('THETA', [DOF,1]); %Kolonnevektor med leddvinklene
xhl=Phl(1,1);
yhl=Phl(2,1);
zhl=Phl(3,1); %Koordinater til håndleddet

theta1=round(atan2(yhl,xhl),2);
handles.Theta_1.String=num2str(rad2deg(theta1)); %theta1

albue_opp=true; %Definerte om albue/ledd  roterer med eller mot klokken.
a=sqrt(xhl^2+yhl^2); %a
b=zhl-(Hm2/2); %b
R1=sqrt(a^2+b^2);
R2=sqrt(L2^2+((Hm3/2)+(Hm4/2))^2); %R1 OG R2

theta2=round(-((pi/2)-atan2(b,a)-acos((R1^2+L1^2-R2^2)/(2*R1*L1))),2);
handles.Theta_2.String=num2str(round(rad2deg(theta2),2)); %theta 2
theta3=round(-(pi-acos((R2^2+L1^2-R1^2)/(2*L1*R2))-
atan2(L2,((Hm3/2)+(Hm4/2)))),2);
handles.Theta_3.String=num2str(round(rad2deg(theta3),2)); %theta 3

if isreal(theta2)==0
    disp('Posisjonen kan ikke nås med gjeldende konfigurasjon(Skulder-fram-albue-
opp+Skulder-fram-albue-ned')
end

if theta3>0
   theta3=round((-pi/2+acos((R2^2+L1^2-R1^2)/(2*L1*R2))-
atan2(((Hm3/2)+(Hm4/2)),L2)),2)
   handles.Theta_3.String=num2str(rad2deg(theta3));%theta 3
end
DH= sym('DH',[DOF 4]); %DH-tabell
T=eye(4); %Transformasjonmatrisen
T_H=zeros(4,4); %Denavit-Hartenberg-transformasjonmatrisen
parametre1=[0;pi/2;Hm2/2;theta1]; %q1
parametre2=[L1;0;0;theta2+pi/2]; %q2
parametre3=[(Hm3/2)+(Hm4/2);pi/2;0;theta3]; %q3
parametre4=[0;-pi/2;L2;theta(4,1)]; %q4
parametre5=[0;-pi/2;0;theta(5,1)-pi/2]; %q5
parametre6=[0;0;Lm6+Lr6+(Hm5/2)+Lv;theta(6,1)]; %q6

for i=1:(DOF/2) %For-loop for å finne rotasjon fra 0 til 3.
```

```
    if i==1
        parametre=parametre1;
    elseif i==2
        parametre=parametre2;
    else
        parametre=parametre3;
    end
    DH(i,1)=parametre(1,1);
    DH(i,2)=parametre(2,1);
    DH(i,3)=parametre(3,1);
    DH(i,4)=parametre(4,1);
    T_H=[cos(DH(i,4)) -sin(DH(i,4))*cos(DH(i,2)) sin(DH(i,4))*sin(DH(i,2))
DH(i,1)*cos(DH(i,4));
             sin(DH(i,4)) cos(DH(i,4))*cos(DH(i,2)) -cos(DH(i,4))*sin(DH(i,2))
DH(i,1)*sin(DH(i,4));
             0 sin(DH(i,2)) cos(DH(i,2)) DH(i,3);
             0 0 0 1];
    T=round(T*T_H,2);
end

R_0_3=T(1:3,1:3); %Rotasjon fra 0 til 3.

T=eye(4);
T_H=zeros(4,4);
for i=1:DOF/2 %For-loop for å finne rotasjonsmatrisen fra 3 til 6.
    if i==1
        parametre=parametre4;
    elseif i==2
        parametre=parametre5;
    else
        parametre=parametre6;
    end
    DH(i,1)=parametre(1,1);
    DH(i,2)=parametre(2,1);
    DH(i,3)=parametre(3,1);
    DH(i,4)=parametre(4,1);
    T_H=[cos(DH(i,4)) -sin(DH(i,4))*cos(DH(i,2)) sin(DH(i,4))*sin(DH(i,2))
DH(i,1)*cos(DH(i,4));
             sin(DH(i,4)) cos(DH(i,4))*cos(DH(i,2)) -cos(DH(i,4))*sin(DH(i,2))
DH(i,1)*sin(DH(i,4));
             0 sin(DH(i,2)) cos(DH(i,2)) DH(i,3);
             0 0 0 1];
    T=T*T_H;
end
R_3_6=T(1:3,1:3);
R_3_6_tall=transpose(R_0_3)*R_0_6; %Fant rotasjonsmatrisen med tallverdier. Invers
av rotasjon fra 0 til 3, ganget med total rotasjon.
R_0_6
R_0_3
transpose(R_0_3)
theta5=double(-pi/2+acos(R_3_6_tall(3,3)));
handles.Theta_5.String=num2str(rad2deg(theta5)); %theta5. Kan ikke være 0 eller
pi.
if theta5 ~=0 | theta5 ~= pi
    theta6=double(-atan2(R_3_6_tall(3,2),R_3_6_tall(3,1)));
handles.Theta_6.String=num2str(round(rad2deg(theta6),2)); %theta6
    theta4=double(atan2(R_3_6_tall(2,3),R_3_6_tall(1,3)));
handles.Theta_4.String=num2str(round(rad2deg(theta4),2)); %theta4
else
```

```
    disp('Vinkel 5 kan ikke være 0 eller pi! Prøve med ny input')
end
robotarm.plot([theta1 theta2+pi/2 theta3 theta4 theta5-pi/2 theta6]);
```

Vedlegg G-Kode for beregning av vrimoment med Lagrange-mekanikk

```
function [T_alle_posisjoner] = vrimoment_beregning()
%%%Kinematikk/dynamikk-program for å regne ut nødvendig tilført vrimoment
%%%Narve Gilje Nordås

close all
clc

%%Først ble fremover kinematikk definert


%Denavit-Hartenberg-parametre: Fra 6 til 4 parametre.


%Definerte grad av frihet, lagde en symmetrisk tabell for parametrene(DH),
%identitetsmatrise(T) som ble brukt til å ganges med
%transformasjonsmatrisene, rammen for en transformasjonsmatrise(T_H),
%dimensjonerog leddvinklene(q1-->q6) i form av variabler(syms).

DOF=6;
DH= sym('DH', [DOF, 4]);
T=eye(4);
L1=0.3; %Lengden på forbindelsen mellom z-aksene til motor2 og motor3.
L2=0.15; %Lengden på forbindelsen mellom x-aksene til motor4 og motor5.
Lm2=(0.072+0.056); Lm3=(0.034+0.0436); Lm4=(0.034+0.0436); Lm5=(0.034+0.0436); Lm6=0.042;
%Motorlengder(Lm).
Hm2=0.057; Hm3=0.035; Hm4=0.035; Hm5=0.035; Hm6=0.042; %Motorhøyder(Hm).
Hv=0.03;Lv=0.052;Bv=0.03;                                    %Definerer total høyde, lengde og bredde
for ventil(Hv, Lv og Bv)
Hm=[Hm2;Hm3;Hm4;Hm5;Hm6;Hv]; %Samler høyder i en vektor.
Bm2=Hm2; Bm3=Hm3; Bm4=Hm4; Bm5=Hm5; Bm6=Hm6; %Motorbredder(Bm), som i vårt tilfelle er lik høyden.
Bm=[Bm2; Bm3; Bm4; Bm5; Bm6; Bv]; %Samler bredder i en vektor.
Lm=[Lm2;Lm3;Lm4;Lm5;Lm6;Lv]; %Samler lengder i en vektor
Lr6=0.02; %Lengden på rotorer(Lr).
syms q1 q2 q3 q4 q5 q6 %Leddposisjoner(q1-->q6)

%Definerte DH-parametre[r, alfa, d, theta] for hvert ledd. Lagde deretter
%en for-loop for å sette parametrene inn i DH-tabellen, samt finne
%transformasjons(T_H1-->T_H6)- rotasjons- og translasjonsmatriser for hvert
%ledd(q1-->q6). Fant
%også transformasjonsmatriser(T_0_1-->T_0_6) for hvert ledd relativt til det globale
%koordinatsystemet ved å gange dem med transformasjonsmatrisen til det
%forrige leddet. Endte til slutt opp med transformasjonsmatrisen fra
%globalt koordinat-system til tuppen av robotarmen(T_0_6).

parametre1=[0;pi/2;Hm2/2;q1]; %q1
parametre2=[L1;0;0;q2+pi/2]; %q2
parametre3=[(Hm3/2)+(Hm4/2);pi/2;0;q3]; %q3
parametre4=[0;-pi/2;L2;q4]; %q4
parametre5=[0;-pi/2;0;q5-pi/2]; %q5
parametre6=[0;0;Lm6+Lr6+(Hm5/2)+Lv;q6]; %q6
for i=1:DOF
    if i==1
        parametre=parametre1;
        R_0_0=T(1:3,1:3);
        d_0_0=T(1:3,4);
    elseif i==2
        parametre=parametre2;
        R_0_1=T(1:3,1:3);
        d_0_1=T(1:3,4);
    elseif i==3
        parametre=parametre3;
        R_0_2=T(1:3,1:3);
        d_0_2=T(1:3,4);
    elseif i==4
```

```
        parametre=parametre4;
        R_0_3=T(1:3,1:3);
        d_0_3=T(1:3,4);
    elseif i==5
        parametre=parametre5;
        R_0_4=T(1:3,1:3);
        d_0_4=T(1:3,4);
    else
        parametre=parametre6;
        R_0_5=T(1:3,1:3);
        d_0_5=T(1:3,4);
    end
    DH(i,1)=parametre(1,1);
    DH(i,2)=parametre(2,1);
    DH(i,3)=parametre(3,1);
    DH(i,4)=parametre(4,1);
    T_H=[cos(DH(i,4)) -sin(DH(i,4))*cos(DH(i,2)) sin(DH(i,4))*sin(DH(i,2)) DH(i,1)*cos(DH(i,4));
             sin(DH(i,4)) cos(DH(i,4))*cos(DH(i,2)) -cos(DH(i,4))*sin(DH(i,2)) DH(i,1)*sin(DH(i,4));
             0 sin(DH(i,2)) cos(DH(i,2)) DH(i,3);
             0 0 0 1];
    T=T*T_H;
end

%Det som var mest relevant for oss er posisjonen til tuppen i forhold til
%basen, rotasjonsmatrisen for tuppen i forhold basen, translasjon i x-, y-
%og z-retning relativt til, og rotasjonen om de ulike aksene i det globale
%koordinatsystemet.

R_0_6=T(1:3,1:3); %Tilsvarende rotasjonsmatrise.
d_0_6=T(1:3,4);   %x-, y- og z-koordinater for tuppen.

%Definerte Jacobmatrise(J), samt lineære og
%rotasjons-komponenter(JV1-->JV6, og JW1-->JW6)
% Ganget
%rotasjonsmatriser for hvert ledd med k-vektoren for å finne
%rotasjonskomponentene. Gjorde det samme for å finne de lineære komponentene,
%men da måtte man også ta kryssproduktet med denne vektoren og
%translasjonsvektoren for hvert ledd. Subtraherte translasjonsvektoren for
%hvert ledd fra translasjonsvektoren ved tuppen for å finne posisjonen
%relativt til tuppen(d_0_6-d_0_0-->d_0_6-d_0_5). Kryssproduktet står
% følgelig vinkelrett på
%rotasjonskomponenten, som da ga oss den lineære hastigheten. Begge er
%3x1-vektorer. Lagde deretter en for-loop for å putte komponentene inn i
%den fullverdige Jacobimatrisen. Lagde i tillegg to 3x6-matriser. En
%lineær(JV)
%og en for rotasjon(JW). Det fortalte oss hvordan hastigheten til et ledd blir
%påvirket av hastigheten til de foregående leddene, og sin egen hastighet.
%Ved ledd 1(basen) er f.eks bare første kolonne noe annet enn 0, da den
%selvsagt ikke blir påvirket av ledd 2-6. Lager også transponerte versjoner
%for å senere kunne bruke dette i dynamikk-beregninger(JV1T-->JV6T, og JW1T-->JW6T).

J=sym('J' , [DOF, DOF]); %Definerte først som "symbolic" for å kunne brukes
for i=1:DOF              %i beregninger. Fyller deretter på med 0.
    for j=1:DOF
        J(i,j)=0;
    end
end

k=[0;0;1]; %k-vektor

JW1=(R_0_0*k); %Komponenter for rotasjonshastighet(JW1-->JW6)
JW2=(R_0_1*k); %Fikk z-komponenten av hver rotasjonsmatrise.
JW3=(R_0_2*k);
JW4=(R_0_3*k);
JW5=(R_0_4*k);
JW6=(R_0_5*k);
JV1=cross((JW1),(d_0_6-d_0_0)); %Komponenter for lineær hastighet(JV1-->JV6)
JV2=cross((JW2),(d_0_6-d_0_1));
JV3=cross((JW3),(d_0_6-d_0_2));
JV4=cross((JW4),(d_0_6-d_0_3));
JV5=cross((JW5),(d_0_6-d_0_4));
JV6=cross((JW6),(d_0_6-d_0_5));

for i=1:DOF %Lagde 6x6-matriser, en for hvert ledd(JV1-->JV6, og JW1-->JW6
```

```
    if i==1 %Fylte opp Jacobimatrise med verdier fra hver komponent(J)
        JV=JV1;
        JW=JW1;
    elseif i==2
        JV1=J(1:3,:);
        JW1=J(4:6,:);
        JV=JV2;
        JW=JW2;
    elseif i==3
        JV2=J(1:3,:);
        JW2=J(4:6,:);
        JV=JV3;
        JW=JW3;
    elseif i==4
        JV3=J(1:3,:);
        JW3=J(4:6,:);
        JV=JV4;
        JW=JW4;
    elseif i==5
        JV4=J(1:3,:);
        JW4=J(4:6,:);
        JV=JV5;
        JW=JW5;
    else
        JV5=J(1:3,:);
        JW5=J(4:6,:);
        JV=JV6;
        JW=JW6;
    end
    J(1,i)=JV(1,1);
    J(2,i)=JV(2,1);
    J(3,i)=JV(3,1);
    J(4,i)=JW(1,1);
    J(5,i)=JW(2,1);
    J(6,i)=JW(3,1);
end

JV6=J(1:3,:);
JW6=J(4:6,:);
JT=transpose(J);

JV1T=transpose(JV1); JW1T=transpose(JW1);  %Transponerte matriser av den generelle Jacobimatrisen(JT),
og
JV2T=transpose(JV2); JW2T=transpose(JW2);  %for hver av de 12 matrisene(JV1T-->JV6T, og JW1T-->JW6T)
JV3T=transpose(JV3); JW3T=transpose(JW3);
JV4T=transpose(JV4); JW4T=transpose(JW4);
JV5T=transpose(JV5); JW5T=transpose(JW5);
JV6T=transpose(JV6); JW6T=transpose(JW6);

m1=1.7; m2=0.36; m3=0.36; m4=0.36; m5=0.23; m6=(0.089); %Definerte masser for motorene(m1 er for motor
2, m2 for motor 3 osv..).
m=[m1; m2; m3; m4; m5; m6]; %Samlet massene i en kolonnevektor.

