Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for ingeniørvitenskap
Institutt for havromsoperasjoner og byggteknikk

Herman Høye
Robin H. Vikra
Øyvind Stavdal

VERKTØY FOR DESIGNOPTIMALISERING AV EN STEMPELKOMPRESSOR

Bacheloroppgave i Produkt og Systemdesign, Maskiningeniør
Veileder: Lars Petter Bryne
Medveileder: Vladimir Krivopolianskii
Mai 2023

NTNU
Kunnskap for en bedre verden

Herman Høye
Robin H. Vikra
Øyvind Stavdal

VERKTØY FOR DESIGNOPTIMALISERING AV EN STEMPELKOMPRESSOR

Bacheloroppgave i Produkt og Systemdesign, Maskiningeniør
Veileder: Lars Petter Bryne
Medveileder: Vladimir Krivopolianskii
Mai 2023

Norges teknisk-naturvitenskapelige universitet
Fakultet for ingeniørvitenskap
Institutt for havromsoperasjoner og byggteknikk

NTNU
Kunnskap for en bedre verden

FORORD

Denne oppgaven viser de ferdigheter og kunnskaper studentene har ervervet gjennom de tre årene av bachelorutdannelsen for maskiningeniør ved NTNU i Ålesund.
Vi vil først og fremst takke Vladimir Krivopolianskii for hans utrettelige veiledning og betydelige støtte under hele prosessen. Hans kompetanse og erfaring var avgjørende for at vi kunne nå våre mål. Vi vil også uttrykke vår takknemlighet til Asbjørn Klokk og Vilmar Æsøy for deres kompetanse og tekniske innsikt.

Programmet som skal utvikles krever at studentene forstår bedriftens krav og behov. Deretter må de vurdere ulike program og metoder for å løse oppgaven på en effektiv måte. Det er viktig at studentene tenker kritisk og evner å ta beslutning for å optimalisere resultatene. Når grunnmuren er på plass kan studentene begynne med en design- og implementeringfase.

Studentene vil jobbe med koding, testing og debugging av programmet for å sikre at det oppfyller alle krav til kvalitet og funksjonalitet. Dette vil gi studentene nyttig programmeringserfaring, og bidra til Sperres tekniske ressurser.

Ved ferdigstillelse av programmet er det viktig at studentene dokumenterer sitt arbeid og sørger for et brukervennlig program. Dette bidrar til at programmet er enkelt å bruke og lett vedlikeholdbart. Dokumentasjonen vil også gi en kontinuitet for videre utvikling av programmet.



Herman Høye

Student ved NTNU Ålesund,
B.Sc.
Ålesund, 22.05.23

Robin Holgersen Vikra

Student ved NTNU Ålesund,
B.Sc.
Ålesund, 22.05.23

Øyvind Stavdal

Student ved NTNU Ålesund,
B.Sc.
Ålesund, 22.05.23

OPPGAVETEKST:
VERKTØY FOR DESIGNOPIMALISERING AV EN STEMPELKOMPRESSOR

Sperre Air Power AS er en ledende aktør innen kompressorer til skipsindustrien. Så mye som 20% av alle skip i verden har en kompressor fra Sperre. Sperre Air Power AS ble grunnlagt i 1938 og omsatte i 2021 for 374,2 millioner. I tillegg til å lage kompressorer så leverer også Sperre alt av reservedeler for alle sine produkter i opptil 30 år. Sperre har også en lovnad om å levere sine reservedeler hvor helst i verden i løpet av 48 timer. Fabrikken på Ellingsøy er delt opp i to deler der administrasjon og produksjon er oppdelt og har 95 ansatte. Sperre har mange distributører, selgere og logistikkpunkt rundt om i verden. Under dette prosjektet er det av ønske fra Sperre å utvikle et program som er i stand til å simulere vibrasjon i kompressorens ben og bidra til en designoptimalisering av stempelkompressoren. Dette vil kunne bidra til å forbedre kompressorene og vil kunne brukes som et verktøy når nye modeller skal utvikles. Oppgavens hovedaspekt er å utvikle et verktøy som kan hjelpe Sperre med å redusere vibrasjon som forplanter seg i innfesting av kompressoren. Oppgaven vil innebære å utvikle et fundament for Sperre som de skal ha mulighet til å videreutvikle gjennom årene som kommer. Alle kandidater må undertegne en taushetserklæring for å få innsyn i Sperre sin data.

Kandidatene av denne oppgaven skal i løpet av prosjektet gjennomføre følgende:

- Sette seg inn i system og program som Sperre ønsker programmet utviklet i.
- Sette seg inn i standarder, klassifiseringer og regelverk (DNV) for vibrasjon og lydforurensninger.
- Utvikle en matematisk modell som kan simulere dynamiske krefter som påvirker systemet under drift.
- Systemet skal realiseres i form av et program som kan orkestrere simuleringer i Siemens NX, 20Sim og andre kommersielle programvarer som vil brukes i oppgaven.
- Programvaren skal kunne håndtere material- og geometriendring.

Veileder ved NTNU er Lars Petter Bryne, og kontaktperson/ faglig veileder ved Sperre Air Power AS er Vladimir Krivopolianskii.
Besvarelsen leveres som en teknisk rapport, med en innledning og abstrakt både på norsk og engelsk. Kandidatene tar for seg retningslinjer for en teknisk rapport under utarbeidelsen av besvarelsen. Kandidatene skal legge vekt på at tekstens oppbygning og innhold skal kunne forstås og gjenskapes av en utenforstående. Det skal også legges vekt på at referanser til tilhørende grafer, tabeller og bilder skal korrespondere med innholdet i besvarelsen.

ABSTRAKT

Prosjektet fokuserer på utviklingen av et program som knytter sammen flere digitale operasjoner til én. Programmet kan endre material for kompressorens stempler og/eller endre geometri av en komponent. Det har her vært stort fokus på brukervennlighet i brukergrensesnittet for å sikre at programmet skal kunne brukes uten omfattende opplæring. Omfattende arbeid er lagt i programmets kode, slik at den er både sikker og effektiv å bruke. Programmet skal lagre simuleringen i en dataramme, hvor brukeren selv har tilgang til resultatene.

Oppgaven starter med å gi en kort bakgrunnsbeskrivelse av kompressorteknologi og vibrasjonsanalyse. Videre inndeles rapporten i seksjoner: Metode, Teori, Utvikling, Resultat, Diskusjon, og Konklusjon. Gjennom disse seksjonene blir hele prosessforløpet til programmet gjennomgått. Det vil drøftes om de ulike løsningene som ble brukt, og hvorvidt bruk av de ulike programmene var hensiktsmessig. Som vedlegg vil det følge instruksjonsmanual for sluttbruker. Endelig levering vil inneholde rapport og mappe med program, skript og tilhørende plugins. Rapporten inneholder alle tilstrekkelige vedlegg for rekreasjon av programmet.

Resultatene gir innblikk i hvordan systemet fungerer under normal drift og ved geometriendring av kontravekt. Resultatet vil også gi innblikk i hvordan materialendring fra aluminium til stål påvirker vibrasjon i kompressoren.

ABSTRACT IN ENGLISH

The project focuses on the development of a program that integrates multiple digital operations into one. The program can modify materials for compressor pistons and/or change the geometry of a component. There has been a strong emphasis on user-friendliness in the user interface to ensure that the program can be used without extensive training. Extensive work has been done on the program's code to make it both secure and efficient to use. The program will store the simulation in a data frame, where the user will have access to the results.

The task begins with a brief background description of compressor technology and vibration analysis. The report is then divided into sections: Method, Theory, Development, Results, Discussion, and Conclusion. These sections cover the entire process flow of the program. The different solutions used will be discussed, as well as the appropriateness of using the various programs. The appendix will include an instruction manual for the end user. The final delivery will consist of the report and a folder containing the program, scripts, and associated plugins. The report includes all necessary attachments for recreating the program.

The results provide insights into how the system operates during normal operation and when changing the geometry of the counterweight. The results will also provide insight into how changing the material from aluminum to steel affects vibration in the compressor.

FORKORTELSER

.CSV Comma-Seperated Values.

API Application Programming Interface.

CMM Coordinate Measuring Machine.

DNV Det Norske Veritas.

GUI Graphical User Interface.

MTF Modulated Transformer.

MVP Minimum Viable Product.

NX Siemens NX.

ODE Ordinary Differentiac Equations.

RPM Revolutions Per Minute.

ShA Shore A.

SoS System of systems.

TF Transformator.

Innholdsfortegnelse

Liste med figurer

Liste med tabeller

1. INTRODUKSJON

Formålet med denne oppgaven er å utvikle et verktøy som knytter sammen ulike simuleringsprosesser gjennom flere programmer for å kunne hjelpe å verifisere en designendring i en XA-150 kompressor. Iterasjonene som her simuleres, vil være vibrasjon som følge av endring på kompressorens indre komponenter.

Grunnlaget for bedriftens interesse rundt vibrasjonsanalyse er at alle kompressorer må typegodkjennes og falle innenfor DNVs retningslinjer for vibrasjon. Dette er derfor en essensiell del av videreutvikling eller fremstilling av nye kompressormodeller. Sperre innehar idag de nødvendige verktøyene for å utføre disse iterasjonene, men mangler bindeledd mellom prosessene. Programmet som utvikles skal kjøre og flytte verdier mellom disse programmene, og presentere resultatet for en bruker. Hovedpunktene til programmet skal være å simulere vibrasjon som følge av endring i veivgeometri, og massetetthet i kompressorens stempler. For dette prosjektet skal det også utvikles en komplett matematisk modell som vil bruke verdier fra prosjektet til å beregne vibrasjonen i kompressorens demperelementer. I ønske av bedriften er arbeidet utført med tanke om at prosjektet skal kunne videreutvikles av bedriften i fremtiden.

Oppgaven er oppbygd som en standard teknisk rapport, og i oppgaven blir det gjennomgått retningslinjer, metode, teori, utvikling, resultater, diskusjon og konklusjon. All kode som er skrevet for prosjektet blir inkludert som vedlegg, samt alle filer som trengs for å kjøre prosjektets ferdigutviklede program. I vedlegg vil det også følge en brukermanual, samt tidlige utkast av prosjektets programmer og simuleringer.

Sperre Air Power AS er en ledende leverandør av kompressorteknologi til skipsbransjen. 20% av alle seilende skip i hele verden har en kompressor fra Sperre. Selskapet ble grunnlagt i 1938 og hadde en omsetning på 374,2 millioner kroner i 2021. Fabrikken på Ellingsøy er delt inn i to områder, administrasjon og produksjon. I Norge har Sperre 95 ansatte, i tillegg til flere distributører, selgere og logistikkpunkter verden over.

1.1. Introduction in english

The purpose of this task is to develop a program that connects various simulation processes across multiple programs to simulate vibrations in an XA-150 compressor. The iterations being simulated here will be vibrations resulting from changes in the compressor's internal components.

The basis for the company's interest in vibration analysis is that all compressors must be approved and comply with DNV's vibration guidelines. This is therefore an essential part of further development or production of new compressor models. Sperre currently has the necessary tools to perform these iterations but lacks the integration between processes. The developed program shall run and transfer values between these programs and present the results to a user. The main points of the program will be to simulate vibrations resulting from changes in crankshaft geometry and the mass density of the compressor's pistons. For this project, a complete mathematical model will also be developed, which will use values from the project to calculate the vibration in the compressor's damping elements. The company's desire is that the work is carried out with the intention that the project can be further developed by the company at the end of this collaboration.

The task is structured as a standard technical report, covering guidelines, methodology, theory, development, results, discussion, and conclusion. All code written for the project will be included as an appendix, along with all files required to run the project's fully developed program. The appendix will also include a user manual, as well as early drafts of the project's programs and simulations.

Sperre Air Power AS is a leading supplier of compressor technology to the maritime industry. 20% of all sailing vessels worldwide have a compressor from Sperre. The company was founded in 1938 and had a turnover of 374.2 million NOK in 2021. The factory in Ellingsøy is divided into two areas, administration and production. Sperre has 95 employees in Norway, in addition to several distributors, sales representatives, and logistics points worldwide.

1.2. Kompressormodeller

Figur 1: Sperre kompressor "Classic" [26]

Figur 2: Sperre X-Range kompressor [28]

Figur 3: Sperre Skruekompressor [27]

1.2.1. Classic kompressor

Classic kompressoren er kjent for sin pålitelighet og ytelse og er verdens mest kjente kompressor. Den har en sterk motor og et robust kompressorkammer som tillater den å operere under særdeles krevende forhold. Sperre Air Power AS har et godt rykte for å levere høykvalitets kompressorer, og deres klassiske kompressor er et populært valg blant kundene.

1.2.2. X-Range kompressor

X-Range kompressoren er en av deres nyeste utviklinger innen luftkompressorteknologi. Denne kompressoren er kjent for sin høye ytelse og energieffektivitet, og er spesielt designet for å møte de kravene som stilles i den moderne industrien. I motsetning til Classic er X-Range helt innkapslet slik at det ikke er noen varme eller roterende deler, som gjør den til en tryggere kompressor. Kompressoren er også designet for fremtiden i den forstand at den sikrer mindre energiforbruk, et mindre fotavtrykk og en lengre holdbarhet. Kompressorens modulsystem betyr at det er høyere pålitelighet, mindre vedlikehold og færre deler. Kompressoren oljebestand er også redusert med 50%, for å redusere modellens karbonfotavtrykk. Kompressoren kommer både som luft- og vannavkjølt og har et nytt og forbedret dreneringssystem.

1.2.3. Skruekompressor

En skruekompressor er en type kompressor som produserer en jevn strøm av lavtrykkskomprimert luft. Den produseres i Kina, tross at den ikke fungerer som en konvensjonell kompressor ser man en økende trend i markedsandelen. Skruekompressoren har en annen virkemåte enn de andre kompressorene, da den bruker 2 skruer til å øke trykket i kompressoren.

1.3. Bakgrunn

En kompressor er et apparat som brukes til å øke trykket av luft til forskjellige bruksområder. Dette kan være arbeidsluft eller kan i Sperre sitt tilfelle også være vinklet mot startluft av maritime fartøy. En kompressor bruker mekanisk energi for å pumpe opp trykket ved hjelp av en bajonettmekanisme, på engelsk kalles dette reciprocating. Under regelverk går Sperre sine kompressorer under dette. Kompressorer finnes i en rekke varianter og har flere bruksområder. En kompressor kan være en stempelkompressor, en skruekompressor eller en enkel håndoperert kompressor. De mest solgte modellene til Sperre Air Power er Classic, X-range og skruekompressoren, der Classic og X-range brukes til høykompressjonsluft og skruekompressoren til lavkompressjonsluft. Denne oppgaven tar for seg en XA-150 som er en X-range, air cooled kompressor som leverer 165 $m^3/timen$.

Denne oppgaven vil ta for seg vibrasjoner mellom kompressoren og underlaget. Grunnet kompressorens V-formede sylindre vil det oppstå naturlig vibrasjon i det horisontale planet. Denne vibrasjonen vil forplante seg i kompressorens festepunkt i bakken. Av hensyn til vibrasjon, lydforurensninger, annet maskineri og utstyr er det viktig at disse vibrasjonene ikke blir for store. Oppgaven vil derfor basere seg utifra DNV sine retningsliner for en "Reciprocating Compressor".

XA-150 er som nevnt en X-range kompressor med et luftbasert kjølesystem. I forhold til resten av modellene til Sperre Air Power AS kan modell XA-150 regnes som en relativt liten kompressor, dette er likevel en av de mest solgte modellene. Oppbyggingen til en kompressor består av 5 hovedelementer: Ramme, motor, veivhus, startskap og kjølesystem. Vibrasjonenen som forekommer, oppstår hovedsaklig i "Bare-shaft" enheten som er kompressorens roterende deler. Det vil også naturligvis oppstå et moment fra motoren, men dette neglisjeres i oppgaven da det er kompressorens vibrasjoner som er i fokus. Vibrasjonene går fra stemplene og ned i veivakselen, hvor de utliknes av kontravekten som

skal motarbeide kreftene. Likevel er det vibrasjoner som oppstår og disse vil da forplante seg i rammen. Rammen har tre festepunkter der hvert segment har en gummidemper for å absorbere mest mulig av de nevnte vibrasjonene før de forplanter seg i bakken.