I1=[(1/12*m(1,1))*((Hm(1,1)^2)+(Lm(1,1)^2)) 0 0; 0 (1/12*m(1,1))*((Bm(1,1)^2)+(Lm(1,1)^2)) 0; 0 0
(1/12*m(1,1))*((Hm(1,1)^2)+(Bm(1,1)^2))];
I2=[(1/12*m(2,1))*((Lm(2,1)^2)+(Hm(2,1)^2)) 0 0; 0 (1/12*m(2,1))*((Bm(2,1)^2)+(Hm(2,1)^2)) 0; 0 0
(1/12*m(2,1))*((Lm(2,1)^2)+(Bm(2,1)^2))];
I3=[(1/12*m(3,1))*((Lm(3,1)^2)+(Hm(3,1)^2)) 0 0; 0 (1/12*m(3,1))*((Bm(3,1)^2)+(Hm(3,1)^2)) 0; 0 0
(1/12*m(3,1))*((Lm(3,1)^2)+(Bm(3,1)^2))];
I4=[(1/12*m(4,1))*((Lm(4,1)^2)+(Hm(4,1)^2)) 0 0; 0 (1/12*m(4,1))*((Bm(4,1)^2)+(Hm(4,1)^2)) 0; 0 0
(1/12*m(4,1))*((Lm(4,1)^2)+(Bm(4,1)^2))];
I5=[(1/12*m(5,1))*((Lm(5,1)^2)+(Hm(5,1)^2)) 0 0; 0 (1/12*m(5,1))*((Bm(5,1)^2)+(Hm(5,1)^2)) 0; 0 0
(1/12*m(5,1))*((Lm(5,1)^2)+(Bm(5,1)^2))];
I6=[(1/12*m(6,1))*((Hm(6,1)^2)+(Lm(6,1)^2)) 0 0; 0 (1/12*m(6,1))*((Bm(6,1)^2)+(Lm(6,1)^2)) 0; 0 0
(1/12*m(6,1))*((Hm(6,1)^2)+(Bm(6,1)^2))];%Beregner treghetsmomentet om de 3 aksene ved massesenteret


q_dot=sym('q_dot', [DOF, 1]); %Definerte leddhastighetene i en kolonnevektor(q_dot), først som
variabler

D1=((m1*JV1T*JV1)+(JW1T*R_0_1*I1*transpose(R_0_1)*JW1)); %Beregnet treghetsmatriser(D1--D6)
D2=((m2*JV2T*JV2)+(JW2T*R_0_2*I2*transpose(R_0_2)*JW2)); %for hvert av leddene.
D3=((m3*JV3T*JV3)+(JW3T*R_0_3*I3*transpose(R_0_3)*JW3));
D4=((m4*JV4T*JV4)+(JW4T*R_0_4*I4*transpose(R_0_4)*JW4));
```

```
D5=((m5*JV5T*JV5)+(JW5T*R_0_5*I5*transpose(R_0_5)*JW5));
D6=((m6*JV6T*JV6)+(JW6T*R_0_6*I6*transpose(R_0_6)*JW6));
D=D1+D2+D3+D4+D5+D6; %Summen av treghetsmatrisene i en matrise(D)

zsenter=sym('z', [DOF, 1]);                              %Definerte z-koordinaten til massesentrene.
zsenter1=(Hm2/2); zsenter(1,1)=zsenter1;
zsenter2=zsenter1+(cos(q2)*L1); zsenter(2,1)=zsenter2;
zsenter3=zsenter2+(cos(q2+q3)*((Hm3)/2+(Hm4/2))); zsenter(3,1)=zsenter3;
zsenter4=zsenter3+(sin(q2+q3)*L2); zsenter(4,1)=zsenter4;
zsenter5=zsenter4+(cos(q2+q3+q5)*((Hm5/2)+(Lm6/2))); zsenter(5,1)=zsenter5;
z_retning_6=((R_0_5*k)*((Lm6/2)+Lr6+(Lv/2)));
zsenter6=zsenter5+z_retning_6(3,1); zsenter(6,1)=zsenter6;
g=9.81;                         %Tyngdekraftakselerasjonen(g)
P=0;
for i=1:DOF                     %Beregnet den potensielle energien for hvert ledd(mgh),
    P=P+m(i,1)*g*zsenter(i,1); %og fant summen av disse ved hjelp av en for-loop.
end

c1=sym('c1', [DOF, DOF]);      %Definerte 6stk. 6x6-matriser for å beregne Christoffel-symbolene(c1-->c6)
c2=sym('c2', [DOF, DOF]);      %Alle elementer i c1 har k som 1, c2 impliserer at k er lik 2 osv.
c3=sym('c3', [DOF, DOF]);
c4=sym('c4', [DOF, DOF]);
c5=sym('c5', [DOF, DOF]);
c6=sym('c6', [DOF, DOF]);
c=sym('c', [DOF, DOF]);  %6x6 C-matrise som består av 216 Christoffel-symboler.
for i=1:6
    for j=1:6
        c1(i,j)=0;c2(i,j)=0;c3(i,j)=0;c4(i,j)=0;c5(i,j)=0;c6(i,j)=0;c(i,j)=0;
    end
end
q=[q1;q2;q3;q4;q5;q6];        %Satt leddposisjonene inn i kolonnevektor for å enklere kunne gjøre beregninger.

for k=1:DOF
%Nøstet for-lopp+if-statement for å beregne de 216 Christoffel-symbolene, og plassere dem inn i C-matrisen
    for i=1:DOF
%Fulgte den skriftlige formelen, og prøvde å gjøre koden så kort og konsis som mulig.
        for j=1:DOF
            if k==1
                c1(i, j)=(1/2)*(diff(D(j, k),q(i,1))+diff(D(i, k),q(j,1))-(diff(D(i, j), q(k, 1))))*q_dot(j,1); %Delvis deriverte med hensyn til de ulike leddvinklene
                c(k,i)=c(k,i)+c1(i,j);
%Plasserte 6 Christoffelsymboler(i fra 1 til 6) inn som et element i C-matrisen.
            elseif k==2
%Gjentok 6 ganger(j fra 1 til 6)
                c2(i, j)=(1/2)*(diff(D(j, k),q(i,1))+diff(D(i, k),q(j,1))-(diff(D(i, j), q(k, 1))))*q_dot(j,1); %Det ga 36 Christoffelsymboler som til sammen ble 6 elementer i C-matrisen(rad for rad).
                 c(k,i)=c(k,i)+c2(i,j);
%Gjentok 6 ganger(k fra 1 til 6)
            elseif k==3
%%Det ga 216 Christoffelsymboler som til sammen blir 36 elementer i C-matrisen(hele matrisen).
                c3(i, j)=(1/2)*(diff(D(j, k),q(i,1))+diff(D(i, k),q(j,1))-(diff(D(i, j), q(k, 1))))*q_dot(j,1);
                c(k,i)=c(k,i)+c3(i,j);
            elseif k==4
                c4(i, j)=(1/2)*(diff(D(j, k),q(i,1))+diff(D(i, k),q(j,1))-(diff(D(i, j), q(k, 1))))*q_dot(j,1);
                c(k,i)=c(k,i)+c4(i,j);
            elseif k==5
                c5(i, j)=(1/2)*(diff(D(j, k),q(i,1))+diff(D(i, k),q(j,1))-(diff(D(i, j), q(k, 1))))*q_dot(j,1);
                c(k,i)=c(k,i)+c5(i,j);
            else
                c6(i, j)=(1/2)*(diff(D(j, k),q(i,1))+diff(D(i, k),q(j,1))-(diff(D(i, j), q(k, 1))))*q_dot(j,1);
                c(k,i)=c(k,i)+c6(i,j);
            end
        end
    end
end
```

```
g=sym('g', [DOF, 1]);             %g-kolonnevektor
for i=1:DOF                       %Plasserte de delvis deriverte av potensiell energi
    g(i,1)=diff(P, q(i));         %med hensyn til leddvinklene inn i g-matrisen.
end

q_dobbel_dot_max_mod=[pi/4;pi/8;pi/8;pi/2;pi;8*pi]; %Input-verdier av de maksimale
leddakselerasjonene(q_dobbel_dot). De modifiserte verdiene.
q_dot_max_mod=[pi/2;pi/4;pi/4;pi;pi;4*pi]; %Input-verdier av maksimale leddhastighetene(q_dot). De
modifiserte verdiene.
%q_dot_max_mod=zeros(6,1); q_dobbel_dot_max_mod=zeros(6,1);
T_inn=D*q_dobbel_dot_max_mod+c*q_dot+g; %Regnet etter formel ut tilført vrimoment for hvert
ledd(T_inn(1,1)-->T_inn(6,1))
T_inn=subs(T_inn,q_dot,q_dot_max_mod); %Substituerte variabler med verdier fra inputen.
IR1=3*(10^-5); IR2=3*(10^-5); IR3=3*(10^-5); IR4=3*(10^-5); IR5=3*(10^-5); IR6=1.8*(10^-6);
%Treghetsmoment for rotasjon av rotor(IR)
IR=[IR1;IR2;IR3;IR4;IR5;IR6];
TR=zeros(6,1);
for i=1:DOF                        %Beregnet vrimoment for rotasjon av rotor for hver av de 6
motorene(TR(1,1)-->TR(6,1))
    TR(i, 1)=IR(i,1)*q_dobbel_dot_max_mod(i,1);
end
Tm=T_inn+TR;                                %Vrimoment som følge av rotasjon rotorer og for å bevege
selve robotarmen(Tm), som er summen av rotordreiemomentet(TR), og det generelle dreimomentet(T_inn).

m_lokk=2.05; l_lokk=0.22; g=9.81;
syms alfa F_tilfoert
F_ytre=round(solve(((m_lokk*g)*(l_lokk/2)*cos(alfa))==F_tilfoert*l_lokk, F_tilfoert),2); %Størrelse på
kraften som virker fra lokket på vaffeljernet på robotarmen ved tuppen(Fz).
Fx=sin(alfa)*F_ytre;
Fz=cos(alfa)*F_ytre;
T_alle_posisjoner=zeros(6,5); %Matrise for å samle dreiemomenter ved alle 5 ulike posisjoner.
for i=1:5
    if i==1
        q_inn=deg2rad([-118.6; -20.63; -31.51; 179.52; -51.80; 61.39]); %Input-verdier for posisjonene
til hvert ledd(q_inn). Ved hvileposisjon.
        Ft=[0; 0; 0; 0; 0; 0];    %Ytre belastning som kolonnevektor(krefter i x-, y- og z-retning,
samt rotasjon om disse aksene(Ft).
    elseif i==2
        q_inn=deg2rad([-107.72; -45.26; 21.77; 179.72; -23.27; 72.47]); %Input-verdier for posisjonene
til hvert ledd(q_inn). Ved senter til nærmeste vaffel.
        Ft=[0; 0; 0; 0; 0; 0];
    elseif i==3
        q_inn=deg2rad([-95.11; -26.36; -14-90; 86.64; -3.83; 131.20]); %Input-verdier for posisjonene
til hvert ledd(q_inn). Ved senter til vaffel lengst borte.
        Ft=[0; 0; 0; 0; 0; 0];
    elseif i==4
        q_inn=deg2rad([-68.75;-38.39;-46.98;110.62;1.72;175.96]); %Input-verdier for posisjonene til
hvert ledd(q_inn). Ved løfte lokk, topp. Ingen ytre belastning.
        alfa_inn=pi/4;
        subs(Fx,alfa,alfa_inn);
        subs(Fz,alfa,alfa_inn);
        Ft=[Fx; 0; Fz; 0; 0; 0];
    else
        q_inn=deg2rad([-53.86; -65.89; -15.47; 125.80; 5.06; 173.00]); %Input-verdier for posisjonene
til hvert ledd(q_inn). Ved løfte lokk, bunn. Med ytre belastning.
        alfa_inn=0;
        subs(Fz,alfa,alfa_inn);
        Ft=[0; 0; Fz; 0; 0; 0];
    end
    JT=subs(JT,q,q_inn); %Satte tallverdier inn i transponert Jacobimatrise(JT).
    T_Ft=round(subs(JT*Ft,q,q_inn),2); %Nødvendig vrimoment som følge av ytre belastning(T_Ft)
    T_alle_posisjoner(1,i)=round(subs(Tm(1,1),q, q_inn),2)+T_Ft(1,1);
    %Vrimoment motor 1 til motor 6(T1-->T6) ved hvileposisjon. Maks. aks. og hastigheter alle ledd.
Ingen ytre belastning.
    T_alle_posisjoner(2,i)=round(subs(Tm(2,1),q, q_inn),2)+T_Ft(2,1);
    T_alle_posisjoner(3,i)=round(subs(Tm(3,1),q, q_inn),2)+T_Ft(3,1);
    T_alle_posisjoner(4,i)=round(subs(Tm(4,1),q, q_inn),2)+T_Ft(4,1);
    T_alle_posisjoner(5,i)=round(subs(Tm(5,1),q, q_inn),2)+T_Ft(5,1);
    T_alle_posisjoner(6,i)=round(subs(Tm(6,1),q,q_inn),2)+T_Ft(6,1);
End
```

Vedlegg H-Kode for beregning av reaksjonskrefter -og momenter med Newton-Euler mekanikk.

```
function [reaksjonskrefter_og_dreiemomenter_maksverdier] = reaksjonskrefter_og_dreiemomenter_inn()
%Funksjon for beregning av reaksjonskrefter
%Reaksjonskrefter beregning
close all
clc

%%Først ble fremover kinematikk definert.

%Denavit-Hartenberg-parametre: Fra 6 til 4 parametre.
%Definerte grad av frihet, lagde en symmetrisk tabell for parametrene(DH),
%identitetsmatrise(T) som ble brukt til å ganges med
%transformasjonsmatrisene, rammen for en transformasjonsmatrise(T_H),
%dimensjonerog leddvinklene(q1-->q6) i form av variabler(syms).

DOF=6;
DH= sym('DH', [DOF, 4]);
T=eye(4);
T_H = zeros(4,4);
L1=0.3; %Lengden på forbindelsen mellom z-aksene til motor2 og motor3.
L2=0.15; %Lengden på forbindelsen mellom x-aksene til motor4 og motor5.
Lm2=(0.072+0.056); Lm3=(0.034+0.0436); Lm4=(0.034+0.0436); Lm5=(0.034+0.0436); Lm6=0.042;
%Motorlengder(Lm).
Hm2=0.057; Hm3=0.035; Hm4=0.035; Hm5=0.035; Hm6=0.042; %Motorhøyder(Hm).
Bm2=Hm2;Bm3=Hm3;Bm4=Hm4;Bm5=Hm5;Bm6=Hm6;
Hv=0.03;Lv=0.052;Bv=0.03;                                   %Definerte total høyde, lengde og bredde
for gripearm og servomotor(Hgs, Lgs og Bgs)
Hm=[Hm2;Hm3;Hm4;Hm5;Hm6;Hv]; %Samlet høyder i en vektor.
Bm=[Bm2; Bm3; Bm4; Bm5; Bm6; Bv]; %Samlet bredder i en vektor.
Lm=[Lm2;Lm3;Lm4;Lm5;Lm6;Lv]; %Samlet lengder i en vektor
Lr6=0.02; %Lengden på rotor til motor 6(Lr6).
syms q1 q2 q3 q4 q5 q6 %Leddposisjoner(q1-->q6)

%Definerte DH-parametre[r, alfa, d, theta] for hvert ledd. Laget deretter
%en for-loop for å sette parametrene inn i DH-tabellen, samt finne
%transformasjons(T_H1-->T_H6)- rotasjons- og translasjonsmatriser for hvert
%ledd(q1-->q6). Fant
%også transformasjonsmatriser(T_0_1-->T_0_6) for hvert ledd relativt til det globale
%koordinatsystemet ved å gange dem med transformasjonsmatrisen til det
%forrige leddet. Endte til slutt opp med transformasjonsmatrisen fra
%globalt koordinat-system til tuppen av robotarmen(T_0_6).

parametre1=[0;pi/2;Hm2/2;q1]; %q1
parametre2=[L1;0;0;q2+pi/2]; %q2
parametre3=[(Hm3/2)+(Hm4/2);pi/2;0;q3]; %q3
parametre4=[0;-pi/2;L2;q4]; %q4
parametre5=[0;-pi/2;0;q5-pi/2]; %q5
parametre6=[0;0;Lm6+Lr6+(Hm5/2)+Lv;q6]; %q6
for i=1:DOF
    if i==1
        i_vektor=[1;0;0]; %Enhetsvektor i x-retning.
        j_vektor=[0;1;0]; %Enhetsvektor i y-retning.
        k_vektor=[0;0;1]; %Enhetsvektor i z-retning.
        parametre=parametre1;
    elseif i==2
        parametre=parametre2;
        R_0_1=T(1:3,1:3);%Henter ut rotasjonsmatriser for hvert ledd i forhold til basen(R_0_1--
>R_0_6).
        R_0_1_T=transpose(R_0_1); %Transponerte rotasjonsmatriser, som er inversen av
rotasjonsmatrisene. Rotasjon fra ledd til base(R_0_1_T-->R_0_6_T)
        z_0_1=R_0_1*k_vektor;
        y_0_1=R_0_1*j_vektor;
    elseif i==3
        parametre=parametre3;
        R_1_2=T_H(1:3,1:3); %Rotasjonsmatrise for neste ledd i forhold til det forrige(R_0_1-->R_5_6).
        R_1_2_T=transpose(R_1_2); %Transponert/invers rotasjonsmatrise. Gir rotasjon av forrige ledd i
forhold til det neste.(R_0_1_T-->R_5_6_T)
        R_0_2=T(1:3,1:3);
        R_0_2_T=transpose(R_0_2);
```