1.4. Problemstilling

Sperre idag, mangler en metode for å verifisere en designendring på kompressorer med en digital tilnærming, spesielt automatisert sammen med andre programvarer. Målet med denne oppgaven er å bygge et verktøy som innfrir bedriftens ønsker om simulering på en effektiv måte. Det vil også bli laget en matematisk modell for simuleringer av vibrasjoner i et bestemt system. Programmet som utvikles skal etter ønske fra bedriften kunne importere materialdata direkte fra Siemens NX, kjøre simuleringer gjennom Simcenter3D og 20-sim, for så å kunne gi ut et klart resultat for brukeren. Uten en slik modell er det veldig tidkrevende og dyrt for bedriften å teste disse metodene, da alle nye deler må spesialbestilles og monteres før de kan testes. Systemet skal lages ved en kombinasjon av matematiske og automatiserte prosesser. Sluttresultatet skal være en robust modell som kan brukes til simuleringer og prognoser for systemet. For å oppnå dette, må nye metoder utvikles, implementeres og testes gjennom ulike simuleringsscenarier for å bekrefte modellens robusthet. Oppgavens hovedprospekt er å danne en grunnmur for et system som skal utvikles og brukes i fremtiden. Dette bidrar til å redusere tids- og kostnadsbruken ved utvikling av nye deler. Modellen skal være enkel å bruke og gi tydelige resultater for brukeren. Det er viktig at det blir gjort grundige undersøkelser og vurderinger av ulike metoder for simulering av vibrasjon, og at det velges den beste metoden for bedriften og deres spesifikke behov. Det skal sørges for at systemet som utvikles kan integreres med eksisterende programvare, slik at det blir en enkel og smidig prosess for bedriften å bruke modellen.

1.5. Eksisterende løsninger på systemintegrasjon, API og designendring

For en prosess som involverer programmer med ulikt grensesnitt er APIer essensielt for kommunikasjon mellom programmene. En API tillater programmer å sende, motta, og dele informasjon uavhengig av kodespråk eller platform de er bygget på. Kode og skript som henvender seg til slike APIer må derfor testes grundig for å forsikre at de kommuniserer og henvender seg til APIen korrekt. Et slikt testforløp kan leses om i artikkel [4]. Det kan tidlig anslås at det i dette prosjektet vil bli nødvendig å flytte verdier, samle data, styre prosesser, og ta mot input. Det er derfor essensielt med en løsning for å knyte disse prosessene sammen, og skape et helhetlig system. Artikkel [13] viser til SoS-metodikk, med integrasjon av flere system og prosesser. Systemet i denne artikkelen inneholder mange verdier som kan samsvare med dette prosjektet: Sammenknyting og styringssystem for nøkkelverdier, konfigurasjoner, lagringspunkt for verdier, monitorer, brukergrensesnitt, porter, sensorer og terminaler. Artikkel [22] tar for seg planlegging, gjennomføring, og konklusjon av designendring. Her gjøres det også en omfattende verifikasjons- og valideringsprosess av designendringens utfall.

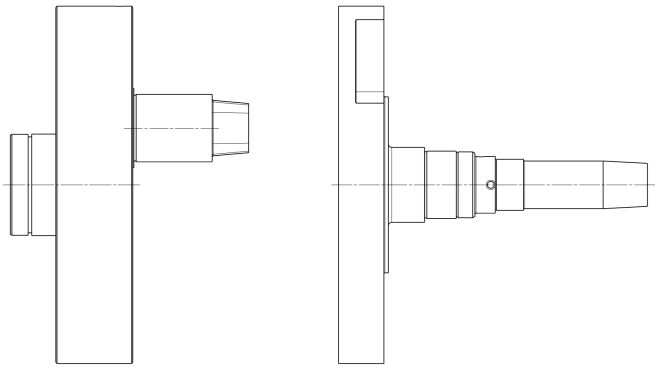
1.6. Eksisterende løsninger på vibrasjonsanalyse

Sperre bruker i dag vibrameter på for å måle vibrasjon i sine kompressorer. Et vibrameter måler hastigheten i det punktet den festes, og presenterer dette som vibrasjon. Vibrasjoner kan også måles ved å bruke et akselerometer eller dataloggere for å måle vibrasjonsnivået og vibrasjonsdata over tid, i ulike deler av kompressoren. Disse målingene kan deretter analyseres ved hjelp av spesialverktøy for å bestemme resonansfrekvensen og dempingen i kompressoren. Som vist i [21] kan vibrasjon i gummidempere avleses nøyaktig ved bruk av et akselerometer. Det kan her avleses verdier både i gummidemper og motorens topper. En nyttig metode for praktisk testing, dog tid- og plasskrevende. En annen metode er å bruke stroboskopiske bilder for å visualisere vibrasjoner. Dette gjøres ved å ta et bilde av kompressoren mens den er i bevegelse, og deretter analysere bildet for å bestemme vibrasjonsfrekvensen og amplituden. Det er også mulig å bruke Bond-Graph metoden for å simulere vibrasjoner i kompressoren ved hjelp av matematiske modeller. Denne metoden gir en mer detaljert analyse av vibrasjoner, og kan brukes til å identifisere problemområder og designe løsninger for å redusere vibrasjoner. Denne metoden brukes i dette prosjektet.

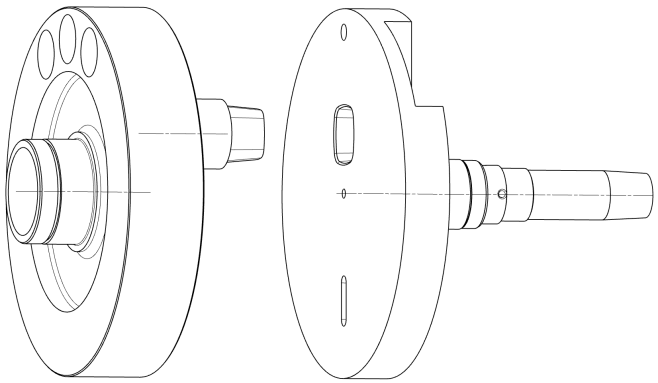
1.7. Vibrasjon som følge av feil

Det er en rekke faktorer som kan føre til uforutsette vibrasjoner i en kompressor. En av de vanligste årsakene er at kompressoren er ubalansert, enten på grunn av en ubalansert aksling eller komponenter i kompressoren. Ubalanser og feil kan skape vibrasjoner som kan skade kompressoren og omgivende utstyr over tid. Noen vanlige feil innebærer lekkasjer i tette mellomrom, skader forårsaket av slitasje, feilmontering, skade på motor eller gir, løse reimer eller kjeder og overbelastning. Denne typen vibrasjoner er uforutsigbare, og vil føre til skader på kompressoren om feilen ikke blir utbedret. Det anbefales å kontinuerlig overvåke og vedlikeholde kompressoren for å unngå vibrasjoner, dette øker både levetiden og sikkerheten til kompressoren. Sperre selger i dag service-kit som skal benyttes når kompressorer som har

gått et gitt antall timer. Det anbefales på det sterkeste å utføre service for å unngå uforutsigbare hendelser. Grunnet at kompressorene ikke skal brukes i skadet tilstand, tas det ikke hensyn til denne typen vibrasjoner i denne oppgaven.

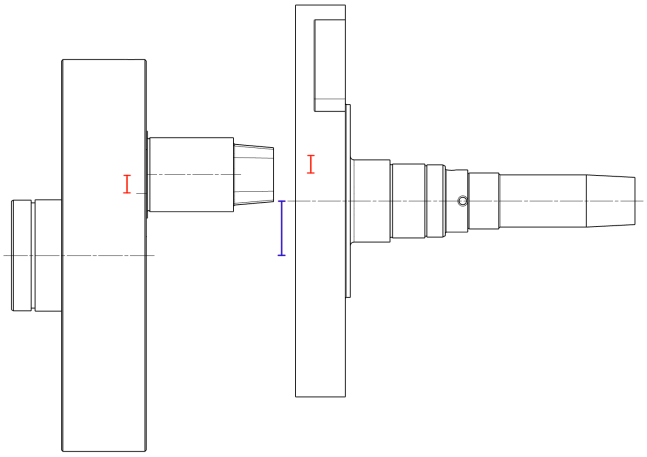


Figur 4: Kontravekt og veivtapp [7]



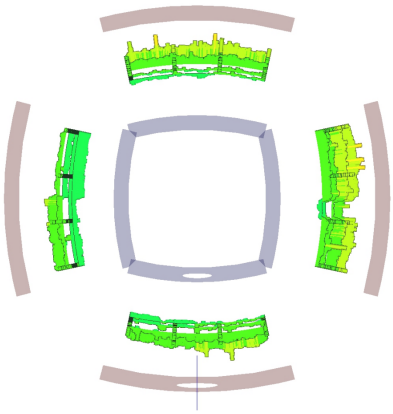
Figur 5: Kontravekt og veivtapp isometrisk [7]

Figur 4 og 5 viser en normal monteringhøyde av veivtapp og kontravekt. Dersom plassering av polygonet er utenfor referanseområde kan det risikeres at sammenstillingen blir skjev og en potensiell årsak til vibrasjoner i systemet. Sperre har i dag en CMM maskin som måler opptil titusendeler av en millimeter.

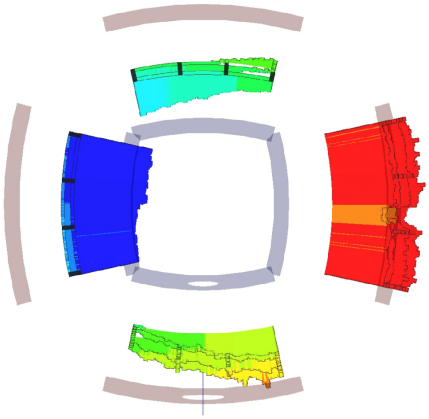


Figur 6: Forskyvning av polygon [7]

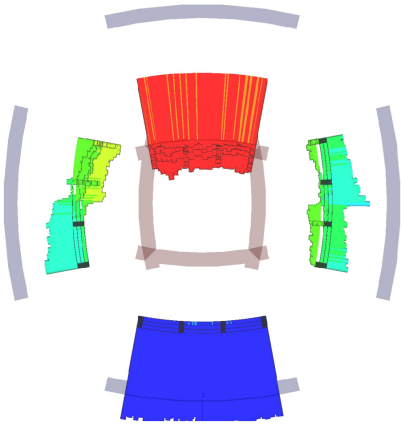
Dersom polygonet i veivtappen og i kontravekten avviker for mye blir sammenstilling som vist i Figur 6. Dette viser til et overdrevet scenario hvor konen er plassert for høyt på kontravekt og for lavt på veivtapp. Røde markeringer viser polygonets forskyvning fra sin opprinnelige posisjon og blå markering viser til endring i senterlinje. Dette kan forekomme av maskinfeil eller støpefeil. Dersom disse delene blir satt sammen vil system bli kraftig ubalansert og skjevt i horisontalt plan som vil påføre ekstra belastning i lager og andre komponenter. Dette ville introdusert en aksiell kraft og en ubalansert kraftfordeling.



Figur 7: Veivtapppolygon innenfor toleranseområde [7]



Figur 8: Veivtapppolygon horisontal forskyvning [7]



Figur 9: Kontravekt vertikal forskyvning [7]

Figur 7, 8 og 9 viser til forskyvning av polygonet der figurene består av to toleranserammer som definerer en øvre og en nedre grense. Målingene er merket grønt og gult der toleransene er innenfor de gitte rammene. Målingene er merket blått der toleranse avviker i negativ retning, og rødt der den avviker i positiv retning.

1.8. DNV Retningslinjer

DNV er en norsk sertifiseringsorganisasjon som utsteder standarder for sikkerhet og kvalitet på en rekke industriprodukter der luftkompressorer inngår. For kompressorer med stempler har DNV etablert retningslinjer med øvre grense for vibrasjoner på 30 mm/s, dette betyr at kompressoren ikke skal overskride dette. For å sikre pålitelig drift og samtidig at kompressoren ikke forårsaker skade på sine komponenter eller omgivelsene er dette et viktig krav. Sperre Air Power AS er forpliktet til å sikre at deres produkter overholder DNVs retningslinjer for å være sertifisert med ISO standarder. Alle Sperres kompressorer er testet og godkjent for å oppfylle disse kravene, som gir en garanti til kundene at kompressoren er sikker og effektiv i drift. Sperres stempelkompressorer er også underlagt andre krav som lydforurensning på maks 110db, men dette vil ikke bli fokusert på da denne oppgaven skal fokusere på vibrasjoner og vibrasjonsanalyse.

Velocity
4 - 200 Hz
30 mm/s
To be measured in any direction on the bearings. Applies for both resilient and fixed mounted.

Tabell 1: Reciprocating compressor vibration class [3]

2. METODE

2.1. Siemens NX

Siemens NX er en kraftfull og allsidig programvare for utvikling av maskintegninger og 3D modeller som brukes i en rekke industrier. NX inneholder et omfattende sett med verktøy for modellering, simulering og analyse. Dette gjør det til et utbredt og foretrukket valg for produktdesign og utvikling.

I NX refererer modellering til prosessen med å opprette digitale 3D-modeller av fysiske objekter, som deretter kan brukes til testing, analyse og produksjon. Programvaren støtter ulike modelleringsmetoder, inkludert solid-body modellering og overflatemodellering, hver med sine egne fordeler og begrensninger. Solid-body modellering er den mest brukte modelleringsmetoden, den involverer å bygge en 3D-modell av et objekt ved å definere geometrien. Dette gjøres ved å tegne opp sketcher og ekstrudere disse på ulike metoder. Denne type modellering er ideell for å opprette komplekse deler med høy grad av nøyaktighet, og den støtter fremstilling av deler med komplekse former og detaljer. NX innehar også mange andre moduler for blant annet strømning, rør, FEM-analyse, elektrisitet, og kjøretøy-design.

I dette prosjektet brukes NX til å simulere kompressorens bare-shaft system, samt å simulere dynamiske krefter under normal drift av kompressoren.

2.1.1. Journals

Journals er en funksjon i NX designet for opptak, kjøring, og redigering av kode i NX. Måten det fungerer på er at NX skriver et script som kopierer oppgavene du utfører mens applikasjonen kjører. På denne måten kan man sette opp ulike scenarier og implementere disse inn i kode man vil kjøre, eller man kan kjøre de uavhengig for enkle automatiserte prosesser.
Journals kan redigeres, skrives, og kjøres i fem språk: C#, C++, Java, Python, og Visual Basic.

2.1.2. NXOpen

NXOpen er en API som kan brukes i NX. Den har evnen til å lese og kjøre kode, og kan på denne måten utføre prosesser alene. Det er på dette viset det i oppgaven blir utført materialendring, geometriendring, og simulering gjennom NX uten å åpne NX.

2.1.3. Batch mode

Ved anvending av script eller journals i NX, har programmet en batch-mode hvor programmet kan kjøre kode og utføre oppgaver uten å åpne programmets brukergrensesnitt. Her unngås hele oppstartsprosessen, og koden kan kjøres effektivt uten at brukeren ser hva som skjer. Batch-mode er et kommandovindu som anvender NXOpen for å kjøre predefinerte journaler.

2.1.4. Simcenter3D

Simcenter3D er en del av Siemens NX, som er et kraftfullt programvareverktøy som kan brukes til å simulere krefter i tre dimensjoner (X, Y og Z) med seks grader av frihet. Programvaren lar brukeren analysere oppførselen til deler og samlinger under en rekke mulige lastbetingelser. Simcenter3D kan knyttes opp til NX da dette inngår i samme program. På denne måten kan modeller sømløst importeres til Simcenter3D. På grunn av Simcenter3D, er NX er godt verktøy å bruke for modellering, da det inneholder et bredt spekter av muligheter for brukere. I tillegg til sine kraftfulle simuleringsegenskaper, tilbyr Simcenter3D også et bredt utvalg av for- og etterprosessering verktøy som kan brukes til å analysere simuleringresultater. Dette gjør at brukere kan raskt og enkelt evaluere ytelsen til designet deres, og gjøre eventuelle nødvendige justeringer for å forbedre ytelsen. En av de viktigste funksjonene til Simcenter3D er evnen til å utføre ikke-lineære simuleringer, noe som kan hjelpe med å nøyaktig forutsi oppførselen til deler og system under virkelige lastebetingelser. Dette er spesielt nyttig når man simulerer oppførselen til deler som er utsatt for høye nivåer av stress og belastning.

2.2. 20-sim

Simuleringsverktøyet 20-sim brukes for utvikling og simulering av matematiske modeller. Det gir brukeren mulighet til å lage modeller av fysiske systemer og undersøke hvordan systemene oppfører seg under forskjellige betingelser. Med 20-sim har brukeren mulighet til å lage modeller av mekaniske, elektriske og hydrauliske systemer og simulere deres oppførsel i sanntid. Det gir også mulighet til å visualisere resultatene av simuleringene, slik at man enkelt kan se hvordan systemet oppfører seg under forskjellige forhold. 20-sim bruker Bond-Graph metoden og bruker første ordens differensiallikninger for å modellere systemene, som gjør det enkelt å representere de ulike komponentene i systemet og deres interaksjon.

20-sim bruker programmeringsspråket SIDOPS+, som er et kodespråk spesielt utviklet for å løse matematiske modeller i fysiske systemer. Modellering av matematiske modeller og Bond graph metode tilhører fag på masterstudium. Det ble derfor i denne oppgaven nødvendig å sette seg inn i et 7,5 studiepoengsfag, IP500515 "MODELING AND SIMULATION OF DYNAMIC SYSTEMS" for å få en komplett forståelse av metodens virkemåte.

2.3. Python

Python er et programmeringsspråk som med tiden har blitt veldig populært. Python er allsidig og kan brukes til mange ulike oppgaver. Som eksempel kan Python brukes til dataanalyse og vitenskapelig beregninger. Python er kjent for sin enkle og lettleselige syntaks som gjør det til et godt valg for både nye og erfarne programmerere. En av Pythons store fordeler er dets store bibliotek og mange tilgjengelige plugins. Biblioteket har mange ferdigbygde verktøy som kan løse en rekke oppgaver og gir brukeren mulighet til å lage sine egne kalkulatorer eller skript. Noen av de mest kjente er NumPy, SciPy, Pandas og Matplotlib. Python er i oppgaven brukt for å knytte 20-sim mot de andre prosessene i form av et skript.

2.4. Visual Basic

Visual Basic er et godt etablert og kraftig kodespråk bygget på Microsofts .NET Framework. Språket går for å være ukomplisert, lett tilgjengelig, med velutviklet biblioteket, og med god tilgang på ressurser. Det utmerker seg spesielt i bruk for konstruering av GUI, samt utvikling av nettbaserte, og skrivebordsbaserte programmer for Windows, IOS og Andriod. Utvikling med språket er ofte gjort gjennom utviklingsmiljøet Visual Studio, hvor Microsoft leverer utgaver både for kommersiell og privat bruk. Visual Basic er i prosjektet brukt for utvikling av GUI, koding av skript for NX, og som bindeledd mellom de to programmene.

2.5. Brukergrensesnittet

Et brukergrensesnitt er i alle system bindeleddet mellom programmer og brukere. Brukergrensesnittet gir brukere en grafisk presentasjon av kontrollpanelet for ulike programmener. Brukergrensesnittet i denne oppgaven er bindeleddet mellom alle prosessene som systemet foretar seg. Det er her bruker velger verdier for material og geometriendring.

2.6. Å skrive om kode

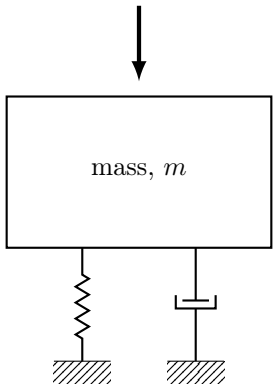
Å skrive om kode er en vanlig forekomst under utviklingen av et program. Når kravene endres eller prosjektet videreutvikles, blir det ofte nødvendig å oppdatere og forbedre koden for å tilpasse seg nye behov. En av grunnene til at koden må skrives om, er at den ble skrevet for å møte spesifikke behov eller begrensninger som ikke lenger eksisterer. En annen grunn for å skrive om koden er at den opprinnelige implementeringen kanskje ikke er tilstrekkelig effektiv eller skalerbar. Når prosjektet vokser og flere funksjoner legges til, kan det opprinnelige designet bli komplekst og vanskelig å vedlikeholde. I slike tilfeller kan utviklere være nødt til å skrive om deler av koden for å gjøre den mer effektiv. Å skrive om kode kan være en tidkrevende oppgave, men det er nødvendig for prosjektets langsikt og vedlikeholdbarhet. Det gir utviklere mulighet til å forbedre designet, optimalisere ytelsen og gjøre koden mer robust og pålitelig.

3. TEORI

Utvikling av matematiske modeller er en viktig del av å forstå og analysere mekaniske systemer. Det gir muligheten til å simulere hvordan systemene oppfører seg under ulike betingelser, uten å måtte utføre eksperimenter i virkeligheten. Ved hjelp av matematiske modeller kan det estimere systemets ytelse, identifisere flaskehalser og forutsi hvordan systemet vil oppføre seg i fremtiden. Utviklingen av en matematisk modell kan omfatte å samle inn data, formulere og løse matematiske ligninger, og validering av modellen med eksperimentelle data. Dette kan gjøres ved hjelp av spesielle verktøy som 20-sim, og kan gi oss en dyptgående forståelse av systemene og hvordan de kan optimaliseres. Den matematiske modellen er hjernen bak simuleringen og systemet. Andre komponenter som inngår i oppgavens system er tillegg for at den matematiske modellen skal fungere. I denne oppgaven blir også Simcenter3D brukt for simulering av krefter som påvirker de roterende delene i veivhuset.

3.1. Mass-spring-dampersystem

Et mass-spring-damper system er en modell som brukes til å studere forskjellige system der en masse er involvert. Systemet består av en fjær, en demper og en masse der fjæren representerer den potensielle energien mens demperen representerer tap av energi som resistans. Massen har én eller flere frihetsgrader, og kan bevege seg langs en akse eller i et plan. Oppførselen kan beskrives ved hjelp av første- eller andreordens differensiallikninger. Disse likningene beskriver hvordan hastigheten, akselerasjonen og posisjonen til massen endrer seg over tid ved en gitt kraft. Systemets bevegelse karakteriseres ved den naturlige frekvensen og dempingsforholdet hvor frekvensen indikerer hvor raskt systemet vibrerer, og dempingsforholdet indikerer hvor raskt systemet mister energi over tid. Disse egenskapene kan endres ved å justere massen, fjærkonstanten eller dempingskoeffisienten til systemet.



Figur 10: Mass-Spring-Damper system [7]

3.1.1. Fysikklover som brukes i 20-sim ved forskjellige element

Fysikklover	Bond Graph Elements	Linearitet
$F = k * \Delta x \rightarrow e = (\frac{1}{C}) * q$ $C = \frac{1}{k}$	$\frac{e}{f}$ C	En normal fjær har et linært forhold mellom kraft og forskyvning. Med stor forskyvning vil fjæren fungere som et gummibånd som har et ikke-linært forhold.

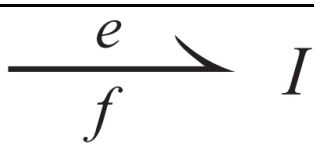
Tabell 2: Fjærelement i Bond-graph [20]

Tabell 2 viser til fysikklover knyttet opp mot fjærelementet i 20-sim, det gir også en beskrivelse av lineariteten i en fjær.

Fysikklover	Bond Graph Elements	Linearitet
$F = c * v \rightarrow e = R * f$ $R = c$		En demper kan modelleres lineært med formelen: $F_f = \mu_v * F_N * v$. Men friksjon kan også være en statisk, retarderende kraft som hindrer bevegelse i å starte.

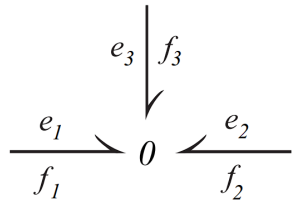
Tabell 3: Demperelement i Bond-graph [20]

På samme måte viser Tabell 3 til fysikklover til demperelementet, det gir også en beskrivele av linearitetet i en demper. I oppgaven er demperkonstanten det eneste elementet som avgir energi i form av varme. I alle andre element anses energien som bevart.

Fysikklover	Bond Graph Elements	Linearitet
$p = m * v \rightarrow p = I * f$ $I = m$		Lineær relasjon, førsteordens logikk.

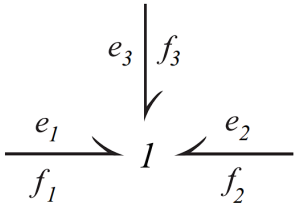
Tabell 4: Masseelement [20]

Tabell 5 viser til elementet for massetreghet med tilhørende formler og relasjon til linearitet.

Fysikklover	Bond Graph Elements
$e_1 = e_2 = e_3$ $f_1 + f_2 + f_3 = 0$	

Tabell 5: 0-Junction [20]

0-Junction brukes til å samle krefter i et punkt der all kraft blir lik, mens all hastighet summeres opp til null. 0-Junction kan brukes dersom det skal distribuere en lik kraft til forskjellige punkt.

Fysikklover	Bond Graph Elements
$f_1 = f_2 = f_3$ $e_1 + e_2 + e_3 = 0$	

Tabell 6: 1-Junction [20]

1-Junction fungerer på samme måte som 0-Junction med unntak at den er motsatt, der kraften summeres opp til null og hastigheten er lik.

	Effort (e)	Flow (f)	Momentum (p)	Displacement (q)
Navn	Kraft	Hastighet	Lineært moment	Lineær forskyvning
Symbol	F	v	(m·V)	X
Enhet	N	m/s	N·s	m

Tabell 7: Mekanisk domene lineært[7]

Tabellen for det mekaniske domenet i 20-sim viser til enheter for de forskjellige kreftene - linært.

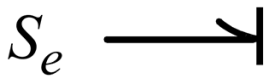
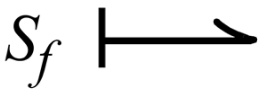
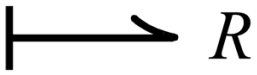
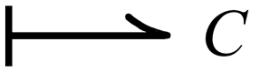
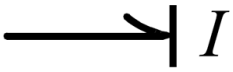
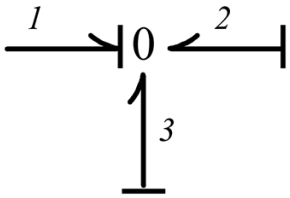
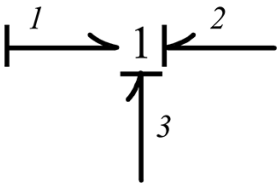
	Effort (e)	Flow (f)	Momentum (p)	Displacement (q)
Name	Dreiemoment	Vinkelhastighet	Dreiemoment	Vinkelforsyvning
Symbol	T	ω	(pτ (l+ω)	θ
Enhet	N· m	rad/s	Nm ·s	rad

Tabell 8: Mekanisk domene angulært[7]

Tabellen for det mekaniske domenet i 20-sim viser til enheter for de forskjellige kreftene - angulært.

3.2. Kausalitet

Kausualitetsanalyse er prosessen å bestemme den korrekte strukturen av modelligningene. I Bond Graph innebærer dette å identifisere retningen til krefter og hastigheter innenfor bindingene, som deretter blir avbildet ved hjelp av kausale streker (representert av "|"). Dette innebærer å identifisere retningen til variabler i forbindelsene, som deretter blir avbildet ved hjelp av kausale piler (representert av "—→"). Essensen av kausualitetsanalyse er å identifisere og representere årsak og virkning innenfor et system.

Navn	Bond Graph Symbol
Effort Source	
Flow Source	
Demper	
Fjær	
Massetreghet	
Transformator	
0-Junction	
1-Junction	

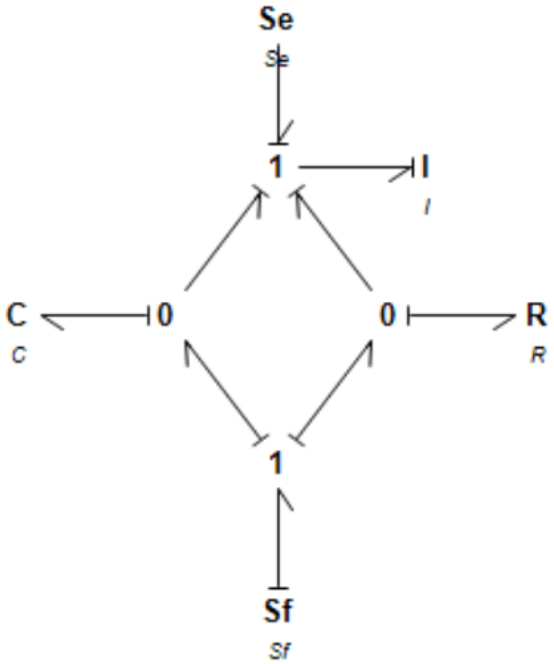
Tabell 9: Kausalitetn for hvert brukte element i 20-sim[7]

Likningen for bevegelse for et mass-spring-damper system kan representeres ved en andregrads ordinær differensiallikning av formen:

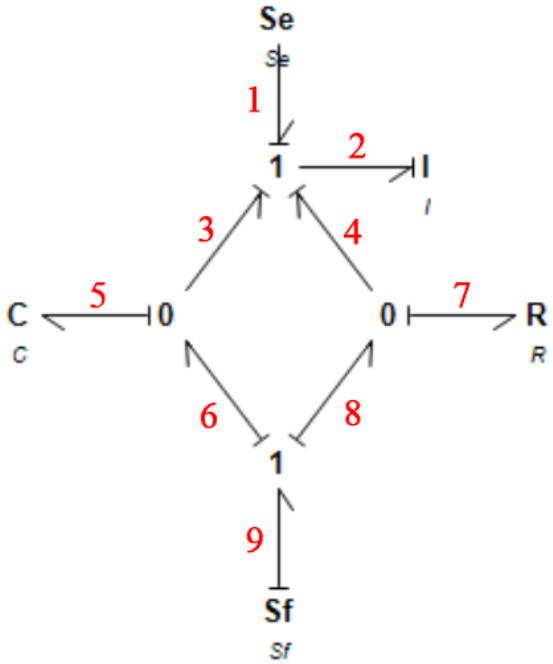
$$mx''(t) + cx'(t) + kx(t) = F(t)$$

- Variablene i likningen er:
  - m er massen til systemet (kg)
  - $x(t)$ er posisjonen til massen ved tid t (m)
  - $x'(t)$ er hastigheten til massen ved tid t (m/s)
  - $x''(t)$ er akselerasjonen til massen ved tid t (m/s^2)
  - c er dempingskoeffisienten (N·s/m eller N·s/kg)
  - k er fjærkonstanten (N/m eller N/kg)
  - $F(t)$ er den eksterne kraften som utøves på systemet ved tid t (N)

Denne andreordens differensialligningen kan løses for å finne posisjonen, hastigheten og akselerasjonen til massen som en funksjon av tid. Avhengig av de initielle forholdene og den eksterne kraften som utøves, kan bevegelsen til massen være periodisk (som harmonisk bevegelse) eller ikke-periodisk (som en dempet eller overdempet bevegelse). Det er viktig å merke seg at denne likningen antar et ideelt scenario, uten friksjon eller andre eksterne krefter som vil påvirke bevegelsen til massen. I praksis, for å få en nøyaktig modell, ville det være nødvendig å ta hensyn til andre variabler som kan påvirke oppførselen til systemet. 20-sim benytter ikke denne metoden for å løse modellen da denne heller bruker to første ordens formler, en for moment og en for forskyvning. En modell av et mass-spring-damper system kan modelleres som Figur 11.



Figur 11: Mass-Spring-Damper i bond-graph [7]



Figur 12: Mass-Spring-Damper med notasjon [7]

Ved å sette opp en notasjon på hvert av båndene blir det mulig å kunne sette opp likningene for systemet.