```
        z_0_2=R_0_2*k_vektor;
        x_0_2=R_0_2*i_vektor;
    elseif i==4
        parametre=parametre4;
        R_2_3=T_H(1:3,1:3);
        R_2_3_T=transpose(R_2_3);
        R_0_3=T(1:3,1:3);
        R_0_3_T=transpose(R_0_3);
        z_0_3=R_0_3*k_vektor;
        x_0_3=R_0_3*i_vektor;
    elseif i==5
        parametre=parametre5;
        R_3_4=T_H(1:3,1:3);
        R_3_4_T=transpose(R_3_4);
        R_0_4=T(1:3,1:3);
        R_0_4_T=transpose(R_0_4);
        z_0_4=R_0_4*k_vektor;
        y_0_4=R_0_4*j_vektor;

    else
        parametre=parametre6;
        R_4_5=T_H(1:3,1:3);
        R_4_5_T=transpose(R_4_5);
        R_0_5=T(1:3,1:3);
        R_0_5_T=transpose(R_0_5);
        z_0_5=R_0_5*k_vektor;
    end
    DH(i,1)=parametre(1,1);
    DH(i,2)=parametre(2,1);
    DH(i,3)=parametre(3,1);
    DH(i,4)=parametre(4,1);
    T_H=[cos(DH(i,4)) -sin(DH(i,4))*cos(DH(i,2)) sin(DH(i,4))*sin(DH(i,2)) DH(i,1)*cos(DH(i,4));
             sin(DH(i,4)) cos(DH(i,4))*cos(DH(i,2)) -cos(DH(i,4))*sin(DH(i,2)) DH(i,1)*sin(DH(i,4));
             0 sin(DH(i,2)) cos(DH(i,2)) DH(i,3);
             0 0 0 1];
    T=T*T_H;
end
R_5_6=T_H(1:3,1:3);
R_5_6_T=transpose(R_5_6);
R_0_6=T(1:3,1:3);
R_0_6_T=transpose(R_0_6);
z_0_6=R_0_6*k_vektor;

syms q_1 q_2 q_3 q_4 q_5 q_6 %Leddhastigheter(q_1-->q_6).
%Vinkelhastigheter for hvert ledd, relativt til seg selv(w_0_0-->w_6_6).
w_0_0=zeros(3,1);
w_1_1=(R_0_1_T*w_0_0)+(R_0_1_T*k_vektor*q_1);
w_2_2=(R_1_2_T*w_1_1)+(R_0_2_T*z_0_1*q_2);
w_3_3=(R_2_3_T*w_2_2)+(R_0_3_T*z_0_2*q_3);
w_4_4=(R_3_4_T*w_3_3)+(R_0_4_T*z_0_3*q_4);
w_5_5=(R_4_5_T*w_4_4)+(R_0_5_T*z_0_4*q_5);
w_6_6=(R_5_6_T*w_5_5)+(R_0_6_T*z_0_5*q_6);

syms q__1 q__2 q__3 q__4 q__5 q__6 %Leddakselerasjoner
%Tidsderivert av vinkelhastighetene for hvert ledd, relativt til seg selv. Med andre ord
vinkelakselerasjon for hvert ledd i forhold til basen(alfa_0_1-->alfa_0_6)
alfa_0_0=zeros(3,1);
alfa_1_1=(R_0_1_T*alfa_0_0)+(R_0_1_T*k_vektor*q__1)+cross(w_1_1,(R_0_1_T*k_vektor*q_1));
alfa_2_2=(R_1_2_T*alfa_1_1)+(R_0_2_T*z_0_1*q__2)+cross(w_2_2,(R_0_2_T*z_0_1*q_2));
alfa_3_3=(R_2_3_T*alfa_2_2)+(R_0_3_T*z_0_2*q__3)+cross(w_3_3,(R_0_3_T*z_0_2*q_3));
alfa_4_4=(R_3_4_T*alfa_3_3)+(R_0_4_T*z_0_3*q__4)+cross(w_4_4,(R_0_4_T*z_0_3*q_4));
alfa_5_5=(R_4_5_T*alfa_4_4)+(R_0_5_T*z_0_4*q__5)+cross(w_5_5,(R_0_5_T*z_0_4*q_5));
alfa_6_6=(R_5_6_T*alfa_5_5)+(R_0_6_T*z_0_5*q__6)+cross(w_6_6,(R_0_6_T*z_0_5*q_6));

r_1_s1=(Hm2/2)*(R_0_1_T)*y_0_1; %Distanse-vektorer fra forrige ledd til neste massesenter(r_1_s1--
>r_2_s2)
r_2_s2=(L1)*(R_0_2_T*x_0_2);
r_3_s3=((Hm3/2)+(Hm4/2))*(R_0_3_T*x_0_3);
r_4_s4=(L2)*(R_0_4_T*-y_0_4);
r_5_s5=((Hm5/2)+(Lm6/2))*(R_0_5_T*z_0_5);
r_6_s6=(Lv/2)*(R_0_6_T*z_0_6);
q=[q1;q2;q3;q4;q5;q6];
q_posisjon=[0;0;0;0;0;0];
```

```
r_2_s1=[0;0;0]; %Distanse-vektorer fra neste ledd til forrige massesenter(r_2_s1-->r_tupp_s6).
r_3_s2=[0;0;0];
r_4_s3=[0;0;0];
r_5_s4=[0;0;0];
r_6_s5=(Lr6+(Lm6/2))*(-R_0_5_T*z_0_5);
r_tupp_s6=(Lv/2)*(-R_0_6_T*z_0_6);

r_1_2=(Hm2/2)*R_0_1_T*y_0_1; %Distanse-vektorer fra forrige ledd til neste ledd(r_1_2-->r_6_tupp).
r_2_3=(L1)*(R_0_2_T*x_0_2);
r_3_4=((Hm3/2)+(Hm4/2))*(R_0_3_T*x_0_3);
r_4_5=(L2)*(R_0_4_T*-y_0_4);
r_5_6=((Hm5/2)+Lm6+Lr6)*(R_0_5_T*z_0_5);
r_6_tupp=(Lv)*(R_0_6_T*z_0_6);

g=-9.81;
g1=(R_0_1_T*k_vektor)*g; %Tyngdekraftakselerasjon i vektorform for hvert ledd/massesenter(g1-->g6).
g2=(R_0_2_T*k_vektor)*g;
g3=(R_0_3_T*k_vektor)*g;
g4=(R_0_4_T*k_vektor)*g;
g5=(R_0_5_T*k_vektor)*g;
g6=(R_0_6_T*k_vektor)*g;

%Lineær akselerasjon for hvert ledd/ende i forhold til seg selv(a_e_0_0-->a_e_6_6).
a_e_0_0=zeros(3,1);
a_1_s_1=(R_0_1_T*a_e_0_0)+cross(alfa_1_1,r_1_s1)+cross(w_1_1,(cross(w_1_1,r_1_s1))); %Lineær
akselerasjon for hvert massesenter/ende(a_1_s1-->a_6_s6).
a_e_1_1=(R_0_1_T*a_e_0_0)+cross(alfa_1_1,r_1_2)+cross(w_1_1,(cross(w_1_1,r_1_2)));
a_2_s_2=(R_1_2_T*a_e_1_1)+cross(alfa_2_2,r_2_s2)+cross(w_2_2,(cross(w_2_2,r_2_s2)));
a_e_2_2=(R_1_2_T*a_e_1_1)+cross(alfa_2_2,r_2_3)+cross(w_2_2,(cross(w_2_2,r_2_3)));
a_3_s_3=(R_2_3_T*a_e_2_2)+cross(alfa_3_3,r_3_s3)+cross(w_3_3,(cross(w_3_3,r_3_s3)));
a_e_3_3=(R_2_3_T*a_e_2_2)+cross(alfa_3_3,r_3_4)+cross(w_3_3,(cross(w_3_3,r_3_4)));
a_4_s_4=(R_3_4_T*a_e_3_3)+cross(alfa_4_4,r_4_s4)+cross(w_4_4,(cross(w_4_4,r_4_s4)));
a_e_4_4=(R_3_4_T*a_e_3_3)+cross(alfa_4_4,r_4_5)+cross(w_4_4,(cross(w_4_4,r_4_5)));
a_5_s_5=(R_4_5_T*a_e_4_4)+cross(alfa_5_5,r_5_s5)+cross(w_5_5,(cross(w_5_5,r_5_s5)));
a_e_5_5=(R_4_5_T*a_e_4_4)+cross(alfa_5_5,r_5_6)+cross(w_5_5,(cross(w_5_5,r_5_6)));
a_6_s_6=(R_5_6_T*a_e_5_5)+cross(alfa_6_6,r_6_s6)+cross(w_6_6,(cross(w_6_6,r_6_s6)));

m1=1.7; m2=0.36; m3=0.36; m4=0.36; m5=0.23; m6=(0.089); %Definerte masser for motorene(m1 er for motor
2, m2 for motor 3 osv..).
m=[m1; m2; m3; m4; m5; m6]; %Samlet massene i en kolonnevektor.
I1=[(1/12*m(1,1))*((Hm(1,1)^2)+(Lm(1,1)^2)) 0 0; 0 (1/12*m(1,1))*((Bm(1,1)^2)+(Lm(1,1)^2)) 0; 0 0
(1/12*m(1,1))*((Hm(1,1)^2)+(Bm(1,1)^2))];
I2=[(1/12*m(2,1))*((Hm(2,1)^2)+(Lm(2,1)^2)) 0 0; 0 (1/12*m(2,1))*((Bm(2,1)^2)+(Lm(2,1)^2)) 0; 0 0
(1/12*m(2,1))*((Hm(2,1)^2)+(Bm(2,1)^2))];
I3=[(1/12*m(3,1))*((Hm(3,1)^2)+(Lm(3,1)^2)) 0 0; 0 (1/12*m(3,1))*((Bm(3,1)^2)+(Lm(3,1)^2)) 0; 0 0
(1/12*m(3,1))*((Hm(3,1)^2)+(Bm(3,1)^2))];
I4=[(1/12*m(4,1))*((Hm(4,1)^2)+(Lm(4,1)^2)) 0 0; 0 (1/12*m(4,1))*((Bm(4,1)^2)+(Lm(4,1)^2)) 0; 0 0
(1/12*m(4,1))*((Hm(4,1)^2)+(Bm(4,1)^2))];
I5=[(1/12*m(5,1))*((Hm(5,1)^2)+(Lm(5,1)^2)) 0 0; 0 (1/12*m(5,1))*((Bm(5,1)^2)+(Lm(5,1)^2)) 0; 0 0
(1/12*m(5,1))*((Hm(5,1)^2)+(Bm(5,1)^2))];
I6=[(1/12*m(6,1))*((Hm(6,1)^2)+(Lm(6,1)^2)) 0 0; 0 (1/12*m(6,1))*((Bm(6,1)^2)+(Lm(6,1)^2)) 0; 0 0
(1/12*m(6,1))*((Hm(6,1)^2)+(Bm(6,1)^2))]; %Beregner treghetsmomentet om de 3 aksene ved massesenteret



q=[q1;q2;q3;q4;q5;q6];%Leddposisjoner i vektorform. Variabler.
q_dot=[q_1;q_2;q_3;q_4;q_5;q_6];%Leddhastigheter i vektorform. Variabler.
q_dobbel_dot=[q__1;q__2;q__3;q__4;q__5;q__6];%Leddakselerasjoner i vektorform. Variabler
q_dot_max=[pi/2;-pi/4;pi/4;pi;-pi;4*pi]; %Maksimale leddhastigheter.
q_dobbel_dot_max=[pi/4;pi/8;pi/8;pi/2;pi;8*pi]; %Maksimale leddakselerasjoner.
reaksjonskrefter_og_dreiemomenter=sym('F',[6,6]); %Samler reaksjonkrefter(3 øverste rader), og
dreimomenter(3 nederste rader) i en matrise.

for i=1:5 %Beregner reaksjonskrefter og dreiemomenter ved de 5 ulike posisjonene.
    if i==1
        %q_posisjon=deg2rad([-81.36; -41.25; -60.16; 98.62; -1.72; -168.47]); %Posisjon 1
        q_posisjon=zeros(6,1);
        F_ytre=[0;0;0];
    elseif i==2
        reaksjonskrefter_og_dreiemomenter_1=reaksjonskrefter_og_dreiemomenter;
        q_posisjon=deg2rad([-84.8; -53.29; -4.01; 94; 2.29; 147.26]); %Posisjon 2
        F_ytre=[0;0;0];
    elseif i==3
        reaksjonskrefter_og_dreiemomenter_2=reaksjonskrefter_og_dreiemomenter;
```

```
        q_posisjon=deg2rad([-91.11; -25.78; -24.06; 89.13; -0.74; 139.48]); %Posisjon 3
        F_ytre=[0;0;0];
    elseif i==4
        reaksjonskrefter_og_dreiemomenter_3=reaksjonskrefter_og_dreiemomenter;
        q_posisjon=deg2rad([-68.75;-38.39;-46.98;110.62;1.72;175.96]); %Posisjon 4
        alfa=pi/4;
        m_lokk=2.05; l_lokk=0.22; g=9.81; syms F_ytre;
        F_ytre=round(solve(((m_lokk*g)*(l_lokk/2)*cos(alfa))==F_ytre*l_lokk, F_ytre),2); %Størrelse på
kraften som virker fra lokket på vaffeljernet på robotarmen ved tuppen(Fz).
        F_z=round(cos(alfa)*F_ytre,2);
        F_x=round(sin(alfa)*F_ytre,2);
        F_ytre=round(subs([F_x;0;F_z],q,q_posisjon),2);
    elseif i==5
        reaksjonskrefter_og_dreiemomenter_4=reaksjonskrefter_og_dreiemomenter;
        q_posisjon=deg2rad([-51.56; -67.04; -18.91; 128.36; 2.49; 176.85]); %Posisjon 5
        alfa=0; %Vinkelen lokket er løftet opp(alfa)
        m_lokk=2.05; l_lokk=0.22; g=9.81; syms F_ytre; %Parametre for å beregne kraften(mass lokk,
lengde lokk, tyngdekraftakselerasjon).
        F_ytre=round(solve(((m_lokk*g)*(l_lokk/2)*cos(alfa))==F_ytre*l_lokk, F_ytre),2); %Størrelse på
kraften som virker fra lokket på vaffeljernet på robotarmen ved tuppen(F_ytre).
        F_z=round(cos(alfa)*F_ytre,2); %Kraft i z-retning.
        F_x=round(sin(alfa)*F_ytre,2); %Kraft i x-retning
        F_ytre=round(subs([F_x;0;F_z],q,q_posisjon),2); %Total kraft i vektorform. I tuppens
koordinatsystem.
    end
    F6=round(subs((R_0_6_T*F_ytre)-(m6*g6)+(m6*a_6_s_6),q,q_posisjon),5);F6=subs(F6,q_dot,q_dot_max);
F6=subs(F6,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(1:3,6)=round(subs(R_5_6*F6,q,q_posisjon),2); %Beregner
reaksjonskrefter(F1-->F6)
    F5=round(subs((R_5_6*F6)-(m5*g5)+(m5*a_5_s_5),q,q_posisjon),5); F5=subs(F5,q_dot,q_dot_max);
F5=subs(F5,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(1:3,5)=round(subs(R_4_5*F5,q,q_posisjon),2);
    F4=round(subs((R_4_5*F5)-(m4*g4)+(m4*a_4_s_4),q,q_posisjon),5); F4=subs(F4,q_dot,q_dot_max);
F4=subs(F4,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(1:3,4)=round(subs(R_3_4*F4,q,q_posisjon),2);
    F3=round(subs((R_3_4*F4)-(m3*g3)+(m3*a_3_s_3),q,q_posisjon),5); F3=subs(F3,q_dot,q_dot_max);
F3=subs(F3,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(1:3,3)=round(subs(R_2_3*F3,q,q_posisjon),2);
    F2=round(subs((R_2_3*F3)-(m2*g2)+(m2*a_2_s_2),q,q_posisjon),5); F2=subs(F2,q_dot,q_dot_max);
F2=subs(F2,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(1:3,2)=round(subs(R_1_2*F2,q,q_posisjon),2);
    F1=round(subs((R_1_2*F2)-(m1*g1)+(m1*a_1_s_1),q,q_posisjon),5); F1=subs(F1,q_dot,q_dot_max);
F1=subs(F1,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(1:3,1)=round(subs(R_0_1*F1,q,q_posisjon),2);

    T_ytre=[0;0;0];%Ytre dreiemomenter. Som er lik 0 om alle akser.

    T6=round(subs(((R_0_6_T*T_ytre)-
(cross(F6,r_6_s6)+cross((R_0_6_T*F_ytre),r_tupp_s6)+(I6*alfa_6_6)+cross(w_6_6,(I6*w_6_6)))),q,q_posisjo
n),5); %Beregner dreiemomenter(T1-->T6)
    T6=subs(T6,q_dot,q_dot_max);T6=subs(T6,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(4:6,6)=round(subs((R_5_6*T6),q,q_posisjon),2);
    T5=round(subs(((R_5_6*T6)-
(cross(F5,r_5_s5)+cross((R_5_6*F6),r_6_s5)+(I5*alfa_5_5)+cross(w_5_5,(I5*w_5_5)))),q,q_posisjon),5);
    T5=subs(T5,q_dot,q_dot_max);T5=subs(T5,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(4:6,5)=round(subs(R_4_5*T5,q,q_posisjon),2);
    T4=round(subs(((R_4_5*T5)-
(cross(F4,r_4_s4)+cross((R_4_5*F5),r_5_s4)+(I4*alfa_4_4)+cross(w_4_4,(I4*w_4_4)))),q,q_posisjon),5);
    T4=subs(T4,q_dot,q_dot_max);T4=subs(T4,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(4:6,4)=round(subs(R_3_4*T4,q,q_posisjon),2);
    T3=round(subs(((R_3_4*T4)-
(cross(F3,r_3_s3)+cross((R_3_4*F4),r_4_s3)+(I3*alfa_3_3)+cross(w_3_3,(I3*w_3_3)))),q,q_posisjon),5);
    T3=subs(T3,q_dot,q_dot_max);T3=subs(T3,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(4:6,3)=round(subs(R_2_3*T3,q,q_posisjon),2);
    T2=round(subs(((R_2_3*T3)-
(cross(F2,r_2_s2)+cross((R_2_3*F3),r_3_s2)+(I2*alfa_2_2)+cross(w_2_2,(I2*w_2_2)))),q,q_posisjon),5);
    T2=subs(T2,q_dot,q_dot_max);T2=subs(T2,q_dobbel_dot,q_dobbel_dot_max);
    reaksjonskrefter_og_dreiemomenter(4:6,2)=round(subs(R_1_2*T2,q,q_posisjon),2);
    T1=round(subs(((R_1_2*T2)-
(cross(F1,r_1_s1)+cross((R_1_2*F2),r_2_s1)+(I1*alfa_1_1)+cross(w_1_1,(I1*w_1_1)))),q,q_posisjon),5);
    T1=subs(T1,q_dot,q_dot_max);T1=subs(T1,q_dobbel_dot,q_dobbel_dot_max)
    reaksjonskrefter_og_dreiemomenter(4:6,1)=round(subs(R_0_1*T1,q,q_posisjon),2);
end
```

```
reaksjonskrefter_og_dreiemomenter_5=reaksjonskrefter_og_dreiemomenter;

reaksjonskrefter_og_dreiemomenter_1 %Ved posisjon 1
reaksjonskrefter_og_dreiemomenter_2 %Ved posisjon 2
reaksjonskrefter_og_dreiemomenter_3 %Ved posisjon 3
reaksjonskrefter_og_dreiemomenter_4 %Ved posisjon 4
reaksjonskrefter_og_dreiemomenter_5 %Ved posisjon 5

reaksjonskrefter_og_dreiemomenter_maksverdier=sym('F',[6,6]); %Fant maksimale verdier ved å sammenligne
alle posisjoner, og ta den største absoluttverdien.
for i=1:6
    for j=1:6
        reaksjonskrefter_og_dreiemomenter_maksverdier(i,j)=reaksjonskrefter_og_dreiemomenter_1(i,j);
        if
abs(reaksjonskrefter_og_dreiemomenter_2(i,j))>abs(reaksjonskrefter_og_dreiemomenter_maksverdier(i,j))

reaksjonskrefter_og_dreiemomenter_maksverdier(i,j)=reaksjonskrefter_og_dreiemomenter_2(i,j);
        end
        if
abs(reaksjonskrefter_og_dreiemomenter_3(i,j))>abs(reaksjonskrefter_og_dreiemomenter_maksverdier(i,j))

reaksjonskrefter_og_dreiemomenter_maksverdier(i,j)=reaksjonskrefter_og_dreiemomenter_3(i,j);
        end
        if
abs(reaksjonskrefter_og_dreiemomenter_4(i,j))>abs(reaksjonskrefter_og_dreiemomenter_maksverdier(i,j))

reaksjonskrefter_og_dreiemomenter_maksverdier(i,j)=reaksjonskrefter_og_dreiemomenter_4(i,j);
        end
        if
abs(reaksjonskrefter_og_dreiemomenter_5(i,j))>abs(reaksjonskrefter_og_dreiemomenter_maksverdier(i,j))

reaksjonskrefter_og_dreiemomenter_maksverdier(i,j)=reaksjonskrefter_og_dreiemomenter_5(i,j);
        end
    end
end
```

Vedlegg I-Kode for dimensjoneringsberegninger av bæredeler

```
function [dimensjonering] = dimensjonering()
%Funksjon for beregning av spenninger for dimensjonering. Ser bort fra
%skjærspenning pga bøyning(engelsk: "transverse shear") Maks ved nøytral
%akse, som regel langt mindre enn skjærspenning pga torsjon, og ikke veldig
%mye større enn gjennomsnittlig skjærspenning.

clc

%Spenningsberegning motorhus/bjelke(80x80mm m. 5mm tykkelse)

F_A_1=40.6; F_R_X_1=5.9; F_R_Y_1=1.5;%Maks aksialkraft(F_A_1), og radielle
krefter(F_R_X_1 og F_R_Y_1) ved tuppen av rotor til motor 1.
l1=(56+43.5+30); %Lengde av motor 1+lengden av rotoren til motor 1:
M_Z_1=1370; M_X_1=-5980-(F_R_Y_1*l1);M_Y_1=-4600-(F_R_X_1*l1); %Maksimale
dreimomenter om hver akse. De radielle kreftene skaper dreiemoment om henholdsvis
x-og y-aksen.
h1_y=80;b1_y=h1_y;h1_i=70;b1_i=h1_i; %Indre og ytre høyde og bredde(h1_y/b1_y og
h1_i/b1_i)
A_1=(h1_y*b1_y)-(h1_i*b1_i); %Spenningsarealet.
I_X_1=(1/12)*((b1_y*h1_y^3)-(b1_i*h1_i^3)); %Treghetsmoment om x-aksen.
I_Y_1=I_X_1; %Treghetsmoment om y-aksen.
J_1=2*I_X_1; %Polart treghetsmoment.
sigma_A_1=F_A_1/A_1; %Aksialspenning(sigma_A_1), i z-retning.
sigma_z_1=sigma_A_1; %Total normalspenning i z-retning.
sigma_b_x_1=(M_Y_1*h1_y)/I_Y_1; %Bøyespenning(sigma_b_x_1) i x-retning.
sigma_x_1=sigma_b_x_1; %Total normalspenning(sigma_x_1) i x-retning.
sigma_b_y_1=(M_X_1*h1_y)/I_X_1; %Bøyespenning(sigma_b_y_1) i y-retning.
sigma_y_1=sigma_b_y_1;  %Total normalspenning(sigma_y_1) i y-retning.
```

```
tau_xy_1=((M_Z_1*h1_y)/J_1)+(sqrt(F_R_X_1^2+F_R_Y_1^2)/A_1); %Total skjærspenning
i xy-planet(tau_xy_1) pga. dreiemoment om z-akse og skjærspenning pga radielle
krefter.
sigma_e_1=sqrt((sigma_z_1^2)+(sigma_x_1^2)+(sigma_y_1^2)-
((sigma_x_1*sigma_z_1)+(sigma_x_1*sigma_y_1)+(sigma_z_1*sigma_y_1))+(3*(tau_xy_1^2
))); %Ekvivalentspenningen(sigma_e_1).
Re_316=200; %Flytegrense for rustfritt 316-stål.
Sikkerhetsfaktor_bjelke1=Re_316/sigma_e_1; %Sikkerhetsfaktor for bjelken(80x80mm
m. 5mm tykkelse).

%Skrueforbindelse til benken. 6 skruer.
%Antar at radielle krefter er likt
%fordelt mellom de 6 boltene.
E_staal=210*10^3; %Elastisetetsmodul for stål(E_staal)
lm_S1=2; d_S1=10.86; %Lengde på klemsone materiale (lm_S1), samt
skruediameter(d_S1)
km_S1=((6*2.2*d_S1^2)*E_staal)/lm_S1; %Stivhet materiale (km_S1)
A_S1=(84.27)*6; r_S1=d_S1/2; I_S1=((pi/4)*r_S1^4)*6; J_S1=2*I_S1; %M12-skruer.
Spenningsareal, radius og treghetsmomenter(A_S1, r_S1, I_S1, J_S1)
lsg_S1=lm_S1; %Gjenget del av skrue i klem(lsg_S1)
ks_S1=(A_S1*E_staal)/lsg_S1; %Stivhet skrue(ks_S1)
k_faktor_S1=ks_S1/(ks_S1+km_S1) ;%Kraftfordelingsfaktor(k_faktor_S1)
M_Z_S1=M_Z_1; M_X_S1=M_X_1; M_Y_S1=M_Y_1;F_R_X_S1=F_R_X_1;F_R_Y_S1=F_R_Y_1;
%Krefter og momenter, som er de samme som for motorhuset/bjelken.
sigma_b_x_S1=(M_Y_S1*r_S1)/I_S1; %Bøyespenning i x-retning(sigma_b_x_S1)
sigma_x_S1=sigma_b_x_S1; %Total normalspenning i x-retning(sigma_x_S1)
sigma_b_y_S1=(M_X_S1*r_S1)/I_S1;%Bøyespenning i y-retning(sigma_b_y_S1)
sigma_y_S1=sigma_b_y_S1; %Total normalspenning i y-retning(sigma_y_S1)
tau_xy_S1=(M_Z_S1*r_S1)/J_S1+(sqrt(F_R_Y_S1^2+F_R_X_S1^2)/A_S1); %Total
skjærspenning i xy-planet(tau_xy_S1)
sigma_e_y_S1=(sqrt((sigma_x_S1^2)+(sigma_y_S1^2)-
(sigma_x_S1*sigma_y_S1)+(3*(tau_xy_S1^2)))*k_faktor_S1);%Ekvivalentspenningen pga
ytre belastning(sigma_e_y_S1)
Re_88=640; %Flytegrense for 8.8-skruer(Re_88).
F_0_S1=Re_88*A_S1*0.5;%Forspenningskraft, 50% av flytegrense(F_0_S1)
sigma_S1=F_0_S1/A_S1; %Normalspenning pga forspenning(sigma_S1)
tau_S1=0.48*sigma_S1;%Skjærspenning pga forspenning(tau_S1), 48% av
normalspenning.
sigma_e_S1=sqrt((sigma_S1^2)+(2*tau_S1^2)); %Ekvivalentspenning pga
forspenning(sigma_e_S1)
t_sigma_e_S1=(sigma_e_y_S1/6)+sigma_e_S1; %Summen av ytre spenning og
forspenning(t_sigma_e_S1), for hver av de 6 skruene.
Sikkerhetsfaktor_S1=Re_88/t_sigma_e_S1; %Sikkerhetsfaktoren for skrueforbindelsen
med 6 skruer.

%Beregne kritisk bøyespenning i 2mm stålplate/sylinder. Vrimomentet om
%rotasjonsaksen til motoren(M_Z_1) skaper også bøyespenning i tversnittet til
%platen. Studerer tversnittet som boltene går i gjennom. Tversnittet går i
%gjennom hullet til 2 bolter, med 60 grader mellom hvert hull.
%Hull til bolter er 13mm.
r_p1=168/2; d_p1=13; t_p1=2; %Radius plate, boltehull(d_p1) og
platetykkelse(t_p1).
l_p1=sqrt((2*r_p1^2)-(2*r_p1^2*cos(deg2rad(120)))); %Beregner lengden av
tversnittet(l_p1) med cosinussetningnen.
%Om z-aksen.
l_h_z1=sin(pi/6)*50; %Lengde fra senter av hull til z-aksen(l_h_z1)
I_Z_p1=((t_p1*l_p1^3)/12)-2*(((2*d_p1^3)/12)+(2*d_p1*l_h_z1^2)); %Treghetsmoment
om z-aksen for tversnittet(I_Z_p1).
```

```
sigma_p1_z=(M_Z_1*(l_p1/2))/I_Z_p1; %Bøyespenning om z-aksen i
tversnittet(sigma_p1).
Sikkerhetsfaktor_p1_z=Re_316/sigma_p1_z; %Sikkerhetsfaktor mot flyting med bøyning
om z-aksen.
%Om x-aksen:
I_X_p1=((l_p1*t_p1^3)/12)-2*(((13*2^3)/12)); %Treghetsmoment om x-aksen for
tversnittet(I_X_p1).
sigma_p1_x=(M_X_1*(t_p1/2))/I_X_p1; %Bøyespenning om z-aksen i
tversnittet(sigma_p1).
Sikkerhetsfaktor_p1_x=Re_316/sigma_p1_x; %Sikkerhetsfaktor mot flyting med bøyning
om x-aksen.
%Siden momentet om y-aksen er mindre enn om x-aksen, og tversnittet vil se
%identisk ut, vil en ikke beregne denne sikkerhetsfaktoren.

%Beregning av nødvendig forspenning for 4stk. M8x1.25mm skruer. Beregne
%rotasjonsvinkel ved montering.
l2=57/2; %l2, halv bredde motor 1, som er avstanden fra friksjonskraften til
rotasjonspunktet.
Sikkerhetsfaktor_bolter_motor1=10; %Sikkerhetsfaktor for bolteforbindelsen. Setter
den til 10.
T1=M_Z_1*Sikkerhetsfaktor_bolter_motor1; %Totalt dreiemoment(T1) som
friksjonskreftene må overvinne. Moment om rotasjonsaksen ganget med
sikkerhetsfaktoren.
FR1=((T1/l2)/4); %Total friksjonskraft hver bolt må tilføre(FR1);
f_koeffisient_staal= 0.6; %Friksjonskoeffisient stål mot stål. Tørre overflater.
F0_1=FR1/f_koeffisient_staal; %Normalkraft/forspenningskraft fra hver bolt(F0_1)
A_s1=36.61; E_staal=210000;lk_1=(11.5); %Spenningsareal til M8-skrue,
elastisistetsmodul, og klemlengde, hvor hele bolten er gjenget.
ks1_1=A_s1*E_staal/lk_1; %Total stivhet i skrue, dvs. stivhet i gjenget del.
delta_s1=F0_1/ks1_1; %Deformasjon(delta_s1) i skrue ved ønsket forspenningskraft.
p_s1=1.25; %Stigning i skruen(p_s1)
rot_vinkel_1=round(rad2deg((delta_s1/p_s1)*2*pi),1); %Total rotasjonsvinkel for å
få ønska forspenning og sikkerhetsfaktor på 10(rot_vinkel_1).

%Beregning av nødvendig forspenning for 4stk. M6x1-settskruer. Beregne
%rotasjonsvinkel ved montering.
a1y=20; a1i=12; l3=a1i/2; %Ytre og indre diameter for akslinger(a1y og a1i) og l3,
radius til aksling motor 1, som er avstanden fra friksjonskraften til
rotasjonspunktet.
Sikkerhetsfaktor_aksling1=10; %Sikkerhetsfaktor for settskrueforbindelsen. Setter
den til 10.
Ta1=M_Z_1*Sikkerhetsfaktor_aksling1; %Totalt dreiemoment(Ta1) som
friksjonskreftene må overvinne. Moment om rotasjonsaksen ganget med
sikkerhetsfaktoren.
FRa1=((Ta1/l3)/4); %Total friksjonskraft hver settskrue må tilføre(FRa1);
F0_a1=FRa1/f_koeffisient_staal; %Normalkraft/forspenningskraft fra hver
bolt(F0_a1)
A_a1=20.12; lk_a1=((a1y/2)-l3); %Spenningsareal til M6-skrue, elastisistetsmodul,
og klemlengde, hvor hele bolten er gjenget.
ks_a1=A_a1*E_staal/lk_a1; %Total stivhet i skrue, dvs. stivhet i gjenget del.
delta_a1=F0_a1/ks_a1; %Deformasjon(delta_a1) i skrue ved ønsket forspenningskraft.
p_a1=1; %Stigning i skruen(p_a1)
rot_vinkel_a1=round(rad2deg((delta_a1/p_a1)*2*pi),1); %Total rotasjonsvinkel for å
få ønska forspenning og sikkerhetsfaktor på 10(rot_vinkel_a1).