Manuell utregning av hvordan 20-sim jobber av et mass-spring-damper system

- Variablene i systemet er:
  - C er et fjærelement
  - R er et dempingselement
  - I er massetregheten
  - Se er kilden av kraft
  - Sf er kilden av hastighet
- For å bruke ODE må det gjøres rede for
  - $\frac{dq}{dt}$ og $\frac{dp}{dt}$
  - der p er moment
  - der q er forskyvning

Benytter regler for 1- og 0-Junction: Benytter notasjon fra Figur 12

$$\dot{q} = f_5 = f_6 - f_3 = S_f - \frac{p}{I}$$

$$f_3 = f_2 = \frac{p}{I}$$
$$f_6 = f_9 = S_f$$

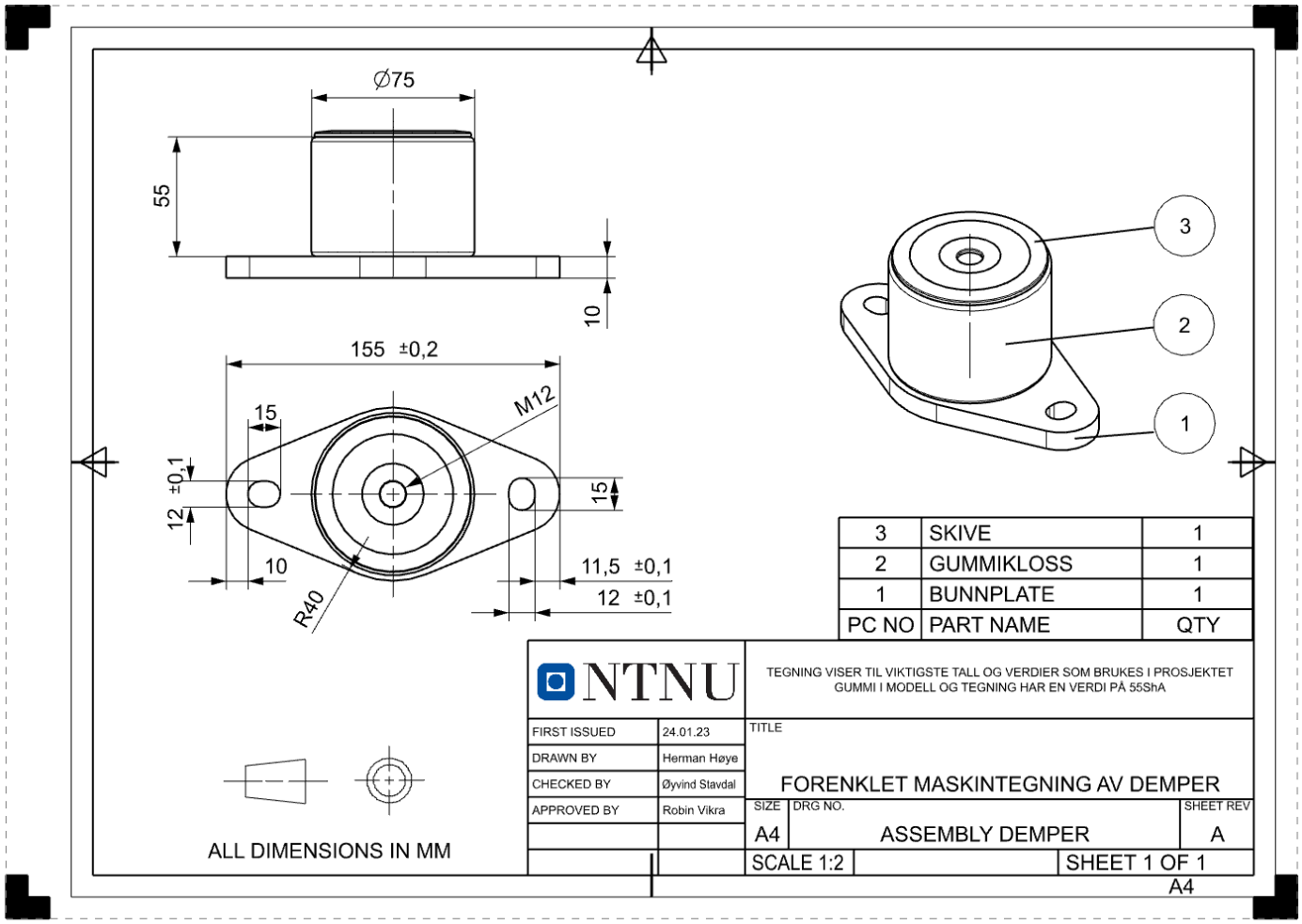
$$\dot{p} = e_2 = e_3 + e_4 + e_9 = \frac{q}{C} + R(S_f - \frac{p}{I}) + S_e$$
$$e_3 = e5 = \frac{q}{C}$$
$$e_4 = e_7 = R * f_7 = R(f_8 - f_4)$$
$$= R(S_f - f2) = R(S_f - \frac{p}{I})$$
$$f_8 = S_e$$
$$e_1 = S_f$$

Ved å modellere i 20-sim kommer kausaliteter automatisk utifra kraftretning. Forskjellige komponenter er definert etter lov om fysikk men kan modelleres feil dersom fokus ikke ivaretas. Det er derfor viktig å påse at kausalitet er riktig, for å sikre at modellen blir autentisk og korrekt. Dette segmentet kan anses som en beskrivelse på oppgavens utregninger, da modellen baserer seg på en Mass-Spring-Damper system som innehar en oppdeling av demperene og er derfor for kompleks til å regne ut for hånd.

3.3. Gummidemper

Bruken av gummidemper for en kompressor som vibrerer, kan være en god løsning når det gjelder å redusere vibrasjoner som overføres til bakken. Gummimaterialer har naturlige dempende egenskaper som gir dem evnen til å absorbere vibrasjoner.

Sammenlignet med ståldemper, kan gummidemper redusere vibrasjoner med opptil 50% eller mer. Dette kan være spesielt viktig i områder der det er økt bekymring for støyforurensning. Gummidemper kan også bidra til å redusere belastningen på kompressoren selv, noe som kan øke levetiden til kompressoren. I tillegg til å redusere vibrasjoner og støy, har gummidemper også flere fordeler. De er motstandsdyktige mot korrosjon og kan tåle høye temperaturer og værforhold. Gummidemper er også lettere og krever mindre vedlikehold enn ståldemper.



Figur 13: Maskintegning demper [7]

Maskintegning av gummidemper, forenklet for bruk i denne oppgaven. Modellen består av 3 forskjellige deler: Fundament for innfesting, gummisylinder og skive. Simuleringen i 20-sim tar kun for seg gummidelen av demperen med høyde 55mm og diameter 75mm på sylinderen.

3.3.1. Shore A

ShA er en måling som brukes til å indikere hardheten til et materiale, spesielt gummi- og plastmaterialer. ShA-skalaen går fra 0 til 100, med høyere tall som indikerer et hardere materiale.

Når det gjelder bruken av ShA i innfestninger for en gjenstand som vibrerer, kan materialene med 40 ShA og 55 ShA være gode valg avhengig av de spesifikke kravene til applikasjonen.

3.3.2. 40 Shore A

40 ShA materialer er relativt myke og ofte brukt i applikasjoner der fleksibilitet og en god tetning er viktig, som for eksempel i pakninger og tetninger. Det har lavere motstand mot deformasjon og slitasje, og kan lettere kompresjonsformes enn hardere materialer. Det kan være et godt valg for gummidemper som skal redusere vibrasjoner og støy i en gjenstand.

3.3.3. 55 Shore A

55 ShA materialer som er mer moderate i hardheten og har mer motstand mot deformasjon og slitasje enn 40 ShA materialer. Det er ofte brukt i applikasjoner der styrke og holdbarhet er viktig. De har mindre fleksibilitet og kompresjonsformes ikke så lett som mykere materialer. Det kan være et godt valg for innfestning som skal øke stabiliteten og holdbarheten til en gjenstand som vibrerer.

Hovedforskjellen mellom 40 ShA og 55 ShA er hardheten, der 55 ShA er hardere og har mer motstand mot deformasjon og slitasje. Valget mellom de to avhenger av de spesifikke kravene til applikasjonen.

3.4. Fjærstivhet i gummidemper

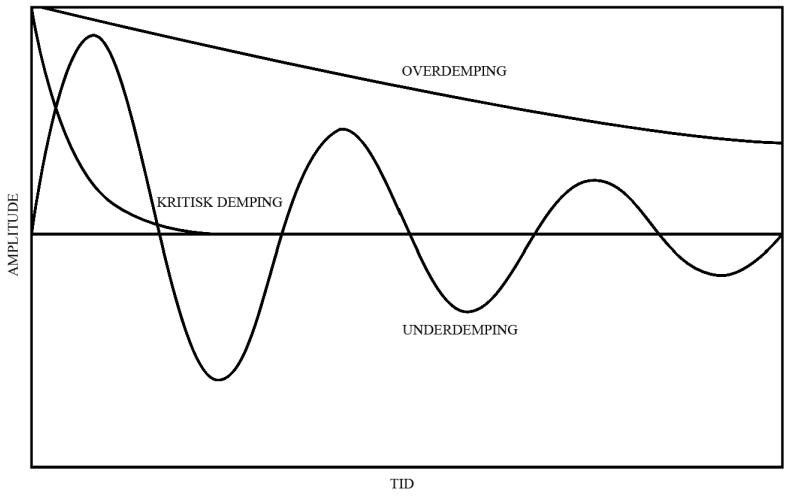
Type	D	H	Compressive loads						Shear stresses					
			Spring rate c2 in N/mm			Permissible load F in N			Spring rate cx, y in N/mm			Permissible load F in N		
			hard	medium	soft	hard	medium	soft	hard	medium	soft	hard	medium	soft
A	20	15	300	190	120	500	320	200	60	40	30	190	120	70
A	30	15	670	410	250	1100	700	400	90	60	40	350	210	130
A	30	30	240	150	100	900	570	340	50	30	20	430	280	170
A	40	30	480	300	170	1800	1110	670	90	60	30	770	500	250
A	50	20	2400	1500	900	5000	3190	1870	240	160	100	1200	770	460
A	50	40	600	380	220	2800	1750	1050	120	80	50	1280	800	460
A	75	25	5000	1655	1700	8000	5000	3300	410	260	160	2800	1750	1030
A	75	55	650	400	240	4700	3000	1750	130	80	50	2100	1300	800

Figur 14: Tabell som viser fjærstivhet for gummidempere [16]

Figur 14, viser til fjærstivhet for materialtype Shore A, der hard = 70ShA, medium = 55ShA og soft = 45ShA. Under D og H finner vi diameter og høyde for demperen. I tilfellet av oppgaven er demperen 75mm i diameter og 55mm i høyde. Sperres gummidempere har 55Sha og fjærstivhet blir derfor 400N/mm for kompresjon og 80N/mm for skjærkraft.

3.5. Over-, under- og kristisk-demping

Overdemping, underdemping og kritisk demping er begreper som brukes i forbindelse med mekaniske systemer for å se hvor effektivt noe dempes. Overdemping skjer ved at systemet er dempet for mye og mister sin evne til å nå resonansfrekvensen innenfor en gitt tid. Dette kan medføre at systemet ikke fungerer optimalt. Underdemping er motsatt av overdemping der det er for lite demping i systemet og kan føre skade eller at systemet blir ustabilt. Kritisk demping er det optimale nivået av demping i et system der det er nok demping til å hindre ustabilitet, men ikke nok til å forhindre resonans. I en kompressor er demping viktig for å kontrollere vibrasjoner og for å unngå skade på kompressoren og omgivelsene rundt. Demping kan regnes ved å måle amplituden av vibrasjonen over tid og sammenliknes med resonansfrekvensen til systemet. Det er også mulig å beregne dempingen dersom dempingsrate er oppgitt.



Figur 15: Over-, under-, og kritisk-demping [10]

3.5.1. Beregning av aktuell demping

Utregning av demping i oppgavens modell:

$$C_c = 2 \cdot \sqrt{km}$$

$$\zeta = 0.075$$

$$\zeta = \frac{C}{C_c}$$

$$\zeta = \frac{C}{(2 \cdot \sqrt{km})}$$

Skriver om formelen for å finne C

$$C = \zeta \cdot 2\sqrt{km}$$

Hvor, hvis dempningsforholdet er større enn 1, indikerer dette at systemet er overdempet.

- Variablene i likningen er:
  - C er aktuell demping
  - k er stivheten til gummi
  - m er massen til systemet
  - ζ er dempingraten / dempingforholdet

Løsningen av denne likningen vil gi dempingsraten:

Vertikal demping

$$C = 0.075 \cdot (2\sqrt{(400 \cdot 570)}) = 71.62$$

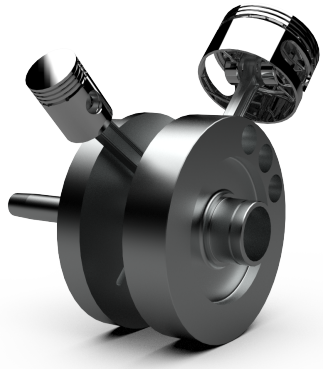
Horisontal demping

$$C = 0.075 \cdot (2\sqrt{(80 \cdot 570)}) = 32.03$$

4. UTVIKLING

Systemet som utvikles inneholder mange ledd der flere prosesser knytes sammen. Deler av systemet er forenklet for å effektivisere utviklingen uten at det går på bekostning av kvaliteten. For systemets del er det da viktig å ikke overkomplisere. 3D modeller i Siemens NX er derfor redusert til kun kompressorens bevegelige deler. Dette gir systemet mulighet til å hente ut krefter som virker ned i aksling fra kompressoren, men at alle andre deler av kompressoren som ikke er relevant for oppgaven bortfaller fra modellen. Dette er deler som utskillerhus, kjøleanlegg, reimer og lignende. Kreftene som hentes ut fra 3D modell er totale krefter som virker i x- og y-retning fra kompressoren der en gitt omdreiningshastighet er satt som lastbetingelse.

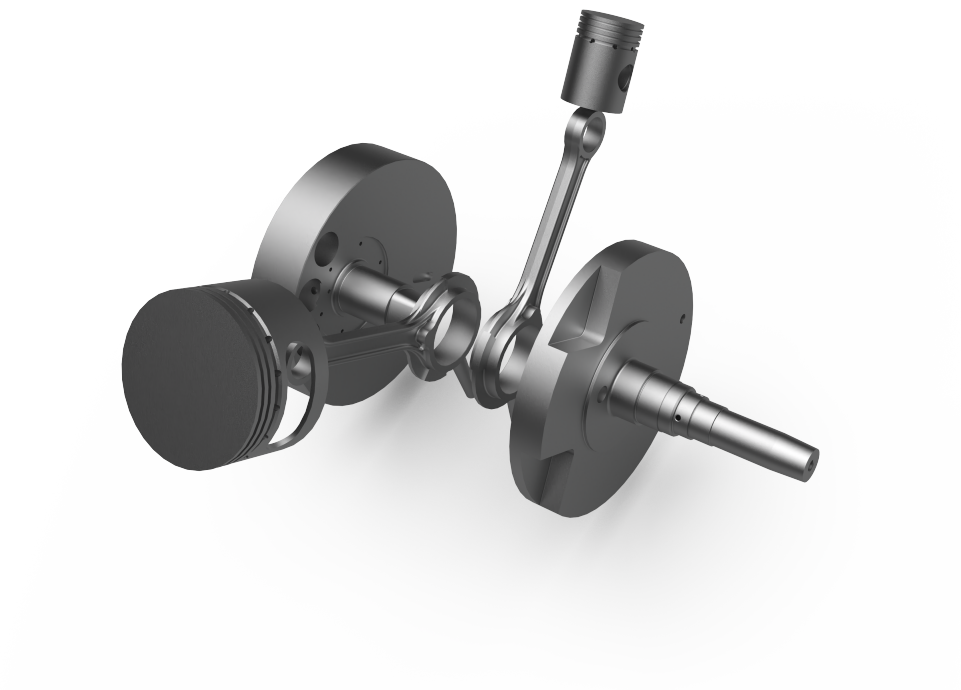
Figur 16: Kompressordeler [7]



Figur 17: Forenklet XA-150 kompressor [7]

4.1. Siemens NX og Simcenter3D

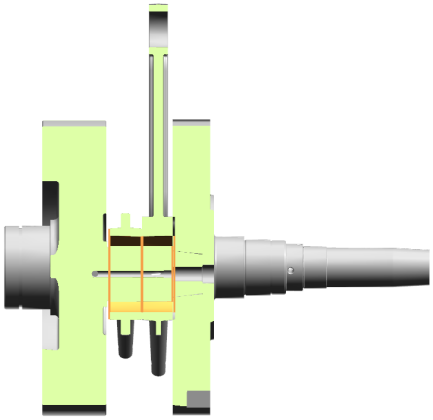
Funksjonen til Siemens NX i denne oppgaven har to hovedpunkter: Å forandre stempelenes material og/eller kontravektens geometri. Deretter videresendes assemblyen til NXs modul for simulering, Simcenter3D. Deler som inngår i modellen er kontravekt, veivtapp, høytrykk- og lavtrykk-stempel og veivstaker uten påmonterte plaskepinner. Kontravekten og veivtappen fungerer som aksling og motvekt. Disse settes sammen med veivstaker mellom. I tillegg er det 2 forskjellige stempler, ett for høytrykk og ett lavtrykk (liten og stor respektivt). Selvom størrelsen på stemplene varierer er det 2 identiske veivstaker. Modellen som skal simuleres er en forenklet modell av en kompressor uten rammeverk og innkapsling. Modellen er en forenklet modell av en "Bare-shaft" som kun er kompressor uten motor. Modellen er uten veivhus, ventiler, sylindre, kjølesystem, utskillerhus og alle tilhørende deler. Alle operasjoner i NX er skrevet og fremstilt i Microsoft Visual Basic. Kodene er fremstilt ved hjelp av ressurser fra Siemens selv, koding i Visual Studio, ressurser på internett, og ved bruk av NXs egen journal funksjon.