%Dimensjonering mot flyting første sammenkobling for aksling og plate.
M_Y_1=-4600;M_X_1=-5980;%Momenter om x- og y-aksen. Ved rotortuppen. Det samme som
ved foregående beregningner, bare at de radielle kreftene ikke generer noe moment.
Vrimoment om z-aksen er det samme.
```

```
r_i_a_1=13/2;d1=20;r_y_a_1=d1/2; %Indre og ytre radius for
akslingen/forbindelsen(r_i_a_1 og r_y_a_1).
D1=60;r1=2; r_d_1=r1/d1;d_D_1=D1/d1; %Diameter på største sylinder(D1),
avrundingsradius(r_d_1), forhold mellom stor og liten sylinder(d_D_1).
stress_k1=2; %Spenningskonsentrasjon pga diameterforskjell.
A_2=pi*((r_y_a_1^2)-(r_i_a_1^2)); %Areal(A_2)
I_2=(pi/4)*((r_y_a_1^4)-(r_i_a_1^4)); %Treghetsmoment(I_2)
J_2=2*I_2; %Polart treghetsmoment(J_2)
sigma_A_2=F_A_1/A_2; %Normalspenning pga. aksialkraft(sigma_A_2)
sigma_b_x_2=(M_Y_1*r_y_a_1)/I_2; %Bøyespenning i x-retning(sigma_b_x_2).
sigma_x_2=sigma_b_x_2; %Total normalspenning i x-retning(sigma_x_2).
sigma_b_y_2=(M_X_1*r_y_a_1)/I_2; %Bøyespenning i y-retning(sigma_b_y_2)
sigma_y_2=sigma_b_y_2; %Total normalspenning i y-retning(sigma_y_2)
sigma_z_2=sigma_A_2; %Normalspenning i z-retning(sigma_z_2)
tau_xy_2=(M_Z_1*r_y_a_1)/J_2+(sqrt(F_R_Y_1^2+F_R_X_1^2)/A_2); %Total
skjærspenning(tau_xy_2)
sigma_e_2=sqrt((sigma_z_2^2)+(sigma_y_2^2)+(sigma_x_2^2)-
((sigma_x_2*sigma_y_2)+(sigma_x_2*sigma_z_2)+(sigma_z_2*sigma_y_2))+(3*(tau_xy_2^2
)))*stress_k1; %Ekvivalentspenningen(sigma_e_2)
Sikkerhetsfaktor_2=Re_316/sigma_e_2; %Sikkerhetsfaktor.

%Dimensjonering mot flyting i skrueforbindelse mellom aksling for motor 1
%og plate for motor 2:
lm1_S2=2; lm2_S2=2; d_S2=4.48; %Lengde på klemsone materiale 1 og 2(lm1_S2,
lm2_S2), samt skruediameter(d_S2)
km1_S2=((4*2.2*d_S2^2)*E_staal)/lm1_S2; %Stivhet materiale 1(km1_S2)
km2_S2=((4*2.2*d_S2^2)*E_staal)/lm2_S2; %Stivhet materiale 2(km2_S2)
km_S2=(km1_S2*km2_S2)/(km1_S2+km2_S2); %Stivhet totalt for materiale i klem(km_S2)
A_S2=(14.18)*4; r_S2=d_S2/2; I_S2=((pi/4)*r_S2^4)*4; J_S2=2*I_S2; %M5-skruer.
Spenningsareal, radius og treghetsmomenter(A_S2, r_S2, I_S2, J_S2)
lsg_S2=lm1_S2+lm2_S2; %Gjenget del av skrue i klem(lsg_S2)
ks_S2=(A_S2*E_staal)/lsg_S2; %Stivhet skrue(ks_S2)
k_faktor_2=ks_S2/(ks_S2+km_S2); %Kraftfordelingsfaktor(k_faktor_2)

F_A_S2=F_A_1/4; %Aksialkraft i hver skrue(F_A_S2)
sigma_A_S2=(F_A_S2)/A_S2; %Aksialspenning i hver skrue(sigma_A_S2)
sigma_b_x_S2=(M_Y_1*r_S2)/I_S2; %Bøyespenning i x-retning(sigma_b_x_S2)
sigma_x_S2=sigma_b_x_S2; %Total normalspenning i x-retning(sigma_x_S2)
sigma_b_y_S2=(M_X_1*r_S2)/I_S2; %Bøyespenning i y-retning(sigma_b_y_S2)
sigma_y_S2=sigma_b_y_S2; %Total normalspenning i y-retning(sigma_y_S2)
sigma_z_S2=sigma_A_S2; %Total normalspenning i z-retning(sigma_z_S2)
tau_xy_S2=(M_Z_1*r_S2)/J_S2+(sqrt(F_R_X_1^2+F_R_Y_1^2)/A_S2); %Total skjærspenning
i xy-planet(tau_xy_S2)
sigma_e_ytre=(sqrt((sigma_z_S2^2)+(sigma_y_S2^2)+(sigma_x_S2^2)-
((sigma_x_S2*sigma_z_S2)+(sigma_y_S2*sigma_z_S2)+(sigma_x_S2*sigma_y_S2))+(3*(tau_
xy_S2^2)))*k_faktor_2); %Ekvivalentspenning pga ytre belastning(sigma_e_ytre)
F_0_S2=Re_88*A_S2*0.5; %Forspenningskraft(F_0_S2)
sigma_S2=F_0_S2/A_S2; %Forspenning(sigma_S2)
tau_S2=0.48*sigma_S2; %Skjærspenning pga forspenning(tau_S2)
sigma_e_forspenning_2=sqrt((sigma_S2^2)+(2*tau_S2^2)); %Ekvialentspenning pga
forspenning(sigma_e_forspenning_2)
total_sigma_e_S2=(sigma_e_ytre+sigma_e_forspenning_2); %Total
ekvialentspenning(total_sigma_e_S2)
Sikkerhetsfaktor_S2=Re_88/total_sigma_e_S2; %Sikkerhetsfaktor mot flyting i skrue.

%Sjekker at deformasjonen i materialet ikke blir for stor, gitt den tynne
tykkelsen.
%Dimensjonering mot deformasjon i plate til benken:
```

```
F_maks_ytre_S1=(sigma_e_y_S1*(A_S1))/k_faktor_S1; %Maksimal ytre
kraft(F_maks_ytre)
Fm_S1=F_0_S1+(F_maks_ytre_S1*(1-k_faktor_S1)); %Maksimal kraft som virker på
materialet(Fm_S1)
delta_m_S1=Fm_S1/km_S1; %Deformasjon i platen(delta_m_S1)
Deformasjon_tillatt_1=0.03*lm_S1; %Definerer tillatt deformasjon som er tre
prosent av opprinnelig tykkelse.
Sikkerhetsfaktor_deformasjon_1=Deformasjon_tillatt_1/delta_m_S1;

%Dimensjonering mot deformasjon i for materialet i
%klemforbindelsen/skrueforbindelsen mellom motorholder til motor 2 og
%forbindelse på aksling til motor 1:
F_maks_ytre_S2=(sigma_e_ytre*(A_S2))/k_faktor_2; %Maksimal ytre kraft(F_maks_ytre)
Fm_S2=F_0_S2+(F_maks_ytre_S2*(1-k_faktor_2)); %Maksimal kraft i materiale(Fm_S2)
delta_m1_S2=Fm_S2/km1_S2; %Deformasjon i materiale 1/motorholder for motor
2(delta_m1_S2)
delta_m2_S2=Fm_S2/km2_S2; %Deformasjon i materiale 2/sammenkobling på aksling for
motor 1(delta_m2_S2)
if abs(delta_m1_S2) >= abs(delta_m2_S2) %Finner ut om deformasjonen i materiale 1
eller 2 er størst.
    delta_m_S2=delta_m1_S2; %Deformasjonen bør jo være like stor i materiale 1 som
2 siden begge er av stål og like tykke,
else                        %men bruker en if-setning gitt at det var ulik
tykkelse eller ulike materialer.
    delta_m_S2=delta_m2_S2;
end
Deformasjon_tillatt_2=0.03*lm1_S2; %Definerer tillatt deformasjon som er tre
prosent av opprinnelig tykkelse.
Sikkerhetsfaktor_deformasjon_2=Deformasjon_tillatt_2/delta_m_S2;

%Kritisk bøyespenning for sylinder/plate som kobler sammen aksling til
%motor 1 og motorholder til motor 2.
r_p2=100/2; d_p2=7; t_p2=lm2_S2+lm1_S2; %Radius plate(r_p2), boltehull(d_p2) og
platetykkelse(t_p2).
l_p2=sqrt(2*r_p2^2); %Beregner lengden av tversnittet(l_p2) med pytagoras
%Om z-aksen.
l_h_z2=29; %Lengde fra senter av hull til z-aksen(l_h_z1)
I_Z_p2=((t_p2*l_p2^3)/12)-2*(((t_p2*d_p2^3)/12)+(t_p2*d_p2*l_h_z2^2));
%Treghetsmoment om z-aksen for tversnittet(I_Z_p2)
sigma_p2_z=((M_Z_1*(l_p2/2))/I_Z_p2)*(1-k_faktor_2); %Bøyespenning om z-aksen i
tversnittet(sigma_p1).
Sikkerhetsfaktor_p2_z=Re_316/sigma_p2_z; %Sikkerhetsfaktor mot flyting med bøyning
om z-aksen.
%Om x-aksen:
I_X_p2=((l_p2*t_p2^3)/12)-2*(((d_p2*t_p2^3)/12)); %Treghetsmoment om x-aksen for
tversnittet(I_X_p2).
sigma_p2_x=((M_X_1*(t_p2/2))/I_X_p2)*(1-k_faktor_2); %Bøyespenning om x-aksen i
tversnittet(sigma_p2_x).
Sikkerhetsfaktor_p2_x=Re_316/sigma_p2_x; %Sikkerhetsfaktor mot flyting med bøyning
om x-aksen.
%Siden momentet om y-aksen er mindre enn om x-aksen, og tversnittet vil se
%identisk ut, vil en ikke beregne denne sikkerhetsfaktoren.

%Dimensjonering mot flyting for forbindelse koblet på akslingen til motor:
M_Z_2=7510; M_Y_2=1370; M_X_2=-700; F_A_2=-5.97; F_R_X_2=-1.32; F_R_Y_2=23.9;
%Momenter om x-, y- og z-aksen(M_Z_2, M_Y_2, M_X_2). Ved rotortuppen. Aksialkraft
og radielle krefter ved rotortupp(F_A_2, F_R_X_2, F_R_Y_2).
r_i_a_2=13/2;d2=20;r_y_a_2=d2/2; %Indre og ytre radius for
akslingen/forbindelsen(r_i_a_2 og r_y_a_2).
```

```
D2=60;r2=2; r_d_2=r2/d2;d_D_2=D2/d2; %Diameter på største sylinder(D2),
avrundingsradius(r2, forhold mellom stor og liten sylinder(d_D_2).
stress_k2=2; %Stresskonsentrasjon pga diameterforskjell(stress_k2).
A_3=pi*((r_y_a_2^2)-(r_i_a_2^2)); %Areal(A_3)
I_3=(pi/4)*((r_y_a_2^4)-(r_i_a_2^4)); %Treghetsmoment(I_3)
J_3=2*I_3; %Polart treghetsmoment(J_3)
sigma_A_3=F_A_2/A_3; %Normalspenning pga. aksialkraft(sigma_A_3)
sigma_b_x_3=(M_Y_2*r_y_a_2)/I_3; %Bøyespenning langs x-aksen(sigma_b_x_3)
sigma_x_3=sigma_b_x_3; %Total spenning i x-retning(sigma_x_3)
sigma_b_y_3=(M_X_2*r_y_a_2)/I_3; %Bøyespenning i y-retning(sigma_b_y_3)
sigma_y_3=sigma_b_y_3; %Total spenning i y-retning(sigma_y_3)
sigma_z_3=sigma_A_3; %BNormalspenning i z-retning(sigma_z_3)
tau_xy_3=(M_Z_2*r_y_a_2)/J_3+(sqrt(F_R_Y_2^2+F_R_X_2^2)/A_3); %Total skjærspenning
i xy-planet(tau_xy_3).
sigma_e_3=sqrt((sigma_z_3^2)+(sigma_y_3^2)+(sigma_x_3^2)-
((sigma_x_3*sigma_y_3)+(sigma_x_3*sigma_z_3)+(sigma_z_3*sigma_y_3))+(3*(tau_xy_3^2
)))*stress_k2; %Ekvivalentspenningen(sigma_e_3).
Sikkerhetsfaktor_3=Re_316/sigma_e_3; %Sikkerhetsfaktor mot flyting.

%Dimensjonering mot flyting i skruen som holder holder kulelageret ved
%motor 2 på plass:
lm_k1=6; d_k1=9.03; %lengde på klemsone og diameter for skrue(lm_k1, d_k1)
km_k1=((2.2*d_k1^2)*E_staal)/lm_k1; %Stivhet materiale(km_k1)
A_k1=58; r_k1=d_k1/2; I_k1=((pi/4)*r_k1^4)*4; J_k1=2*I_k1;% %M10-skrue. Areal,
radius og treghetsmomenter(A_k1, r_k1, I_k1, J_k1)
lsg_k1=lm_k1; %Gjenget del av skrue i klem(lsg_k1)
ks_k1=(A_k1*E_staal)/lsg_k1; %Stivhet skrue(ks_k1)
k_faktor_k1=ks_k1/(ks_k1+km_k1); %Kraftfordelingsfaktor(k_faktor_k1)

sigma_A_k1=(F_A_2)/A_k1; %Aksialspenning(sigma_A_k1)
sigma_b_x_k1=(M_Y_2*r_k1)/I_k1; %Bøyespenning i x-retning(sigma_b_x_k1)
sigma_x_k1=sigma_b_x_k1; %Total spenning i x-retning(sigma_x_k1).
sigma_b_y_k1=(M_X_2*r_k1)/I_k1;%Bøyespenning i y retning(sigma_b_y_k1)
sigma_y_k1=sigma_b_y_k1; %Total spenning i y-retning(sigma_y_k1)
sigma_z_k1=sigma_A_k1; %Total spenning i z-retning(sigma_z_k1)
tau_xy_k1=(M_Z_2*r_k1)/J_k1+(sqrt(F_R_X_2^2+F_R_Y_2^2)/A_k1); %Total
skjærspenning(tau_xy_k1)
sigma_e_ytre_k1=(sqrt((sigma_z_k1^2)+(sigma_y_k1^2)+(sigma_x_k1^2)-
((sigma_x_k1*sigma_z_k1)+(sigma_y_k1*sigma_z_k1)+(sigma_x_k1*sigma_y_k1))+(3*(tau_
xy_k1^2)))*k_faktor_k1); %Ytre tilført ekvivalentspenning(sigma_e_ytre_k1)
F_0_k1=Re_88*A_k1*0.5; %Forspenningskraft(F_0_k1)
sigma_k1=F_0_k1/A_k1; %Forspenning(sigma_k1)
tau_k1=0.48*sigma_k1; %Skjærspenning av forspenning(tau_k1)
sigma_e_forspenning_k1=sqrt((sigma_k1^2)+(2*tau_k1^2)); %Ekvivalentspenning av
forspenning.
total_sigma_e_k1=(sigma_e_ytre_k1+sigma_e_forspenning_k1); %Total
ekvivalentspenning(total_sigma_e_k1)
Sikkerhetsfaktor_k1=Re_88/total_sigma_e_k1; %Sikkerhetsfaktor

%Dimensjonering mot knekking for motorholder-plate-2 som kulelager-skrue går i
gjennom.
t_mh1=2; b_mh1=60; %Tykkelse og bredde brakett (t_mh1, b_mh1)
A_mh1=t_mh1*b_mh1; %Spenningsareal(A_mh1)
l_mh1=62/2; %Vinkelrett distanse fra fra krefter til knekkepunkt i bunn av platen.
I_X_mh1=(1/12)*(b_mh1*t_mh1^3);
I_Z_mh1=(1/12)*(t_mh1*b_mh1^3);
sigma_euler_z_mh1=((pi^2)*E_staal*I_Z_mh1)/(A_mh1*(l_mh1^2)); %Knekkespenning om
z-aksen(sigma_euler_z_mh1).
```