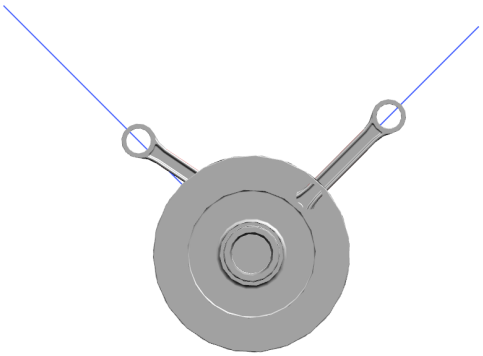
Figur 18: Deler brukt i NX[7]

4.1.1. Modellens oppbygning

Delene brukt i prosjektet er hentet fra Sperres egne modellfiler for en XA-150 kompressor. I NX er delene sammensatt ved hjelp av ulike begrensninger definert i en assemblyfil. I denne introduseres alle de ulike delene, og det defineres begrensninger for hver del for å forsikre at sammensetninger representerer en ekte bare-shaft assembly. Den første delen som introduseres til assemblyfilen er veivtappen. Denne ligger plassert sentrert i XZ planet, og fungerer som et godt utgangspunkt for videre introdusering av deler. Neste del som introduseres er kontravekten. Denne har en firkantet polygon som skal presses og skrues fast til veivtappen. Siden NX ikke fester deler ved hjelp av skruer, legges det her inn et kriterium om at kontravektens tapp må røre innerveggen i kontravektens matchende utfresning. På denne måten er veivtappen og kontravekten låst i riktig posisjon. Videre introduseres veivstakene. Kriteriet som settes her, er at stakene skal sentreres rundt akslingen på veivtappen. Deretter defineres det at veivstakenes sidevegger skal røre hverandre, og til sist at de skal berøre enden av akslingens sylindriske overflate. Veivstakene har da to frihetsgrader, RY og langs Y-aksen. At veivstakene kan bevege seg i Y-akse betyr at det er rom for seiling lags aksen og bevegelse i RY tilsier at de kan rotere at den kan om Y aksen.
Videre må stempelbanene defineres. Kompressorens sylindere har en V utforming, med 90° mellom sylinderne, eller 45° fra systemets Z akse. For å løse dette introduseres det to enkle sketcher tilsvarende stemplenes bane i kompressorens sylindere, se Figur [20]. Veivstakenes kryssbolt-hull sentreres deretter til stempelbanene, og er da korrekt definert i forhold til assemblyen.



Figur 19: Snittet viser kriterier for plassering av veivstaker [7]



Figur 20: Sketcher illustrerer stemplenes bane i syllinderne [7]

Det eneste som gjenstår da er å introdusere stemplene til assemblyen. Først blir lavtrykk-stempelets kriterier definert. Det første kriteriet som introduseres er at stemplenes kryssbolt-hull sentreres etter veivstakens kryssbolt-hull. Deretter settes et kriterium for at stemplets senterpunkt skal være på linje med den definerte stempelbanen. Den eneste frihetsgraden som gjenstår da, er stemplets rotasjon rundt kryssbolten. Her defineres det at stemplets vegger må være parallell med stempelbanen. Stempelet korrekt posisjon i forhold til assemblyen, og kan ansees som ferdig definert. Denne prosessen gjentas for høytrykks-stempelet, og systemet er ferdig definert.

4.1.2. Materialvalg

Den første oppgaven som skal håndteres av NX i det autonome simuleringsforløpet, er endring av material i systemets stempler. I NX kan alle deler i et system tildeles en materialattributt. Gjennom NXs mangfoldige materialbiblioteker finner man metaller, plastikk, og gasser. For denne oppgaven er det kun metaller som er interessant. Attributtene delene får når de tildeles et material innebærer styrke, varme, formbarhet og massetetthet. For simulering av vibrasjoner er massetetthet ofte den avgjørende faktoren for om systemet vil vibrere mer eller mindre ved endring, som følge av vektendring i stempelet.
Da stemplene er to uavhengige filer er det to måter å løse dette på: Enten åpner man hver fil og redigerer stemplene hver for seg. Eller så redigerer man stemplene i assemblyen de ligger i. Å redigere i assemblyfilen krever et par ekstra steg, men å redigere stemplene hver for seg krever åpning av flere filer. Da åpning av nye filer krever mer prosessorkraft enn å kjøre en litt lengre kode, blir det her derfor utført redigering i assembly. For å utføre autonome oppgaver er det NXs egen API, NXOpen som blir brukt.

4.1.3. Skript for materialendring

Den første automatiserte oppgaven i NX er da Visual Basic skriptet som skal forandre stemplenes materialer i bare-shaft assemblyen. Kriteriet her er at skriptet skal startes via GUI, deretter skal skriptet kjøres gjennom NXs egen batch mode. På denne måten ser ikke brukeren at NX arbeider, og nødvendig prosessorkraft fra brukerens PC reduseres betraktelig. Det blir tidlig bestemt at skriptet skulle skrives i Visual Basic, da manualer for NXs API var skrevet i Visual Basic. I skriptet er det spesifisert hvilken fil som skal åpnes, og hvor den ligger. Skriptet leser selv hvilket material som skal brukes, påfører dette, og lagrer den nye assemblyen klar for simulering.

Den første utfordringen ved et automatisert materialskifte, er at verdien som skal angi materialet vil variere hver gang skriptet kjøres. Materialet blir i prosessen definert i tekstfilen `Selected_material.txt` som en variabel, og skriptet for materialskifte kan lese dette ved hjelp av koden vist på Figur 21.

```
15  Sub Main (ByVal args() As String)
16          Dim theSession As NXOpen.Session = NXOpen.Session.GetSession()
17          '----------------------------------------------------
18          ' Material lagres som streng fra tekstfil
19          '----------------------------------------------------
20          Dim material As String
21          Dim FilePath As String = "C:\Menu\selected_material.txt"
22          Using reader As StreamReader = New StreamReader(FilePath)
23              material = reader.ReadLine
24              End Using
```

Figur 21: Kode for lesning av fil og setter material som variabel [7]

Når materialet er lagret som variabel starter neste prosess, som er å åpne assembly filen med bare-shaft delene. Lokasjonen til denne filen er definert i koden, men vil etter installasjon av programmet alltid være på samme lokasjon:

`"C:\Menu\Kompressor\assembly.prt"`

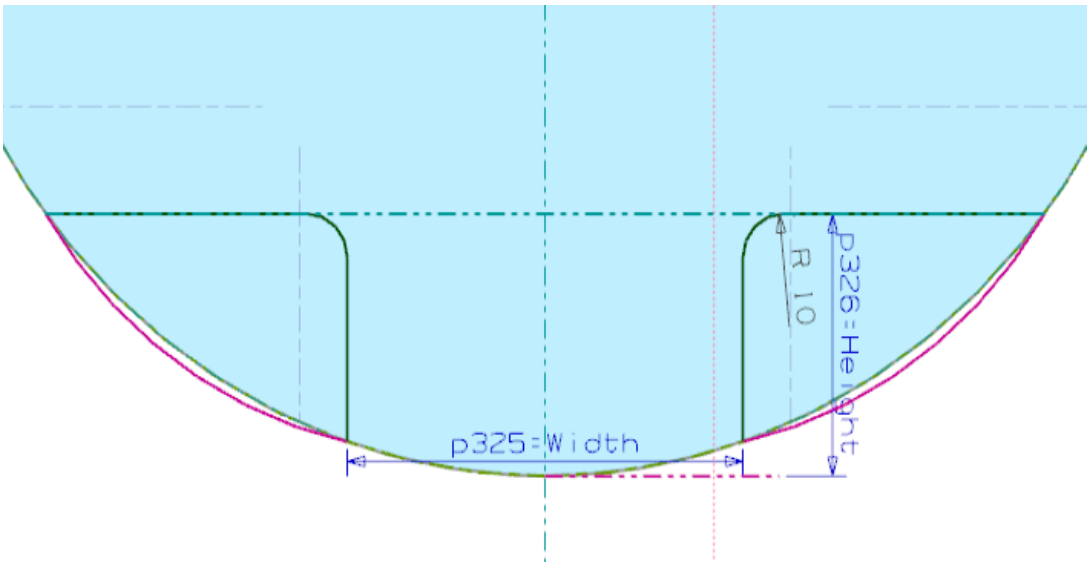
Deretter spesifiseres det at arbeidet skal utføres i NXs Modellingsmodul. Gjennom dette skriptet åpnes det kun én fil, hvor det behandles to stempler. På denne måten blir det unngått å måtte åpne to separate filer, som ville økt tiden brukt for å utføre prosessen. Et viktig aspekt med skriptet er at det arbeider ut ifra modellnavnene til stemplene. Dette fungerer bra til bruk i denne oppgaven, men skal det introduseres andre kompressorer med ulike stempel, må dette konfigureres i koden.

Når sammenstillingen er åpnet, spesifiseres det hvilket stempel som først skal behandles. Når dette stempelet er valgt slettes stemplenes nåværende, og ubrukte materialer i filen. Å slette ubrukte lastede materialer er viktig for å forhindre bugs og feilmeldinger ved lasting av et allerede innlastet materiale. Med alle innlastede materialer fjernet, kan den nye materialverdien problemfritt tildeles stempelet. Denne prosessen gjentas for det andre stempelet, og begge har da materialverdien angitt fra tekstfilen. Når materialendringen er utført, går NX tilbake til main assembly-view, og lagrer delene med de nye material-attributtene. Systemet er da klar for simulering.

4.1.4. Geometriendring

Et annet punkt av oppgaven, var å kunne utføre geometriendring på kompressorens roterende sammenstilling. Med tanke på klaring i veivhuset, og for å ikke skape utfordringer ved maskinering ble det valgt å legge disse endringene til kontravekten. I denne eksisterer det allerede to utfresinger som er med på å balansere massesenteret til systemet som brukes i dag. Ved å forandre denne utfresingen til å fjerne mer elle mindre masse, vil dette påvirke systemets massesenter og vibrasjon som følge av dette. For å kunne ha et skript som selv skulle kunne utføre en geometriendring var parametrisering av de eksisterende sketchene til utfresingene av kontravekten det første steget. Her var gode definisjoner av mål og sketcher essensielt, for at utfresingene skulle holde riktig geometri til tross for forandring av mål.

De ulike målene knyttes opp mot variablene: Width, Height, og Depth. På denne måten kan man enkelt forandre utfresningen til å ha mer eller mindre masse. Med en parametrisert sketch kan fremstillingen av et geometriendrende skript iverksettes.



Figur 22: Variabler i sketch [7]

4.1.5. Skript for geometriendring

For denne prosessen er det også utvikles et Visual Basic skript som kan selvstendig kjøres av NXs batch mode. Som vist Figur 23 brukes variablene: Height, Depth, og Width, for å bestemme lengden til de ulike veggene i utfresingen. I starten av skriptet plukkes variablene på samme måte som i skriptet for materialendring, ved bruk av funksjonene "reader" og "ReadLine". Når de tre variablene er definert i skriptet, deklareres det at de ulike variablene skal settes som de nye verdiene i utfresingen. Videre utføres prosessen med geometriendring manuelt gjennom NXs journalfunksjon for å lage en "dummy" kode. Prosessen gjennomføres fra starten ved åpning av modellen. Derfra endres alle tre verdiene, og modellen lagres. Koden som da fremstilles inneholder alle elementer som trengs from å lage et automatisert og variabelt skript. Skriptene kombineres og verdiene fra journalen erstattes med variablene definert i oppgavens program. Koden som fremstilles av journalen inneholder derimot mye unødvendig kode uten forklaringer, og etter funksjonstesting av skriptet fjernes og erstattes de unødvendige kodelinjene og forklaringer legges til.

```
56
57      Dim expression1 As NXOpen.Expression = CType(workPart.Expressions.FindObject("Height"), NXOpen.Expression)
58      Dim unit1 As NXOpen.Unit = CType(workPart.UnitCollection.FindObject("MilliMeter"), NXOpen.Unit)
59      workPart.Expressions.EditExpressionWithUnits(expression1, unit1, Height)
60
61      Dim expression2 As NXOpen.Expression = CType(workPart.Expressions.FindObject("Depth"), NXOpen.Expression)
62      workPart.Expressions.EditExpressionWithUnits(expression2, unit1, Depth)
63
64      Dim expression3 As NXOpen.Expression = CType(workPart.Expressions.FindObject("Width"), NXOpen.Expression)
65      workPart.Expressions.EditExpressionWithUnits(expression3, unit1, Width)
66
67      '------------------------------------------------
68      '    Variabler deklareres som nye verdier
69      '------------------------------------------------
70
71      Dim objects1(2) As NXOpen.NXObject
72      objects1(0) = expression1
73      objects1(1) = expression2
74      objects1(2) = expression3
75
```

Figur 23: Kode for målsetting med variabler [7]

4.2. Simcenter3D

For å skape et virkelighetsnært scenario defineres det i forkant hvor og hvordan delene er sammenkoblet, hvilke deler som er fast innspent og hvilke deler som kan bevege seg fritt. På denne måten kan man definere en setting hvor delene utsettes for krefter som tilsvarer en ekte kompressor. I denne oppgaven brukes Sperres egne filer for veivhusets bevegelige komponenter, og en simuleringsmodell blir satt opp fra bunnen av. På denne måten forsikres det om at simuleringene er så virkelighetsnære som mulig. Videre defineres kraften fra motoren, tyngdekraft, og simuleringen kan kjøres. For å kunne kjøre en simulering må først en assembly importeres til Simcenter3D. Her importeres sammenstillingen av de roterende delene i en XA-150. Da disse ble posisjonert korrekt tidligere holder de nå riktige utgangsposisjoner, men her må joints, motion bodies, drivkraft, og et scenario introduseres for å ha en komplett simulering.

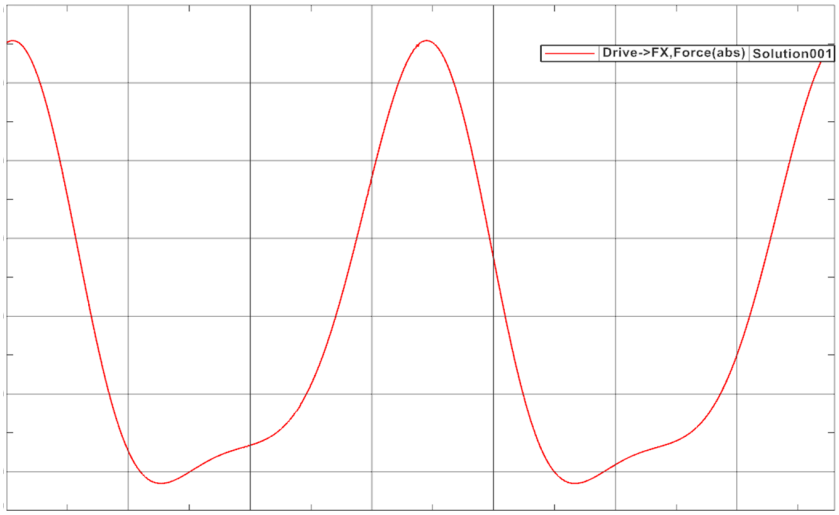
4.2.1. Motion Bodies

Motion bodies må først defineres, slik at Simcenter3D vet hvilke deler som finnes i simuleringen. Her blir sammenstillingen delt inn i sine respektive deler: Veivtapp, kontravekt, stempler og veivstaker. Her har Simcenter3D selv en automatisk funksjon som brukes for å beregne massesenter og egenvekt. Det er essensielt at denne funksjonen brukes, da stemplene og veivtappens verdier vil variere og krever oppdatering hver gang modellen skal simuleres. Når delene har fått definerte verdier, må det kobles inn joints for å binde delene sammen.

4.2.2. Joints

For å definere koblingspunktene brukes ulike joints for å definere hvor mange frihetsgrader de ulike delene har. Når komponenter introduseres til Simcenter3D er det essensielt at komponentene defineres med korrekt antall frihetsgrader. Blir det gjort feil vil simuleringen presentere ukorrekte resultater, og være til liten nytte. Etter definering av motion bodies i den nye simuleringsmodellen, består systemet av 6 komponenter. For å få en realistisk simulering brukes det joints tilsvarende delenes aktuelle festemetode. De aktuelle jointsene som er brukt i dette oppsettet er:

- **Cylindrical joint:** Brukes i koblinger hvor deler er festet til en sylindrisk flate. Dette tillater rotasjon rundt overflaten, samt bevegelse langs flatens akse.
- **Revolute driver-joint:** Låser komponenten til null frihetsgrader. Her skal komponenten kun rotere rundt én bestemt akse i en gitt hastighet. Denne rotasjonen er konstant og regnes derfor ikke som en frihetsgrad. Rotasjonen her er satt til 1780 RPM, som tilsvarer standard arbeidsturtall for den gitte kompressoren.
- **Fixed joint:** Låser to komponenter til hverandre. Her kan det defineres hvilke frihetsgrader som skal ekskluderes.