```
sigma_euler_x_mh1=((pi^2)*E_staal*I_X_mh1)/(A_mh1*(l_mh1^2)); %Knekkespenning om
x-aksen(sigma_euler_x_mh1).
%Om x-aksen:
sigma_b_z_mh1=(M_X_2*(t_mh1/2))/I_X_mh1; %Normalspenning om x-aksen i z-
retning(sigma_b_z_mh1).
%Om z_aksen:
sigma_b_x_mh1=(M_Z_2*(b_mh1/2))/I_Z_mh1; %Bøyespenning om z-aksen i x-
retning(sigma_b_x_mh1).
if sigma_euler_x_mh1/sigma_b_z_mh1>sigma_euler_z_mh1/sigma_b_x_mh1 %Finner ut om
man er nærmest knekkespenning om x- eller z-aksen.
    Sikkerhetsfaktor_knekking_mh1=sigma_euler_z_mh1/sigma_b_x_mh1;
else
    Sikkerhetsfaktor_knekking_mh1=sigma_euler_x_mh1/sigma_b_z_mh1;
end
Sikkerhetsfaktor_knekking_mh1;

%Dimensjonering mot flyting ved hull til skrue for kulelager på motorholder
%til motor nummer 2.
D_h1=11; %Hulldiameter(D_h1)
A_h1=(t_mh1*b_mh1)-(t_mh1*11); %Areal(A_h1)
I_X_h1=((b_mh1*t_mh1^3)/12)-((D_h1*t_mh1^3)/12);%Treghetsmoment om x-aksen(I_X_h1)
I_Z_h1=((t_mh1*b_mh1^3)/12)-((t_mh1*D_h1^3)/12); %Treghetsmoment om z-
aksen(I_Z_h1)
J_h1=(((t_mh1*b_mh1)*(t_mh1^2+b_mh1^2))/12)-(((t_mh1*D_h1)*(t_mh1^2+D_h1^2))/12);
%Polart treghetsmoment(J_h1)
sigma_A_h1=F_A_2/A_h1; %Normalspenning pga. aksialkraft(sigma_A_h1)
sigma_b_x_h1=(M_Z_2*(b_mh1/2)/I_Z_h1); %Bøyespenning langs x-aksen(sigma_b_x_h1)
sigma_x_h1=sigma_b_x_h1; %Total spenning i x-retning(sigma_x_h1)
sigma_b_z_h1=(M_X_2*(t_mh1/2))/I_X_h1; %Bøyespenning i z-retning(sigma_b_z_h1)
sigma_z_h1=sigma_b_z_h1; %Total spenning i Z-retning(sigma_z_h1)
sigma_y_h1=(F_R_Y_2/A_h1); %Normalspenning i y-retning(sigma_y_h1)
tau_xz_h1=(M_Y_2*(t_mh1))/J_h1+(F_R_X_2/A_h1)+sigma_A_h1; %Total skjærspenning i
xz-planet(tau_xz_h1).
sigma_e_h1=sqrt((sigma_z_h1^2)+(sigma_y_h1^2)+(sigma_x_h1^2)-
((sigma_x_h1*sigma_y_h1)+(sigma_x_h1*sigma_z_h1)+(sigma_z_h1*sigma_y_h1))+(3*(tau_
xz_h1^2))); %Ekvivalentspenningen(sigma_e_h1).
Sikkerhetsfaktor_h1=Re_316/sigma_e_h1; %Sikkerhetsfaktor mot flyting.

%Dimensjonering mot flyting for skrueforbindelse mellom aksling til motor 2
%og 150mm brakett.
lm1_S3=3; lm2_S3=2; d_S3=5.35; %Lengde på klemsone materiale 1 og 2(lm1_S3,
lm2_S3), samt skruediameter(d_S3)
km1_S3=((4*2.2*d_S3^2)*E_staal)/lm1_S3; %Stivhet materiale 1(km1_S3)
km2_S3=((4*2.2*d_S3^2)*E_staal)/lm2_S3; %Stivhet materiale 2(km2_S3)
km_S3=(km1_S3*km2_S3)/(km1_S3+km2_S3); %Stivhet materiale totalt(km_S3)
A_S3=(20.12)*4; r_S3=d_S3/2; I_S3=((pi/4)*r_S3^4)*4; J_S3=2*I_S3; %M6-skruer:
Areal, radius, treghetsmomenter
lsg_S3=lm1_S3+lm2_S3; %Lengde klemsone gjenget del skrue(lsg_S3)
ks_S3=(A_S3*E_staal)/lsg_S3; %Stivhet gjenget del skrue(ks_S3)
k_faktor_3=ks_S3/(ks_S3+km_S3); %Kraftfordelingsfaktor(k_faktor_3)

sigma_A_S3=(F_A_2)/A_S3; %Aksialspenning i hver skrue(sigma_A_S3)
sigma_b_x_S3=(M_Y_2*r_S3)/I_S3; %Bøyespenning i x-retning(sigma_b_x_S3)
sigma_x_S3=sigma_b_x_S3; %Total normalspenning i x-retning(sigma_x_S3)
sigma_b_y_S3=(M_X_2*r_S3)/I_S3; %Bøyespenning i y-retning(sigma_b_y_S3)
sigma_y_S3=sigma_b_y_S3; %Total normalspenning i y-retning(sigma_y_S3)
sigma_z_S3=sigma_A_S3; %Total normalspenning i z-retning(sigma_z_S3)
tau_xy_S3=(M_Z_2*r_S3)/J_S3+(sqrt(F_R_X_2^2+F_R_Y_2^2)/A_S3); %Total skjærspenning
i xy-plan(tau_xy_S3)
```

```
sigma_e_ytre_S3=(sqrt((sigma_z_S3^2)+(sigma_y_S3^2)+(sigma_x_S3^2)-
((sigma_x_S3*sigma_z_S3)+(sigma_y_S3*sigma_z_S3)+(sigma_x_S3*sigma_y_S3))+(3*(tau_
xy_S3^2)))*k_faktor_3); %Ekvivalentspenning pga ytre belastning(sigma_e_ytre_S3)
F_0_S3=Re_88*A_S3*0.5; %Forspenningskraft(F_0_S3)
sigma_S3=F_0_S3/A_S3; %Forspenning(sigma_S3)
tau_S3=0.48*sigma_S3; %Skjærspenning pga forspenning(tau_S3)
sigma_e_forspenning_3=sqrt((sigma_S3^2)+(2*tau_S3^2)); %Ekvivalentspenning pga
forspenning(sigma_e_forspenning_3)
total_sigma_e_S3=(sigma_e_ytre_S3+sigma_e_forspenning_3); %Total
ekvivalentspenning i skrue(total_sigma_e_S3)
Sikkerhetsfaktor_S3=Re_88/total_sigma_e_S3; %Sikkerhetsfaktor mot flyting i skrue.

%Beregning av sikkerhetsfaktor mot gliding i krympeforbindelsen for kulelager til
%motor 2.
rn1i=25.98/2; rn1o=60.0/2; ra1i=10.0/2; ra1o=26.0/2; %Ytre og indre radiuser for
hull og kulelager(rn1i, rn1o, ra1i og ra1o)
delta1=ra1o-rn1i; poisson_staal=0.3; %Diametralt pressmonn og poissons
forhold(delta1, poisson_staal)
alfa_n1=((rn1i^3)/(E_staal*((rn1o^2)-(rn1i^2))))*((1-
poisson_staal)+((1+poisson_staal)*((rn1o^2)/(rn1i^2)))); %Influenskoeffisient
hull(alfa_n1)
alfa_a1=((ra1i^3)/(E_staal*((ra1o^2)-(ra1i^2))))*((1-
poisson_staal)+((1+poisson_staal)*((ra1o^2)/(ra1i^2)))); %Influenskoeffisient
kulelager(alfa_a1)
kontakttrykk_1=(delta1)/(2*(alfa_a1+alfa_n1)); %Kontakttrykk generert i
krympeforbindelsen(kontakttrykk_1).
h1=3; %Vrimoment som skal overføres, aksialkraft ved forbindelsen, samt bredden på
hullet(M_Z_2 og F_Z_2, h1).
tau1_a=(F_A_2)/(2*pi*rn1i*h1); %Skjærspenning pga aksialkraft(tau1_a).
tau1_t=(M_Z_2)/(2*pi*(rn1i^2)*h1); %Tangentiell skjærspenning pga
vrimoment(tau1_t)
tau1_total=sqrt((tau1_a^2)+(tau1_t^2));
Sikkerhetsfaktor_glidning_1=kontakttrykk_1/tau1_total; %Sikkerhetsfaktor mot
glidning.

%Beregning av krympespenningene.
sigma1_aa=-kontakttrykk_1; sigma1_na=-kontakttrykk_1; %Krympespenninger i
radialretnning(sigma1_aa, sigma1_na).
sigma1_at=(-kontakttrykk_1*(((ra1o^2)+(ra1i^2))/((ra1o^2)-(ra1i^2))));
%Krympespenning i tangentialretning for kulelager(sigma1_at)
sigma1_nt=(kontakttrykk_1*(((rn1o^2)+(rn1i^2))/((rn1o^2)-(rn1i^2))));
%Krympespenning i tangentialretning for hull(sigma1_nt)
if abs(sigma1_nt) >= abs(sigma1_na)
    krymp_max=sigma1_nt;
else
    krymp_max=sigma1_na;
end
Sikkerhetsfaktor_krymp1=Re_316/krymp_max;

%Kritisk bøyespenning for sammenkoblingen på akslingen til motor 2.
r_p3=60/2; d_p3=7; t_p3=lm1_S3+lm2_S3; %Radius plate(r_p3), boltehull(d_p3) og
platetykkelse(t_p3).
l_p3=sqrt(2*r_p3^2); %Beregner lengden av tversnittet(l_p3) med pytagoras
%Om z-aksen.
l_h_z3=15; %Lengde fra senter av hull til z-aksen(l_h_z1)
I_Z_p3=((t_p3*l_p3^3)/12)-2*(((t_p3*d_p3^3)/12)+(t_p3*d_p3*l_h_z3^2));
%Treghetsmoment om z-aksen for tversnittet(I_Z_p2)
sigma_p3_z=((M_Z_2*(l_p3/2))/I_Z_p3)*(1-k_faktor_3); %Bøyespenning om z-aksen i
tversnittet(sigma_p1).
```

```
Sikkerhetsfaktor_p3_z=Re_316/sigma_p3_z; %Sikkerhetsfaktor mot flyting med bøyning
om z-aksen.
%Om y-aksen:
I_Y_p3=((l_p3*t_p3^3)/12)-2*(((d_p3*t_p3^3)/12)); %Treghetsmoment om x-aksen for
tversnittet(I_X_p2).
sigma_p3_y=((M_Y_2*(t_p3/2))/I_Y_p3)*(1-k_faktor_3); %Bøyespenning om x-aksen i
tversnittet(sigma_p2_x).
Sikkerhetsfaktor_p3_y=Re_316/sigma_p3_y; %Sikkerhetsfaktor mot flyting med bøyning
om y-aksen.
%Siden momentet om x-aksen er mindre enn om y-aksen, og tversnittet vil se
%identisk ut, vil en ikke beregne denne sikkerhetsfaktoren.

%Dimensjonering mot knekking for 150mm u-brakett. Analyserer knekking om
%2-akser.
%3mm tykkelse. 150mm avstand fra senter til rotor for motor
%2.
t_b1=3; b_b1=60; %Tykkelse og bredde brakett (t_b1, b_b1)
A_b1=t_b1*b_b1; %Spenningsareal brakett(A_b1)
l1_b1=150; %Vinkelrett lengde fra krefter ved aksling to til skrueforbindelse
I_X_b1=(1/12)*((b_b1*t_b1^3)); %Treghetsmoment om x-aksen(I_X_b1);
I_Z_b1=(1/12)*((t_b1*b_b1^3)); %Treghetsmoment om z-aksen(I_Z_b1);
sigma_euler_z_b1=((pi^2)*E_staal*I_Z_b1)/(A_b1*(l1_b1^2)); %Knekkespenning om z-
aksen(sigma_euler_z_b1).
sigma_euler_x_b1=((pi^2)*E_staal*I_X_b1)/(A_b1*(l1_b1^2)); %Knekkespenning om x-
aksen(sigma_euler_x_b1).
%Om x-aksen:
M_X_2=M_X_2+(F_A_2*l1_b1); %Moment om x-aksen. Aksialkraften skaper også et
bøyemoment om x-aksen.
sigma_b_z_b1=(M_X_2*(t_b1/2))/I_X_b1; %Normalspenning i z-retning.
%Om z_aksen:
M_Z_2=M_Z_2+(F_R_X_2*l1_b1); %Moment om z-aksen. Radialkraft i x-retning skaper
også et bøyemoment om z-aksen.
sigma_b_x_b1=(M_Z_2*(b_b1/2))/I_Z_b1;
if sigma_euler_x_b1/sigma_b_z_b1>sigma_euler_z_b1/sigma_b_x_b1
    Sikkerhetsfaktor_knekking_b1=sigma_euler_z_b1/sigma_b_x_b1;
else
    Sikkerhetsfaktor_knekking_b1=sigma_euler_x_b1/sigma_b_z_b1;
end
Sikkerhetsfaktor_knekking_b1;

%Beregning av nødvendig forspenning for 4stk. M6x1-settskruer på aksling til motor
2. Beregne
%rotasjonsvinkel ved montering.
a2y=20; a2i=12; l=a2i/2; %Ytre og indre diameter for akslinger(a2y og a2i) og l,
radius til aksling motor 2, som er avstanden fra friksjonskraften til
rotasjonspunktet.
Sikkerhetsfaktor_aksling2=10; %Sikkerhetsfaktor for settskrueforbindelsen. Setter
den til 10.
Ta2=M_Z_2*Sikkerhetsfaktor_aksling2; %Totalt dreiemoment(Ta2) som
friksjonskreftene må overvinne. Moment om rotasjonsaksen ganget med
sikkerhetsfaktoren.
FRa2=((Ta2/l)/4); %Total friksjonskraft hver settskrue må tilføre(FRa2);
F0_a2=FRa2/f_koeffisient_staal; %Normalkraft/forspenningskraft fra hver
bolt(F0_a2)
A_a2=20.12; lk_a2=((a2y/2)-l); %Spenningsareal til M6-skrue, elastisistetsmodul,
og klemlengde, hvor hele bolten er gjenget.
ks_a2=A_a2*E_staal/lk_a2; %Total stivhet i skrue, dvs. stivhet i gjenget del.
delta_a2=F0_a2/ks_a2; %Deformasjon(delta_a2) i skrue ved ønsket forspenningskraft.
p_a2=1; %Stigning i skruen(p_a2)
```

```
rot_vinkel_a2=round(rad2deg((delta_a2/p_a2)*2*pi),1); %Total rotasjonsvinkel for å
få ønska forspenning og sikkerhetsfaktor på 10(rot_vinkel_a2).