Figur 24: Graf av x-kraft fra simulering [7]

4.2.3. Chebychev–Grüebler–Kutzbachs kriterium

Det er etter introduksjonen av joints at Chebychev–Grüebler–Kutzbachs kriterium blir relevant for oppgaven. Men dette kriteriet kan man enkelt regne ut hvor mange frihetsgrader det komplette systemet har. På denne måten kan man enkelt se om systemet har for mange eller for få frihetsgrader. Har systemet for få, vil de ulike delene ikke bevege seg riktig i forhold til hverandre. Har det derimot for mange, vil systemet ikke klare å bevege seg som ønsket. NX løser dette problemet med å fjerne ulike constraints slik at systemet kan bevege seg, dette gir feil resultater da delene ikke nødvendigvis er festet på et vis som lar krefter overføres. Det er derfor essensielt å velge riktige koblinger. Når koblinger er definert kan man bruke dette kriteriet for å kontrollere at systemet har riktig antall frihetsgrader. Ved bruk av dette kriteriet, skal systemet Gruebler count være ≥ 0 avhengig av hvor mange frihetsgrader det har.

Chebychev–Grübler–Kutzbach kriterie er gitt ved formelen:

$$M = 6n - \sum_{i=1}^{j}(6 - f_i)$$

- Der variablene i systemet er:
  - M er det samlede systemets frihetsgrader.
  - n er individuelle komponenter i systemet.
  - j er antall joints som blir brukt i systemet.
  - f er antall frihetsgrader i det definerte joint som brukes mellom komponentene.
- I denne assemblien er det 6 individuelle komponenter.
  - Det er brukt 6 cylindrical joint, som låser bevegelse til 2 frihetsgrader.
  - Det er brukt 1 fixed joint, som låser bevegelse til 2 frihetsgrader.
  - Det er brukt 1 Cyllindrical driver-joint, som låser bevegelse til 0 frihetsgrader.

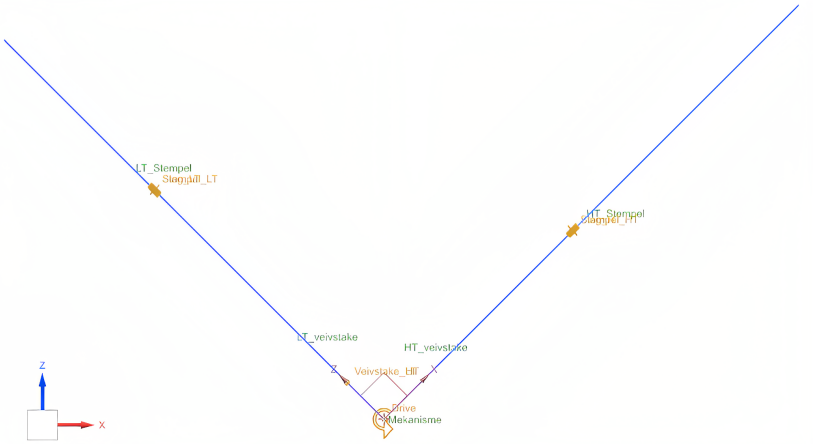
Ved å sette dette inn i formelen blir likningen:

$$6 \cdot 6 - (6-2) - (6-2) - (6-2) - (6-2) - (6-2) - (6-2) - (6-2) - (6-0) = 2$$

Her viser Grüblers kriterie at systemet er definert med 2 frihetsgrader. Det kan da konkluderes med at joints og komponenter er definert korrekt, og at dataene som produseres i simuleringen vil være av tilstrekkelig kvalitet.

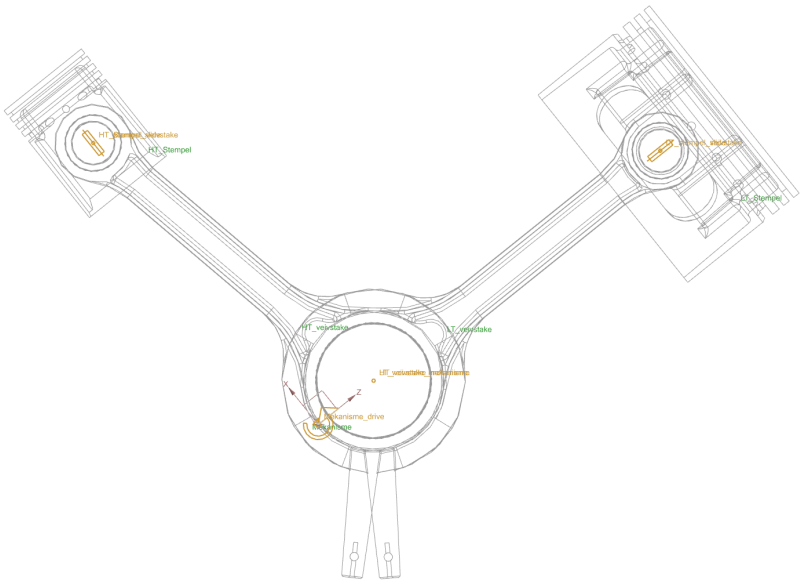
4.2.4. Skript for simulering

Når programmet har kjørt skript for materialendring og den oppdaterte filen er lagret, kjøres skriptet for simulering. Her defineres det først hvilken fil som skal åpnes, som i denne oppgaven er mech_simulation. Videre defineres det også at denne skal åpnes i NXs Simcenter3D-modul, hvor simuleringer gjennomføres.



Figur 25: Defineringer av baner og festepunkter [7]

Figur 25 viser koblingspunktene i simuleringensmodellen. Her defineres det hvor de er festet, og hvilke baner de skal følge. Sketchen er sentrert rundt veiven som er låst med en revolute driver joint, og driver hele modellen. Banene er definert som føringer i senter av kompressorens sylindre.



Figur 26: Simulering av veivstaker med slider, joints og drive [7]

4.2.5. Krefter i simulering

Scenariet som settes for simuleringen går over 0.0674s, som tilsvarer 2 komplette omdreininger av systemet når det roterer med 1780 RPM. Simuleringen kjører avlesninger i systemet med faste intervaller under simuleringen, satt til å leses hvert 9.3611111111e-05 sekund. Med disse verdiene satt, leser Simcenter3D FX og FY verdier i systemet hver 5. grad i to komplette omdreininger.

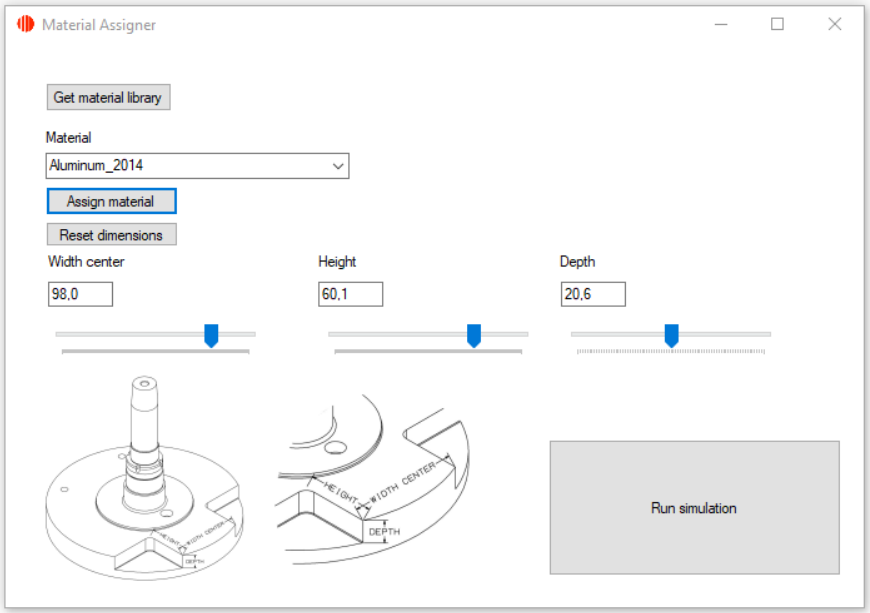
4.2.6. Resultatet fra NX

Når simuleringen er komplett, kan resultater for FX og FY krefter leses av i grafer. Verdiene fra grafene eksporteres deretter som .CSV filer, som videre kan importeres av 20-sim for å simulere vibreringen disse kreftene vil forårsake.

4.3. Graphical User Interface

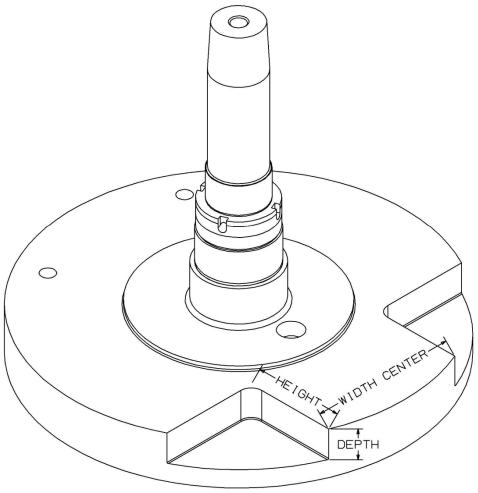
Opprinnelig ble utviklingen av GUI gjort i Python for å dele grensesnitt med 20-sim. NXOpen pakken til Python er relativt ny og lite utviklet som gjør det mer utfordrende å arbeide med Python enn andre godt etablerte kodespråk som Visual Basic. Visual Basic har et godt utviklingsmiljø, Visual Studio, som er kjent for sin brukervennlige modul for design av GUI.

I oppgaveteksten er det definert at det skal kunne endres material i stempler og geometri av kontravekt. GUI må derfor inneha forskjellige funksjoner som gir bruker mulighet til å endre systemets simuleringskriterier. Materialbibliotek importeres her direkte fra NX, presenteres i GUI, og gir bruker mulighet til å velge blant metalliske materialer. Bruker kan endre verdiene for kontravektutfresning i et område innenfor definerte mål. Disse kan tilbakestilles til utgangsverdier ved hjelp av en resetknapp. Det er nødvendig å ha en knapp for å starte simuleringen, og ved bruk av denne starter brukergrensesnittet skript for material- og geometriendring og simuleringene.

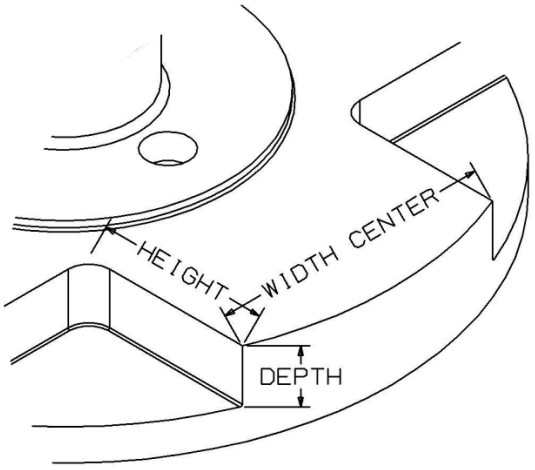


Figur 27: Graphical User Interface [7]

Et skript for filtrering av materialdata henter ut alle metall og presenterer disse i en rullegardinsliste. For å lagre valgt materiale må knappen Assign Material benyttes. For geometriendring er standard verdier av utfresning satt, verdiene kan endres for å variere størrelse på utfresning. Dette gjøres ved å skrive inn verdien manuelt eller dra glider til ønsker verdi. Som representasjon av de forskjellige parameterne er det bygget inn en illustrasjonssketch. For å sikre at programmet kjører feilfritt et det lagt en øvre og nedre grense til alle parametere.



Figur 28: Parameterisering av kontravekt



Figur 29: Detailview av parameterinstillinger

Parametriseringen av kontravekt legger til rette for at massesenteret kan flyttes opp eller ned. Normalt ligger veivtappen inne med utfresing som vist på Figur 28. Målet med å implementere en geometrientring på denne måten gir mulighet til å ta bort og legge til utfresing på kontravekt. Dette kan da justeres ut ifra ønsket resultat. Er ikke bruker fornøyd med resultatet kan operasjonen gjøres på nytt med forskjellige tallverdier.

4.3.1. Initialiseringskript

GUI er brukergrensesnittet som styrer initialiseringskriptet. Funksjonen til dette skriptet er å starte de individuelle prosessene i riktig rekkefølge. I likhet med batch mode åpnes alle kommandovindu i silentmode. Ved bruk av dette skriptet åpnes ingen kommandovinduer, vist i Figur 30. Når skriptet kjøres i bakgrunnen, får ikke brukeren varsel om skriptene blir kjørt. Derfor er det lagt inn tekstbokser som viser om simuleringen har begynt og når den er ferdig. Da vet brukeren om skriptene kjøres eller ikke.

```
179        Dim runJournalInfo As New ProcessStartInfo(runJournalFile, runJournalArgs)
180        runJournalInfo.CreateNoWindow = True ' Gjør at skriptet kjøres i silentmode.
181        runJournalInfo.UseShellExecute = False
```

Figur 30: Kommando for kjøring i silent mode [7]

4.4. Controllab, 20-sim og pythonskript

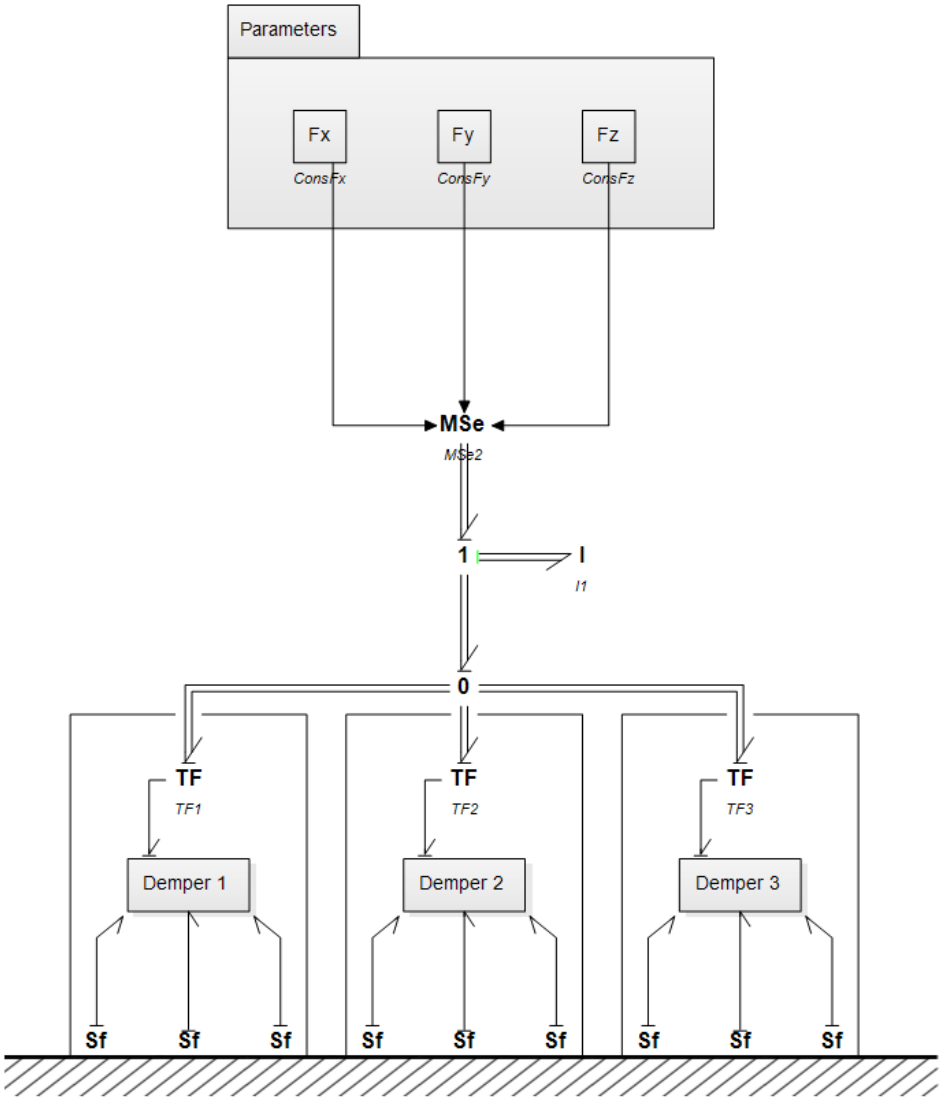
Skriptet i Python bruker Controllab-biblioteket som manuelt må installeres. Skriptet tar inn CSV for x- og y-krefter og konverterer disse til lister. Disse verdiene sender skriptet til 20-sim som en funksjon av tid. 20-sim mottar kreftene og erstatter parametrene y og alpha. Videre settes det opp en funksjon for å definere at hver kraft har sin respektive vinkel, og at alle kreftene og vinklene samsvarer med turtallet.

4.4.1. Modellering i 20-Sim

Utviklet modell bygger på Mass-spring-damper systemet, der modellen er sammensatt av krefter som virker inn i systemet, bindeledd og oppdeling av gummiben. Systemet er modellert så oversiktlig som mulig ved bruk av sub-modeller, for å begrense det totale bildet av systemet til en forståelig modell.

Modellen og mye av prosjektet baserer seg på forskningsartikkelen: Khalil. Abdelrahman et al. «Fault Detection in Flexible Beams Based on Output Only Measurements». In: 2020 American Control Conference (ACC). 2020[1] da artikkelen har en lignende problemstilling til prosjektet. Forskningsartikkelen har derimot vist seg å inneholde en del feil og gjorde at utviklingen krevde verifisering. Artikkelen er ikke bare vitenskapelig feil, men det er heller ikke mulig å gjenskape forskningen i etterkant. Utviklingsdelen av den matematiske modellen er gjort ut ifra feilsøking av denne artikkel. Oppgaven vil ha en egen seksjon der feil rettes opp. Metoden for modellering er ønsket av Sperre grunnet sin kompleksitet og introduksjon av ulinearitet. Ved å modellere på denne måten kan det ved videre utvikling introduseres utmattelse og sprekker, noe som kan gi Sperre en mer nøyaktig indikasjon på levetid. Det er også tatt i betraktning at dette ikke er den eneste tilnærmingen til å løse problemstillingen. Det er også vurdert med veileder om segmentene skal byttes ut med et fjærelement som baserer seg på å bruke en funksjon av k i motsetning til en konstant.

4.4.2. Overall-system



Figur 31: Total modell [7]

Det totale systemet, se Figur 31, anses som at bunn er innfestning og fast innspent der modellen er definert slik at hastigheten fra Sf er satt til 0. Modellen samler inn krefter fra X, Y og Z og sender disse til en MSe. Kreftene innehar et signalbånd som betyr at kausalitet ikke har en innvirkning her til tross for pil. MSe distribuerer kreftene i form at en matrisebond der 0-Junction samler alle krefter og fordeler likt til alle ben. TF transformerer fra matrisesignal til enkle krefter. Systemet er modellert med tanke på videre utvikling og ligger inne med X, Y og Z selvom kun X benyttes i modellen. Y- og Z-krefter defineres derfor som 0 og neglisjeres i TF. Modellen kan sammenlignes med en mappe på datamaskinen der flere mapper ligger i andre mapper. Dette er for å holde orden, men ikke minst for å kunne få en god oversikt over modellen. Det er fullt mulig å modellere alt i samme bilde, men det ville da blitt uoversiktlig og mange komponenter. Modellen er derfor modellert med submodeller som innehar hver sine respektive koblinger.

ConsFx henter en liste fra controllabskript som konverterer NX simuleringskrefter fra CSV til liste. Listen tas her inn og settes opp mot enda en liste. Denne listen har 72 linjer, der hver linje definerer 5 grader rotasjon. Ved å gjøre dette kan en kontinuerlig kraft for hver rotasjon sendes ned i systemet og simuleringen kan kjøres til gitt tid er nådd.

For å skape struktur i modellen er det modellert en parameterboks som inneholder forskjellige parameter, variabler og likninger. Ved å modellere på denne måten kan bruker alltid gå inn på samme sted for å endre verdier som benytter seg av "Real Global" parameter. Ved å endre en parameter i parameterboks sendes disse til modellen og

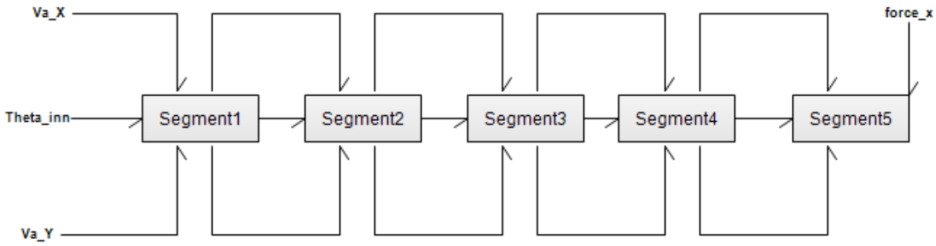
parameter kan lett endres. Dette gjør det også lettere å overskue og forstå modellen, og gjør det enklere å identifisere eventuelle feil eller problemområder. Ved å dele opp modellen i flere mindre deler, kan man også arbeide mer målrettet med spesifikke deler av modellen, noe som kan gjøre det enklere å utføre endringer eller justeringer.

TF fra $\begin{bmatrix} x & y & z \end{bmatrix}$ til x

```
parameters
        real r = 1;
equations
        p1.f[1] = r * p2.f;
        p1.f[2] = 0;
        p1.f[3] = 0;
        p2.e = r * p1.e[1];
```

[1] [2] [3] er i denne likningen [x] [y] [z] hvor det tydelig definerers at [2] og [3] neglisjeres

4.4.3. Oppdeling av fot

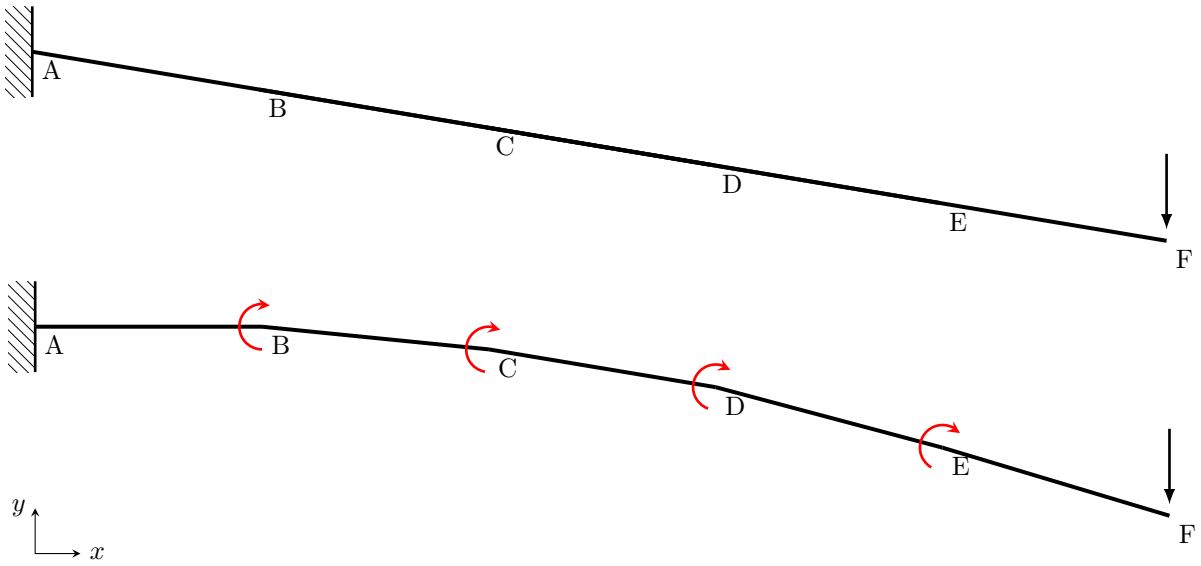


Figur 32: Segment oppdeling [7]

På innsiden av demperene fra hovedbildet som vist i Figur 31 finner vi en oppdeling av gummidemperen, der den deles opp i flere segment. Ved å gjøre dette innføres det ulinearitet i statikken. Målet med å gjøre dette er å kunne modellere en gummidemper som representerer virkeligheten best mulig. Figur 32 viser til 5 segment. Dersom flere segment introduseres vil resultatet bli mer nøyaktig, men simuleringen vil også ta betraktelig lengre tid.

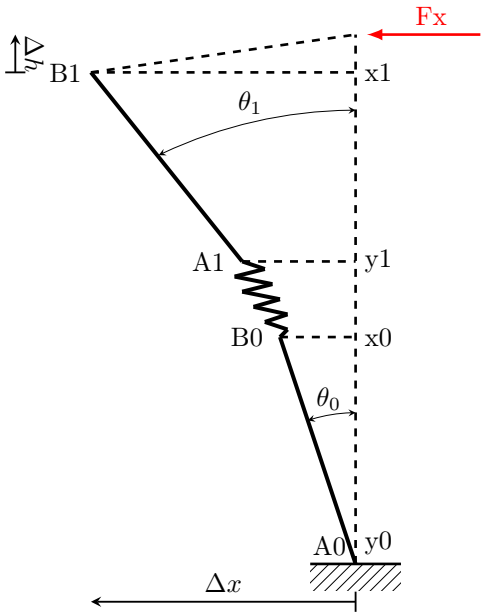
Måten gummidemperen er modellert, kan beskrives som en utkrager, hvor den faste innspenningen er i bakken og den frie enden er i innfestingen til kompressoren. Hvis gummidemperen hadde blitt modellert som en hel del, ville utbøyingen blitt lineær. En effektiv metode for å forhindre linearitet i bevegelsen er å dele opp en gummidemper i flere segmenter. Ved å inndele gummidemperen i mindre deler, skapes et mer komplekst og ikke-lineært bevegelsesmønster. Systemet er modellert ved at gummidemperen er oppdelt i 5 deler. Hvert segment representerer en spesifikk del av gummidemperen, hvor hver del er 11 mm før kompresjon. Ved å analysere bevegelsen og kreftene til hvert segment individuelt, gir dette en mer detaljert representasjon av det totale bevegelsesmønsteret til gummidemperen. Selv om bevegelsen til hvert segment er lineær, blir gummidemperen som helhet en ikke-lineær bevegelse. Dette skyldes at segmentene interagerer med hverandre, som skaper et mer komplekst bevegelsesmønster.

4.4.4. Introduksjon av segmentering av gummidemper



Figur 33: Introduksjon av segment i en utkrager [7]

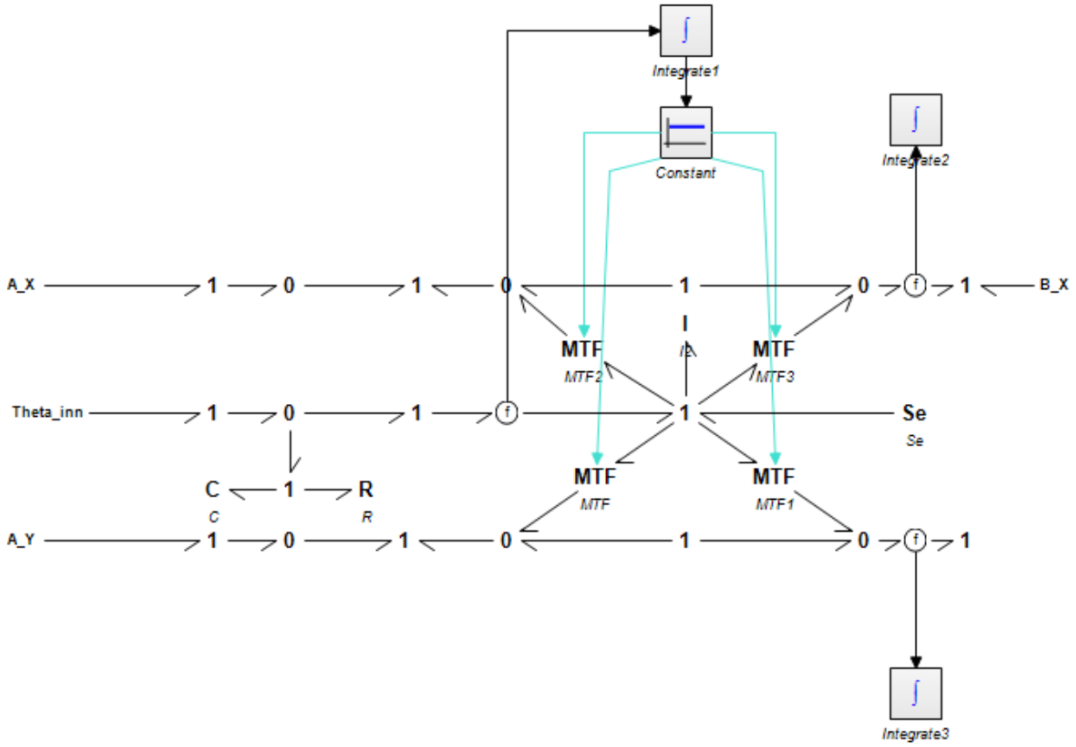
Figur 33 illustrerer to utkragere med kraft påført i den frie ende, der segmentering er introdusert til den siste. Figuren illustrerer forskjellen mellom en lineær og en ulineær utkrager. 20-sim modellen fungerer på sammen måte i den forstand at en kraft inntreffer i en ende og fast innspenning i den andre enden.



Figur 34: Gummidemper delt i 2 segment [7]

Figur 34 representerer 2 segment i en gummidemper hvor demping og fjær inntreffer mellom segmentene. θ_0 og θ_1 viser til endring i vinkel mellom de forskjellige segmentene der vinkelen θ_1 er vesentlig større en vinkelen θ_0.

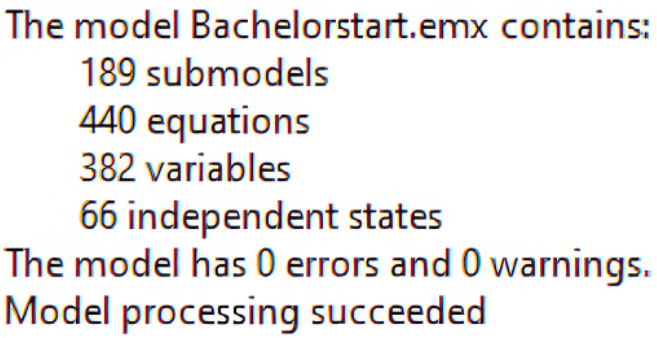
4.4.5. Segment



Figur 35: Individuelle segment [7]

Ved modelleringen av hvert enkelt segment fokuseres det på bøyedefleksjon mens aksial og skjærkrefter neglisjeres. Forholdet mellom fleksjonsformene etableres ved hjelp av MTF elementer, som her knyter kreftene inn mot utbøying basert på vinkelen.

Bond-graph modellen brukes til å koble sammen ulike segmenter og ledd av bjelken, der en 1-Junction brukes for å koble elementer med samme hastighet, og sørge for at den rettede summen av tilknyttede kraft er lik null. På samme måte brukes en 0-Junction for å koble elementer med samme kraft, og summere deres hastighet til null. B_X er en kraftkilde, som representerer inngangen der kraften virker på utkragerbjelken, og A_X og A_Y representerer en hastighetskilde til rotpunktet for hvert segment oppover.

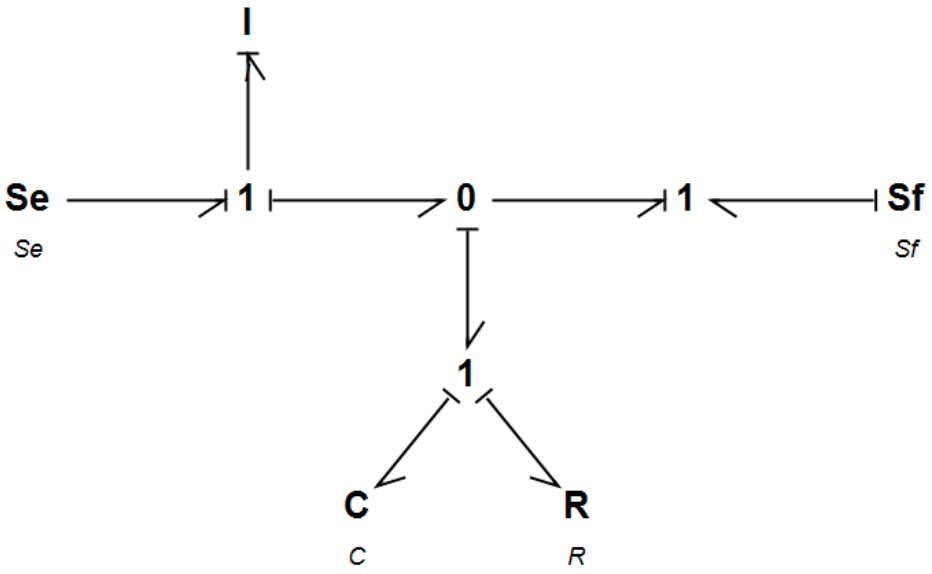


Figur 36: Modellprosessering [7]

For komplette, ferdigutviklede modellen viser Figur 36 til antall submodeller, likninger, variabler og uavhengige tilstander i modellen. Modellen viser null feil og varsler.