%Dimensjonering mot flyting skrueforbindelse mellom de to u-brakettene.
lm1_S4=3; lm2_S4=3; d_S4=5.35; %Klemlengde materiale 1 og 2, samt
spenningsdiameter for M6-skrue(lm1_S4,lm2_S4 og d_S4)
km1_S4=((4*2.2*d_S4^2)*E_staal)/lm1_S4; %Stivhet materiale 1(km1_S4)
km2_S4=((4*2.2*d_S4^2)*E_staal)/lm2_S4; %Stivhet materiale 2(km2_S4)
km_S4=(km1_S4*km2_S4)/(km1_S4+km2_S4); %Total stivhet materiale(km_S4)
A_S4=(20.12)*4; r_S4=d_S4/2; I_S4=((pi/4)*r_S4^4)*4; J_S4=2*I_S4; %M6-skruer
lsg_S4=lm1_S4+lm2_S4; %Gjenget del av skrue i klem(lsg_S4)
ks_S4=(A_S4*E_staal)/lsg_S4; %Stivhet skrue(ks_S4)
k_faktor_4=ks_S4/(ks_S4+km_S4); %Kraftfordelingsfaktor

sigma_A_S4=(F_A_2)/A_S4; %Aksialspenning i hver skrue(sigma_A_S4)
l2_b1=158; %Bredden av braketten(total bredde av motor, l2_b1)
M_Y_2=M_Y_2+((l2_b1/2)*F_R_X_2); %Total bøyemoment om y-aksen(M_Y_2). Radialkraft
i x-retning skaper også et bøyemoment.
sigma_b_x_S4=(M_Y_2*r_S4)/I_S4; %Bøyespenning i x-retning(sigma_b_x_S4)
sigma_x_S4=sigma_b_x_S4; %Total normalspenning i x-retning(sigma_x_S4)
M_X_2=M_X_2-(F_R_Y_2*l1_b1);  %Total bøyemoment om x-aksen(M_X_2). Radialkraft i
y-retning skaper også et bøyemoment.
sigma_b_y_S4=(M_X_2*r_S4)/I_S4; %Bøyespenning i y-retning(sigma_b_y_S4)
sigma_y_S4=sigma_b_y_S4; %Total normalspenning i y-retning(sigma_y_S4)
sigma_z_S4=sigma_A_S4; %Total normalspenning i z-retning(sigma_z_S4)
tau_xy_S4=(M_Z_2*r_S4)/J_S4+(sqrt(F_R_X_2^2+F_R_Y_2^2)/A_S4); %Skjærspenning pga
torsjon og skjærkrefter(tau_xy_S4).
sigma_e_ytre_S4=(sqrt((sigma_z_S4^2)+(sigma_y_S4^2)+(sigma_x_S4^2)-
((sigma_x_S4*sigma_z_S4)+(sigma_y_S4*sigma_z_S4)+(sigma_x_S4*sigma_y_S4))+(3*(tau_
xy_S4^2)))*k_faktor_4); %Ytre ekvivalentspenning, med kraftfordelingsfaktor.
F_0_S4=Re_88*A_S4*0.5; %Forspenningskraft(F_0_S4)
sigma_S4=F_0_S4/A_S4; %Forspenning(sigma_S4)
tau_S4=0.48*sigma_S4; %Skjærspenning pga forspenning(tau_S4)
sigma_e_forspenning_4=sqrt((sigma_S4^2)+(2*tau_S4^2)); %Ekvivalentspenning pga
forspenning(sigma_e_forspenning_4)
total_sigma_e_S4=(sigma_e_ytre_S4+sigma_e_forspenning_4); %Total
ekvivalentspenning.
Sikkerhetsfaktor_S4=Re_88/total_sigma_e_S4; %Sikkerhetsfaktor mot flyting i
skrueforbindelse.

%Beregning av nødvendig forspenning for 4stk. M4x0.7-settskruer på aksling til
motor 3. Beregne
%rotasjonsvinkel ved montering.
a3y=12; a3i=6; l=a3i/2; %Ytre og indre diameter for akslinger(a3y og a3i) og l,
radius til aksling motor 3, som er avstanden fra friksjonskraften til
rotasjonspunktet.
Sikkerhetsfaktor_aksling3=10; %Sikkerhetsfaktor for settskrueforbindelsen. Setter
den til 10.
M_Z_3=870; %Vrimoment fra motor 3(M_Z_3)
Ta3=M_Z_3*Sikkerhetsfaktor_aksling3; %Totalt dreiemoment(Ta2) som
friksjonskreftene må overvinne. Moment om rotasjonsaksen ganget med
sikkerhetsfaktoren.
FRa3=((Ta3/ l)/4); %Total friksjonskraft hver settskrue må tilføre(FRa3);
F0_a3=FRa3/f_koeffisient_staal; %Normalkraft/forspenningskraft fra hver
bolt(F0_a2)
A_a3=8.78; lk_a3=((a3y/2)-l); %Spenningsareal til M4-skrue, elastisistetsmodul, og
klemlengde, hvor hele bolten er gjenget.
ks_a3=A_a3*E_staal/lk_a3; %Total stivhet i skrue, dvs. stivhet i gjenget del.
delta_a3=F0_a3/ks_a3; %Deformasjon(delta_a3) i skrue ved ønsket forspenningskraft.
```

```
p_a3=0.7; %Stigning i skruen(p_a3)
rot_vinkel_a3=round(rad2deg((delta_a3/p_a3)*2*pi),2); %Total rotasjonsvinkel for å
få ønska forspenning og sikkerhetsfaktor på 10(rot_vinkel_a3).

%Beregning av sikkerhetsfaktor mot gliding i krympeforbindelsen for kulelager til
%motor 3.
rn2i=18.98/2; rn2o=50.0/2; ra2i=6.0/2; ra2o=19.0/2; %Ytre og indre radiuser for
hull og kulelager(rn2i, rn2o, ra2i og ra2o)
delta2=ra2o-rn2i; %Diametralt pressmonn(delta2)
alfa_n2=((rn2i^3)/(E_staal*((rn2o^2)-(rn2i^2))))*((1-
poisson_staal)+((1+poisson_staal)*((rn2o^2)/(rn2i^2)))); %Influenskoeffisient
hull(alfa_n2)
alfa_a2=((ra2i^3)/(E_staal*((ra2o^2)-(ra2i^2))))*((1-
poisson_staal)+((1+poisson_staal)*((ra2o^2)/(ra2i^2)))); %Influenskoeffisient
kulelager(alfa_a2)
kontakttrykk_2=(delta2)/(2*(alfa_a2+alfa_n2)); %Kontakttrykk generert i
krympeforbindelsen(kontakttrykk_2).
F_A_3=-5.71; b2=3; %Aksialkraft ved forbindelsen, samt lengden på hullet(F_Z_3,
b2).
tau2_a=(F_A_3)/(2*pi*rn2i*b2); %Skjærspenning pga aksialkraft(tau2_a).
tau2_t=(M_Z_3)/(2*pi*(rn2i^2)*b2); %Tangentiell skjærspenning pga
vrimoment(tau1_t)
tau2_total=sqrt((tau2_a^2)+(tau2_t^2));
Sikkerhetsfaktor_glidning_2=kontakttrykk_2/tau2_total; %Sikkerhetsfaktor mot
glidning.

%Beregning av krympespenningene for kulelager ved motorholder 3.
sigma2_aa=-kontakttrykk_2; sigma2_na=-kontakttrykk_2; %Krympespenninger i
radialretnning(sigma2_aa, sigma2_na).
sigma2_at=(-kontakttrykk_2*(((ra2o^2)+(ra2i^2))/((ra2o^2)-(ra2i^2))));
%Krympespenning i tangentialretning for kulelager(sigma2_at)
sigma2_nt=(kontakttrykk_2*(((rn2o^2)+(rn2i^2))/((rn2o^2)-(rn2i^2))));
%Krympespenning i tangentialretning for hull(sigma2_at)
if abs(sigma2_nt) >= abs(sigma2_na)
    krymp_max=sigma2_nt;
else
    krymp_max=sigma2_na;
end
Sikkerhetsfaktor_krymp2=Re_316/krymp_max;

%Dimensjonering mot flyting for skrueforbindelse mellom aksling til motor 3
%og 150mm brakett.
lm1_S5=3; lm2_S5=2; d_S5=3.55; %Klemlengde materiale 1 og 2, samt spennngsdiameter
for M4-skrue(lm1_S5, lm2_S5 og d_S5)
km1_S5=((4*2.2*d_S5^2)*E_staal)/lm1_S5; %Stivhet materiale 1.
km2_S5=((4*2.2*d_S5^2)*E_staal)/lm2_S5; %Stivhet materiale 2.
km_S5=(km1_S5*km2_S5)/(km1_S5+km2_S5); %Total stivhet materiale(km_S5)
A_S5=(8.78)*4; r_S5=d_S5/2; I_S5=((pi/4)*r_S5^4)*4; J_S5=2*I_S5; %M4-skruer
lsg_S5=lm1_S5+lm2_S5; %Gjenget del av skrue i klem.
ks_S5=(A_S5*E_staal)/lsg_S5; %Stivhet skrue(ks_S5)
k_faktor_5=ks_S5/(ks_S5+km_S5); %Kraftfordelingsfaktor.

F_R_X_3=12.7; F_R_Y_3=19.1; M_Y_3=-400; M_X_3=640; %Krefter og momenter i x- og y-
retning.
sigma_z_S5=(F_A_3)/A_S5; %Normalspenning i z-retning pga aksialspenning.
sigma_x_S5=(M_Y_3*r_S5)/I_S5; %Normalspenning i x-retning pga bøyepspenning.
sigma_y_S5=(M_X_3*r_S5)/I_S5;  %Normalspenning i y-retning pga bøyepspenning.
tau_xy_S5=(M_Z_3*r_S5)/J_S5+(sqrt(F_R_X_3^2+F_R_Y_3^2)/A_S5); %Skjærspenning.
```

```
sigma_e_ytre_S5=(sqrt((sigma_z_S5^2)+(sigma_y_S5^2)+(sigma_x_S5^2)-
((sigma_x_S5*sigma_z_S5)+(sigma_y_S5*sigma_z_S5)+(sigma_x_S5*sigma_y_S5))+(3*(tau_
xy_S5^2)))*k_faktor_5); %Ytre ekvivalentspenning.
F_0_S5=Re_88*A_S5*0.5; %Forspenningskraft
sigma_S5=F_0_S5/A_S5; %Forspenning
tau_S5=0.48*sigma_S5; %Skjærspenning pga forspenning.
sigma_e_forspenning_5=sqrt((sigma_S5^2)+(2*tau_S5^2)); %Ekvivalentspenning pga
forspenning.
total_sigma_e_S5=(sigma_e_ytre_S5+sigma_e_forspenning_5); %Total
ekvivalentspenning.
Sikkerhetsfaktor_S5=Re_88/total_sigma_e_S5; %Sikkerhetsfaktor mot flyting.

%Kritisk bøyespenning for sammenkoblingen på akslingen til motor 3.
r_p4=50/2; d_p4=4.5; t_p4=lm1_S5+lm2_S5; %Radius plate(r_p4), boltehull(d_p4) og
platetykkelse(t_p4).
l_p4=sqrt(2*r_p4^2); %Beregner lengden av tversnittet(l_p4) med pytagoras
%Om z-aksen.
l_h_z4=10; %Lengde fra senter av hull til z-aksen(l_h_z4) langs enten x- eller y-
aksen.
I_Z_p4=((t_p4*l_p4^3)/12)-2*(((t_p4*d_p4^3)/12)+(t_p4*d_p4*l_h_z4^2));
%Treghetsmoment om z-aksen for tversnittet(I_Z_p4)
sigma_p4_z=((M_Z_3*(l_p4/2))/I_Z_p4)*(1-k_faktor_5);%Bøyespenning om z-aksen i
tversnittet(sigma_p4_z).
Sikkerhetsfaktor_p4_z=Re_316/sigma_p4_z; %Sikkerhetsfaktor mot flyting med bøyning
om z-aksen.
%Om x-aksen:
I_X_p4=((l_p4*t_p4^3)/12)-2*(((d_p4*t_p4^3)/12)); %Treghetsmoment om x-aksen for
tversnittet(I_X_p4).
sigma_p4_x=((M_X_3*(t_p4/2))/I_X_p4)*(1-k_faktor_5); %Bøyespenning om x-aksen i
tversnittet(sigma_p2'4_x).
Sikkerhetsfaktor_p4_x=Re_316/sigma_p4_x; %Sikkerhetsfaktor mot flyting med bøyning
om y'x-aksen.
%Siden momentet om x'y-aksen er mindre enn om y'x-aksen, og tversnittet vil se
%identisk ut, vil en ikke beregne denne sikkerhetsfaktoren.

%Dimensjonering mot flyting i skruen som holder holder kulelageret ved
%motor 3 på plass:
lm_k2=6; d_k2=5.35; %lengde på klemsone og diameter for skrue(lm_k2, d_k2)
km_k2=((2.2*d_k2^2)*E_staal)/lm_k2; %Stivhet materiale(km_k2)
A_k2=20.12; r_k2=d_k2/2; I_k2=((pi/4)*r_k2^4)*4; J_k2=2*I_k2;% %M6-skrue. Areal,
radius og treghetsmomenter(A_k2, r_k2, I_k2, J_k2)
lsg_k2=lm_k2; %Gjenget del av skrue i klem(lsg_k2)
ks_k2=(A_k2*E_staal)/lsg_k2; %Stivhet skrue(ks_k2)
k_faktor_k2=ks_k2/(ks_k2+km_k2); %Kraftfordelingsfaktor(k_faktor_k2)

sigma_A_k2=(F_A_3)/A_k2; %Aksialspenning(sigma_A_k2)
sigma_b_x_k2=(M_Y_3*r_k2)/I_k2; %Bøyespenning i x-retning(sigma_b_x_k2)
sigma_x_k2=sigma_b_x_k2; %Total spenning i x-retning(sigma_x_k2).
sigma_b_y_k2=(M_X_3*r_k2)/I_k2;%Bøyespenning i y retning(sigma_b_y_k2)
sigma_y_k2=sigma_b_y_k2; %Total spenning i y-retning(sigma_y_k2)
sigma_z_k2=sigma_A_k2; %Total spenning i z-retning(sigma_z_k2)
tau_xy_k2=(M_Z_3*r_k2)/J_k2+(sqrt(F_R_X_3^2+F_R_Y_3^2)/A_k2); %Total
skjærspenning(tau_xy_k2)
sigma_e_ytre_k2=(sqrt((sigma_z_k2^2)+(sigma_y_k2^2)+(sigma_x_k2^2)-
((sigma_x_k2*sigma_z_k2)+(sigma_y_k2*sigma_z_k2)+(sigma_x_k2*sigma_y_k2))+(3*(tau_
xy_k2^2)))*k_faktor_k2); %Ytre tilført ekvivalentspenning(sigma_e_ytre_k2)
Re_k2=640; %Flytegrense(Re_k2)
F_0_k2=Re_k2*A_k2*0.5; %Forspenningskraft(F_0_k2)
sigma_k2=F_0_k2/A_k2; %Forspenning(sigma_k2)
```

```
tau_k2=0.48*sigma_k2; %Skjærspenning av forspenning(tau_k2)
sigma_e_forspenning_k2=sqrt((sigma_k2^2)+(2*tau_k2^2)); %Ekvivalentspenning av
forspenning.
total_sigma_e_k2=(sigma_e_ytre_k2+sigma_e_forspenning_k2); %Total
ekvivalentspenning(total_sigma_e_k2)
Sikkerhetsfaktor_k2=Re_k2/total_sigma_e_k2; %Sikkerhetsfaktor

%Dimensjonering mot flyting for skrueforbindelse mellom motorholder 3 og
%motorholder 4.
lm1_S6=2; lm2_S6=2; d_S6=3.55; %Klemlengde materiale 1 og 2, samt spennngsdiameter
for M4-skrue(lm1_S6, lm2_S6 og d_S6)
km1_S6=((4*2.2*d_S6^2)*E_staal)/lm1_S6; %Stivhet materiale 1.
km2_S6=((4*2.2*d_S6^2)*E_staal)/lm2_S6; %Stivhet materiale 2.
km_S6=(km1_S6*km2_S6)/(km1_S6+km2_S6); %Total stivhet materiale(km_S6)
A_S6=(8.78)*4; r_S6=d_S6/2; I_S6=((pi/4)*r_S6^4)*4; J_S6=2*I_S6; %M4-skruer
lsg_S6=lm1_S6+lm2_S6; %Gjenget del av skrue i klem.
ks_S6=(A_S6*E_staal)/lsg_S6; %Stivhet skrue(ks_S6)
k_faktor_6=ks_S6/(ks_S6+km_S6); %Kraftfordelingsfaktor.

sigma_y_S6=(F_R_Y_3)/A_S6; %Normalspenning i y-retning pga aksialspenning.
sigma_x_S6=(M_Z_3*r_S6)/I_S6; %Normalspenning i x-retning pga bøyepspenning.
sigma_z_S6=(M_X_3*r_S5)/I_S6;  %Normalspenning i z-retning pga bøyepspenning.
tau_xz_S6=(M_Y_3*r_S6)/J_S6+(sqrt(F_R_X_3^2+F_A_3^2)/A_S6); %Skjærspenning i xz-
planet.
sigma_e_ytre_S6=(sqrt((sigma_z_S6^2)+(sigma_y_S6^2)+(sigma_x_S6^2)-
((sigma_x_S6*sigma_z_S6)+(sigma_y_S6*sigma_z_S6)+(sigma_x_S6*sigma_y_S6))+(3*(tau_
xz_S6^2)))*k_faktor_6); %Ytre ekvivalentspenning.
F_0_S6=Re_88*A_S6*0.5; %Forspenningskraft
sigma_S6=F_0_S6/A_S6; %Forspenning
tau_S6=0.48*sigma_S6; %Skjærspenning pga forspenning.
sigma_e_forspenning_6=sqrt((sigma_S6^2)+(2*tau_S6^2)); %Ekvivalentspenning pga
forspenning.
total_sigma_e_S6=(sigma_e_ytre_S6+sigma_e_forspenning_6); %Total
ekvivalentspenning.
Sikkerhetsfaktor_S6=Re_88/total_sigma_e_S6; %Sikkerhetsfaktor mot flyting.

%Dimensjonering mot flyting for skrueforbindelse mellom aksling til motor 4
%og 120mm u-brakett.
lm1_S7=2; lm2_S7=2; d_S7=3.55; %Klemlengde materiale 1 og 2, samt spennngsdiameter
for M4-skrue(lm1_S7, lm2_S7 og d_S7)
km1_S7=((4*2.2*d_S7^2)*E_staal)/lm1_S7; %Stivhet materiale 1.
km2_S7=((4*2.2*d_S7^2)*E_staal)/lm2_S7; %Stivhet materiale 2.
km_S7=(km1_S7*km2_S7)/(km1_S7+km2_S7); %Total stivhet materiale(km_S7)
A_S7=(8.78)*4; r_S7=d_S7/2; I_S7=((pi/4)*r_S7^4)*4; J_S7=2*I_S7; %M4-skruer
lsg_S7=lm1_S7+lm2_S7; %Gjenget del av skrue i klem.
ks_S7=(A_S5*E_staal)/lsg_S7; %Stivhet skrue(ks_S7)
k_faktor_7=ks_S7/(ks_S7+km_S7); %Kraftfordelingsfaktor.

M_Z_4=-60; F_A_4=-16.82; F_R_X_4=3.88; F_R_Y_4=-5.44; M_Y_4=590; M_X_4=730;
%Krefter og momenter i x-, y- og z-retning.
sigma_z_S7=(F_A_4)/A_S7; %Normalspenning i z-retning pga aksialspenning.
sigma_x_S7=(M_Y_4*r_S7)/I_S7; %Normalspenning i x-retning pga bøyepspenning.
sigma_y_S7=(M_X_4*r_S7)/I_S7;  %Normalspenning i y-retning pga bøyepspenning.
tau_xy_S7=(M_Z_4*r_S7)/J_S7+(sqrt(F_R_X_4^2+F_R_Y_4^2)/A_S7); %Skjærspenning.
sigma_e_ytre_S7=(sqrt((sigma_z_S7^2)+(sigma_y_S7^2)+(sigma_x_S7^2)-
((sigma_x_S7*sigma_z_S7)+(sigma_y_S7*sigma_z_S7)+(sigma_x_S7*sigma_y_S7))+(3*(tau_
xy_S7^2)))*k_faktor_7); %Ytre ekvivalentspenning.
F_0_S7=Re_88*A_S7*0.5; %Forspenningskraft
sigma_S7=F_0_S7/A_S7; %Forspenning
```

```
tau_S7=0.48*sigma_S7; %Skjærspenning pga forspenning.
sigma_e_forspenning_7=sqrt((sigma_S7^2)+(2*tau_S7^2)); %Ekvivalentspenning pga
forspenning.
total_sigma_e_S7=(sigma_e_ytre_S7+sigma_e_forspenning_7); %Total
ekvivalentspenning.
Sikkerhetsfaktor_S7=Re_88/total_sigma_e_S7; %Sikkerhetsfaktor mot flyting.

%Dimensjonering mot flyting for skrueforbindelse mellom aksling til motor 5
%og 120mm u-brakett.
lm1_S8=2; lm2_S8=2; d_S8=3.55; %Klemlengde materiale 1 og 2, samt spennngsdiameter
for M4-skrue(lm1_S8, lm2_S8 og d_S8)
km1_S8=((4*2.2*d_S8^2)*E_staal)/lm1_S8; %Stivhet materiale 1.
km2_S8=((4*2.2*d_S8^2)*E_staal)/lm2_S8; %Stivhet materiale 2.
km_S8=(km1_S8*km2_S8)/(km1_S8+km2_S8); %Total stivhet materiale(km_S8)
A_S8=(8.78)*4; r_S8=d_S8/2; I_S8=((pi/4)*r_S8^4)*4; J_S8=2*I_S8; %M4-skruer
lsg_S8=lm1_S8+lm2_S8; %Gjenget del av skrue i klem.
ks_S8=(A_S8*E_staal)/lsg_S8; %Stivhet skrue(ks_S8)
k_faktor_8=ks_S8/(ks_S8+km_S8); %Kraftfordelingsfaktor.

M_Z_5=100; F_A_5=-1.82; F_R_X_5=-5.65; F_R_Y_5=13.2; M_Y_5=60; M_X_5=0; %Krefter
og momenter i x-, y- og z-retning.
sigma_z_S8=(F_A_5)/A_S8; %Normalspenning i z-retning pga aksialspenning.
sigma_x_S8=(M_Y_5*r_S8)/I_S8; %Normalspenning i x-retning pga bøyepspenning.
sigma_y_S8=(M_X_5*r_S8)/I_S8;  %Normalspenning i y-retning pga bøyepspenning.
tau_xy_S8=(M_Z_5*r_S8)/J_S8+(sqrt(F_R_X_5^2+F_R_Y_5^2)/A_S8); %Skjærspenning.
sigma_e_ytre_S8=(sqrt((sigma_z_S8^2)+(sigma_y_S8^2)+(sigma_x_S8^2)-
((sigma_x_S8*sigma_z_S8)+(sigma_y_S8*sigma_z_S8)+(sigma_x_S8*sigma_y_S8))+(3*(tau_
xy_S8^2)))*k_faktor_8); %Ytre ekvivalentspenning.
F_0_S8=Re_88*A_S8*0.5; %Forspenningskraft
sigma_S8=F_0_S8/A_S8; %Forspenning
tau_S8=0.48*sigma_S8; %Skjærspenning pga forspenning.
```

Vedlegg J-Styreprogram

```
int stoppPin=27;

int LEDstopp=23;

int startPin=29;

int LEDstart=25;

int ventilPin=31;

int vaskepin=33;

int LEDvaske=35;

int startny;

int startgammel=1;

int stoppny;
```

```
int stoppgammel=1;

int vaskeny;

int vaskegammel=1;

boolean stopp_aktivert=false;

boolean forste_gang=true;


long posisjoner[6]; //Samler posisjonene i en array.

#include <AccelStepper.h>

#include <MultiStepper.h>


// Definerer stegmotorene og pinsene som de skal styres fra. Følgende syntaks:
stegmotor(modus, puls, retning);


AccelStepper stepper1(1, 2, 3);

AccelStepper stepper2(1, 4, 5);

AccelStepper stepper3(1, 6, 7);

AccelStepper stepper4(1, 8, 9);

AccelStepper stepper5(1, 10, 11);

AccelStepper stepper6(1, 12, 13);


MultiStepper steppers; //Samler stegmotorene i en gruppe ved hjelp av MultiStepper.


void setup()

{

  pinMode(stoppPin, INPUT);
```

```
pinMode(startPin, INPUT);
pinMode(vaskepin, OUTPUT);
pinMode(LEDstopp, OUTPUT);
pinMode(LEDstart, OUTPUT);
pinMode(ventilPin, OUTPUT);

digitalWrite(stoppPin, HIGH);
digitalWrite(startPin, HIGH);
digitalWrite(vaskepin, HIGH)
digitalWrite(LEDstopp, LOW);
digitalWrite(LEDstart, HIGH);
digitalWrite(ventilPin, LOW);

Serial.begin(9600);

stepper1.setMaxSpeed(1600);
stepper1.setAcceleration(800);

stepper2.setMaxSpeed(2350);
stepper2.setAcceleration(1175);

stepper3.setMaxSpeed(20000);
stepper3.setAcceleration(10000);
```

```
  stepper4.setMaxSpeed(800000);
  stepper4.setAcceleration(400000);

  stepper5.setMaxSpeed(800000);
  stepper5.setAcceleration(800000);

  stepper6.setMaxSpeed(3200);
  stepper6.setAcceleration(6400);

  steppers.addStepper(stepper1);
  steppers.addStepper(stepper2);
  steppers.addStepper(stepper3);
  steppers.addStepper(stepper4);
  steppers.addStepper(stepper5);
  steppers.addStepper(stepper6);
}

void loop()
{
  startny=digitalRead(startPin);
  stoppny=digitalRead(stoppPin);
  if (stoppny==0 && stoppgammel==1){
    stopp_aktivert=true;
  }
```

```
if (stopp_aktivert==true){
  digitalWrite(ventilPin, LOW);
  digitalWrite(LEDvaske, LOW);
  hvileposisjon();
  tilbake_til_start();
  forste_gang=true;
  stopp_aktivert=false;
}
if (vaskeny==0 && vaskegammel==1){
  (if stopp_aktivert==false){
  digitalWrite(LEDvaske, HIGH);
  digitalWrite(ventilPin, HIGH);
  }
}
Serial.println(stopp_aktivert);
if (startny==0 && startgammel==1){
    digitalWrite(LEDstart, LOW);
    digitalWrite(LEDstopp, HIGH);
    if (forste_gang==true){
      til_hvileposisjon();
    }
    vaffel_venstre_naermest();
    aapne_ventil();
    vaffel_venstre_lengst_borte();
```

```
      aapne_ventil();
      vaffel_hoyre_naermest();
      aapne_ventil();
      vaffel_hoyre_lengst_borte();
      aapne_ventil();
      til_hvileposisjon();
      forste_gang=false;
  }
}
void til_hvileposisjon()
{
  if (stopp_aktivert==false){
    posisjoner[0]=0;
    posisjoner[1]=0;
    posisjoner[2]=-40000;
    posisjoner[3]=0;
    posisjoner[4]=0;
    posisjoner[5]=0;

    steppers.moveTo(posisjoner);
    steppers.runSpeedToPosition();
    delay(5000);

    digitalWrite(LEDstart, HIGH);
```

```
    digitalWrite(LEDstopp, LOW);
  }
}
void vaffel_venstre_naermest()
{
  if (stopp_aktivert==false){
    posisjoner[0]=1440;
    posisjoner[1]=2141;
    posisjoner[2]=-13333;
    posisjoner[3]=44000;
    posisjoner[4]=889;
    posisjoner[5]=0;

    steppers.moveTo(posisjoner);
    steppers.runSpeedToPosition();
    delay(5000);
  }
}
void vaffel_venstre_lengst_borte()
{
  if (stopp_aktivert==false){
    posisjoner[0]=1511;
    posisjoner[1]=2768;
    posisjoner[2]=-38222+5000;
```

```
    posisjoner[3]=41778;
    posisjoner[4]=-889;
    posisjoner[5]=0;

    steppers.moveTo(posisjoner);
    steppers.runSpeedToPosition();
    delay(3000);
  }
}
void vaffel_hoyre_naermest()
{
  if (stopp_aktivert==false){
    posisjoner[0]=-1440;
    posisjoner[1]=2141;
    posisjoner[2]=-13333;
    posisjoner[3]=-44000;
    posisjoner[4]=889;
    posisjoner[5]=0;
    steppers.moveTo(posisjoner);
    steppers.runSpeedToPosition();
    delay(3000);
  }
}
void vaffel_hoyre_lengst_borte()
```

```
{
  if (stopp_aktivert==false){
    posisjoner[0]=-1511;
    posisjoner[1]=2768;
    posisjoner[2]=-38222+5000;
    posisjoner[3]=-41778;
    posisjoner[4]=-889;
    posisjoner[5]=0;

    steppers.moveTo(posisjoner);
    steppers.runSpeedToPosition();
    delay(3000);
  }
}
void tilbake_til_start()
{
    posisjoner[0]=0;
    posisjoner[1]=0;
    posisjoner[2]=0;
    posisjoner[3]=0;
    posisjoner[4]=0;
    posisjoner[5]=0;
    digitalWrite(LEDstopp, HIGH);
```

```
    steppers.moveTo(posisjoner);
    steppers.runSpeedToPosition();
    delay(3000);

    digitalWrite(LEDstart, HIGH);
    digitalWrite(LEDstopp, LOW);
    stopp_aktivert=false;

}
void aapne_ventil()
{
  digitalWrite(ventilPin, HIGH);
  delay(700);
  digitalWrite(ventilPin, LOW);
  delay(6000);
}
```

Vedlegg K-Kode for pumpeberegninger

```
%%Narve Gilje Nordås
%%Beregninger av trykk i pumpesystem.
%%Med 0.5-bar-pumpe. Kjølevæske-pumpe. Åpen ventil.
P1=101300; v1=0; h1=0.25; p_rore=1002; g=9.81; P_pumpe=50000; D1=0.26;
h2=0; D2=0.021; L2=0.45; fh_2=0.13; u_rore=2; A2=((D2/2)^2)*pi;
v2=(fh_2*p_rore*g*(D2^2))/(32*u_rore*L2);
P2=P1+P_pumpe+(h1*g*p_rore)-(0.5*v2*p_rore)-(fh_2*g*p_rore);
Q=((A2*pi*v2));

h3=0.45; D3=0.013; L3=0.9; A3=((D3/2)^2)*pi; v3=Q/A3;
Re_3=(p_rore*D3*v3)/u_rore;
f3=64/Re_3;
fh_3=(f3*L3*(v3^2))/(2*D3*g);
P3=P2+(0.5*(v2^2)*p_rore)-(h3*g*p_rore)-(0.5*(v3^2)*p_rore)-(fh_3*g*p_rore);

h4=0.25; D4=0.013; L4=0.3; A4=((D4/2)^2)*pi; v4=Q/A4;
Re_4=(p_rore*D4*v4)/u_rore;
```

```
f4=64/Re_4;
fh_4=(f4*L4*(v4^2))/(2*D4*g);
P4=P3+(0.5*(v3^2)*p_rore)+((h3-h4)*g*p_rore)-(0.5*(v4^2)*p_rore)-(fh_4*g*p_rore);

h5=0.25-0.052; D5=0.008; L5=0.052; A5=((D5/2)^2)*pi; v5=Q/A5;
Re_5=(p_rore*D5*v5)/u_rore;
f5=64/Re_5;
fh_5=(f5*L5*(v5^2))/(2*D5*g);
P5=P4+(0.5*(v4^2)*p_rore)+((h4-h5)*g*p_rore)-(0.5*(v5^2)*p_rore)-(fh_5*g*p_rore);

%%Beregninger av trykk i pumpesystem.
%%Med 3.8-bar-pumpe . Trykkvannspumpe. Lukket ventil.
P1=101300; h1=0.25; p_rore=1002; g=9.81; P_pumpe=380000;
h2=0;
P2=P1+P_pumpe+(h1*g*p_rore);

h3=0.45;
P3=P2+((h2-h3)*g*p_rore);

h4=0.25;
P4=P3+((h3-h4)*g*p_rore);

h5=0.25-0.052;
P5=P4+((h4-h5)*g*p_rore);

%%Beregninger av trykk i pumpesystem.
%%Med 3.8-bar-pumpe. Trykkvannspumpe. Åpen ventil.
P1=101300; v1=0; h1=0.25; p_rore=1002; g=9.81; P_pumpe=380000; D1=0.26;
h2=0; D2=0.021; L2=0.45; fh_2=0.44; u_rore=2; A2=((D2/2)^2)*pi;
v2=(fh_2*p_rore*g*(D2^2))/(32*u_rore*L2);
P2=P1+P_pumpe+(h1*g*p_rore)-(0.5*v2*p_rore)-(fh_2*g*p_rore);
Q=((A2*pi*v2));

h3=0.45; D3=0.013; L3=0.9; A3=((D3/2)^2)*pi; v3=Q/A3;
Re_3=(p_rore*D3*v3)/u_rore;
f3=64/Re_3;
fh_3=(f3*L3*(v3^2))/(2*D3*g);
P3=P2+(0.5*(v2^2)*p_rore)-(h3*g*p_rore)-(0.5*(v3^2)*p_rore)-(fh_3*g*p_rore);

h4=0.25; D4=0.013; L4=0.3; A4=((D4/2)^2)*pi; v4=Q/A4;
Re_4=(p_rore*D4*v4)/u_rore;
f4=64/Re_4;
fh_4=(f4*L4*(v4^2))/(2*D4*g);
P4=P3+(0.5*(v3^2)*p_rore)+((h3-h4)*g*p_rore)-(0.5*(v4^2)*p_rore)-(fh_4*g*p_rore);

h5=0.25-0.052; D5=0.008; L5=0.052; A5=((D5/2)^2)*pi; v5=Q/A5;
Re_5=(p_rore*D5*v5)/u_rore;
f5=64/Re_5;
fh_5=(f5*L5*(v5^2))/(2*D5*g);
P5=P4+(0.5*(v4^2)*p_rore)+((h4-h5)*g*p_rore)-(0.5*(v5^2)*p_rore)-(fh_5*g*p_rore);
```