4.4.6. Versjon med fjærelement som funksjon

I en utvikling er det viktig å kunne vurdere andre alternativ dersom hovedideen frafaller. Det er Sperre som ønsker modell med segmentoppdeling og derfor er denne versjonen blitt prioritert. Det finnes likevel andre alternativ, som ikke nødvendigvis er like komplekse.



Figur 37: Mass-spring-damper system med fjærelement som en funksjon [7]

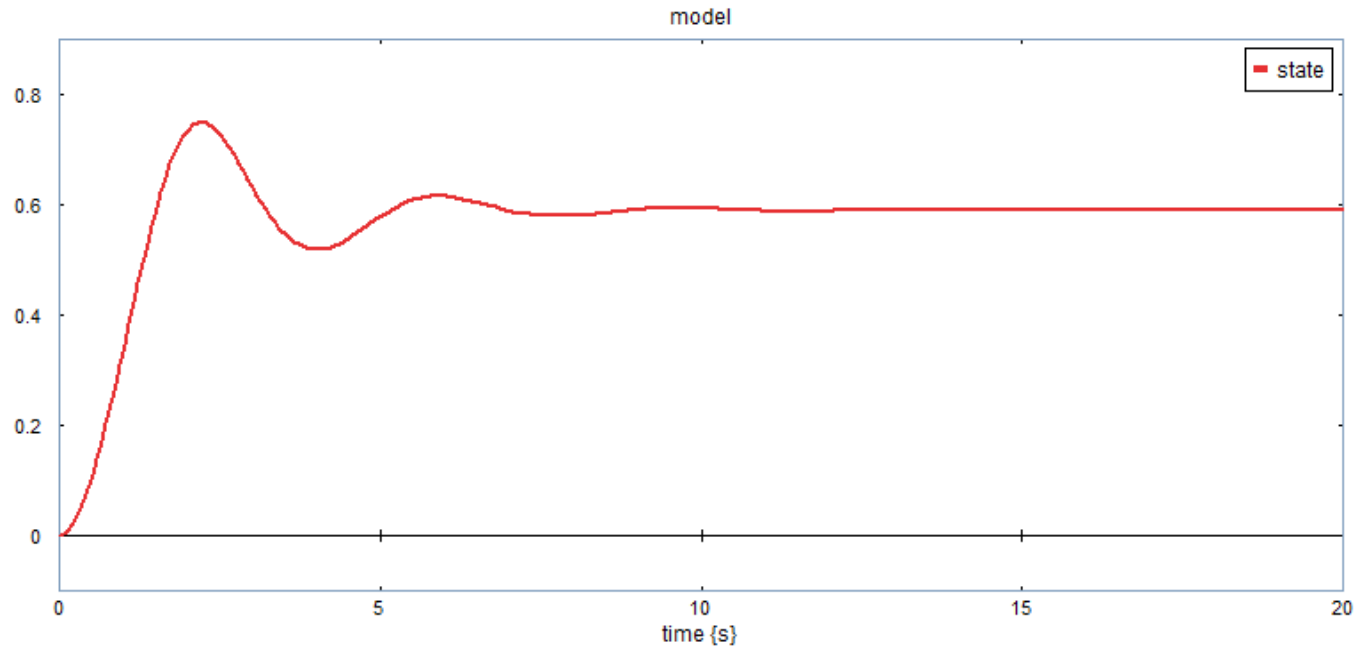
Alternativ 2, se Figur 37 er en modell som fungerer som en mass-spring-damper system der fjærelementet er definert som en funksjon. På denne måten kan det modelleres en ulineær fjær uten bruk av segment og komplekse system. Utviklet modell er forkortet og forenklet og fungerer som et alternativ og et bevis på at andre alternativ er mulige.

```
parameters
   real k1 = 1;
   real k2 = 2;

variables
   real dy;

equations

   state = int(p.f);
   dy = state;
   p.e = dy* k1 + dy^3 * k2;
```

Figur 38: Fjærelement som funksjon [7]

For å implementere ulinearitet i modellen kan formelen for p.e i C elementet endres. Istedenfor å bruke en lineær formel kan det inkluderes en ulineær formel som reflekterer systemets oppførsel på en annen måte. For å gjøre dette kan det benyttes forskjellige funksjoner. Det er i Figur 37 valgt en potensfunksjon som tar hensyn til en kubisk respons av fjæren hvor "dy" er endring i tilstanden, opphøyning i tredje representerer en kubisk respons og k1 og k2 som bestemmer styrken av den ulineære responsen.

Det er viktig å merke seg det her er det ikke gjort undersøkelser for å bestemme funksjonen i modellen og heller satt en vilkårlig funksjon som eksempel for å forklare virkemåten til en slik modell.

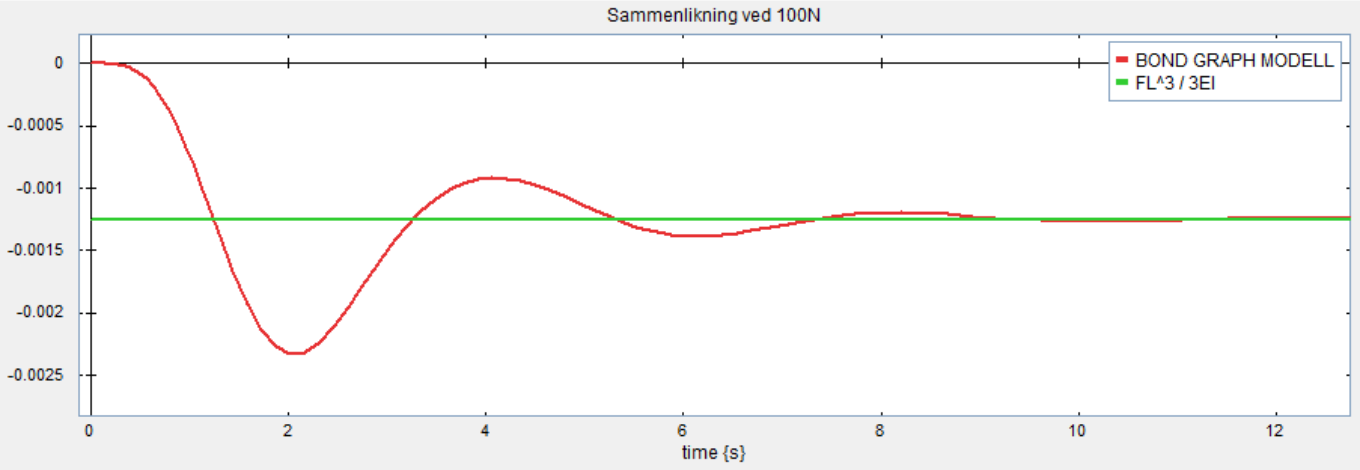


Figur 39: Eksempelgraf av modell med C som funksjon [7]

Figur 39 viser til tilstanden i inertia-element ved et satt scenario for det alternative systemet. Grafen minner godt om verdier som plottes i resultatdel, og anses som et anstendig alternativ til segmentmodellen. Det vil i fremtiden og ved videre utvikling bli Sperre sin oppgave å velge videre fremgangsmåte for modellen.

5. VERIFIKASJON AV MODELL

For å verifisere modellen mot virkelighet er statikk benyttet i sammenligning av en utkrager med fast innspenning i den ene enden og kraft som virker ned i den andre enden. For å kunne sammenligne at modellen sammenfaller med statikk er formelen for nedbøying av en utkrager plottet som graf der den tar inn svaret av likningen skrevet i parameter. Modellen bruker stivhet i gummidemperen for å avgjøre nedbøying og ikke den konvensjonelle e-modulen og annet arealmoment.



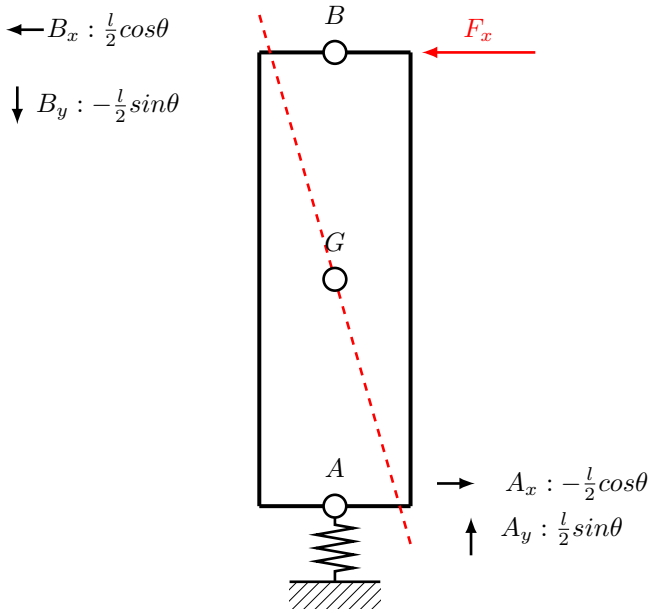
Figur 40: Sammenlikning modell og statikk [7]

Figur 40 viser til at statikk og Bond graph modell samsvarer med utbøying. Vi kan se en statisk utbøying fra statikken og en kurve med demping hos modellen. Dette er ved en gitt kraft på 100N. Grunnet kompleksiteten av modellen og mengden med tid disponert til oppgaven er det ikke nok tid til å fullverifisere hele Bond graph modellen. For videreutviklingens del er da verifikasjon et hovedpunkt før tilleggsfunksjoner implementeres. Dette betyr at modellen samsvarer med statikk ved 100N statisk kraft, men avviker når den dynamiske kraften brukes. På en annen side er det introdusert ulinearitet, noe som forvansker verifikasjonen. Som tidligere nevnt inneholdt forskingsartikkelen [1] som dannet grunnlaget for dette prosjektet flere feil. Dette førte til flere utfordringer og mye feilsøking innad i prosjektet. For en positiv kraft som virker inn mot et segment vil de forskjellige sidene oppleve forskjellige krefter basert på vinkelen de får. Som vist og verifisert i Figur 42 samsvarer ikke kreftene med verdiene fra artikkelen. Artikkelen viser til at V_a vil ha både en -cos og en -sin, men V_a vil bli $-\frac{l}{2}cos\theta$ og $\frac{l}{2}sin\theta$ respektivt ved et reelt scenario.

$$\vec{V}_{A/G} = \frac{l}{2}\dot{\theta}\begin{bmatrix} -\cos\theta & 0 & -\sin\theta \end{bmatrix}^{\mathrm{T}},$$
$$\vec{V}_{B/G} = \frac{l}{2}\dot{\theta}\begin{bmatrix} \cos\theta & 0 & -\sin\theta \end{bmatrix}^{\mathrm{T}},$$

Figur 41: Utdrag fra forskningsartikkel [1]

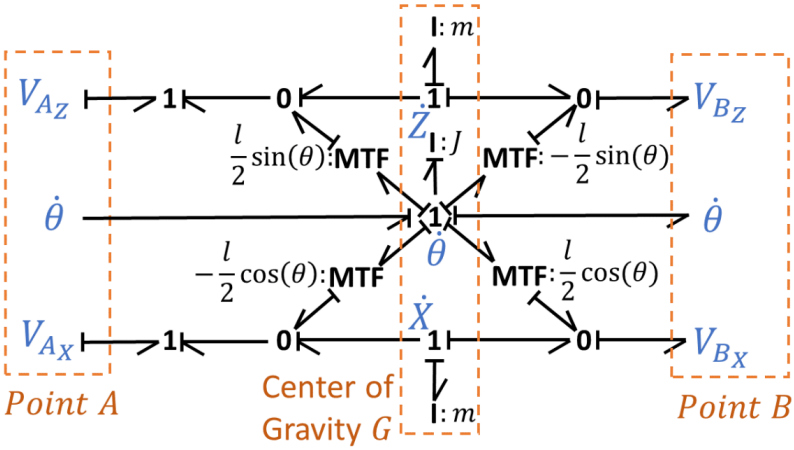
For å rette opp i feil fra forskningsartikkel [1] er det skissert opp en modell av et segment der en kraft Fx virker inn i systemet. Ax, Ay, Bx og By har piler som viser kraftretningen i punktene under kondisjon Fx. Det er her utregnet tilhørende cos og sin for hvert tilhørende punkt.



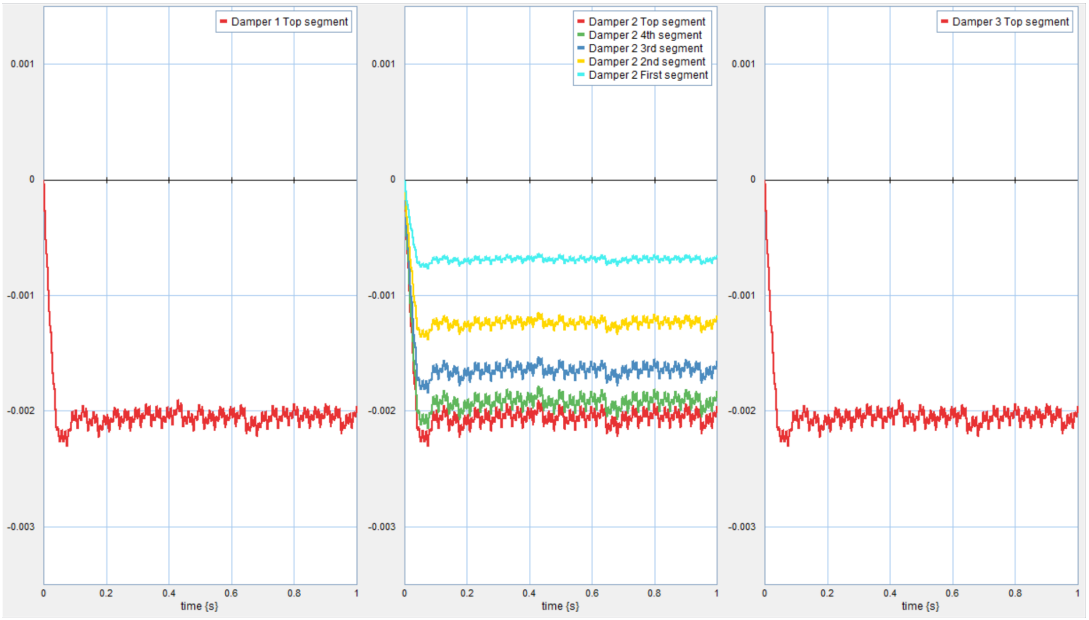
Figur 42: Illustrert virkemåte av MTF

Figur 42 viser til hvordan MTF jobber og hvordan segmentene er tiltenkt med omgjøring av kraften ut ifra hvordan segmentet vil bevege seg. Rød stiplet linje visualiserer utbøyingen av dette enkle segmentet som den lineære delen hvert segment vil ha. Figur 42 benytter notasjon A, B og G der A er ende mot innspenning og B er der kraft inntreffer. G er senteret av gravitasjon i ett spesifikt segment. Figuren er tiltenkt som et tertiærsegment da det er det siste elementet koblet i rekken. Denne tilnærmingen fungerer veldig godt ved en statisk kraft, men kreftene fra NX er derimot dynamiske og varierer mellom (+) og (-).

Modellen fra artikkelen er modellert slik at interia i midten av krysset integreres for å finne vinkelhastighet, dette er riktig. Modellen har også inertia der det er nødvendig å differensiere. Grunnen til at disse må differensieres og ikke integreres er fordi kausaliteten i modellen ikke tillater det. Se Figur 43, dette skaper en algebraisk løkke per inertia element. Dette gjør at modellen blir veldig treg å kjøre da modellen venter på en variabel som ikke enda er regnet ut. Ved å kjøre 5 segment vil det bli 2 algebraiske løkker per segment, og modellen vil få 10 algebraiske løkker. I oppgavens modell er derfor disse inertia-elementene neglisjert, da det kun er treghet disse legger til, som kun påvirker dempingen. Siden elementene er oppsatt med vekt per segment blir ikke vekten mer enn 200g og er derfor uten betydning. Ved å fjerne disse elementene har modellen mulighet til å kjøre fritt uten en sperre. Modellen er også tegnet med andre verdier enn tidligere oppgitt.

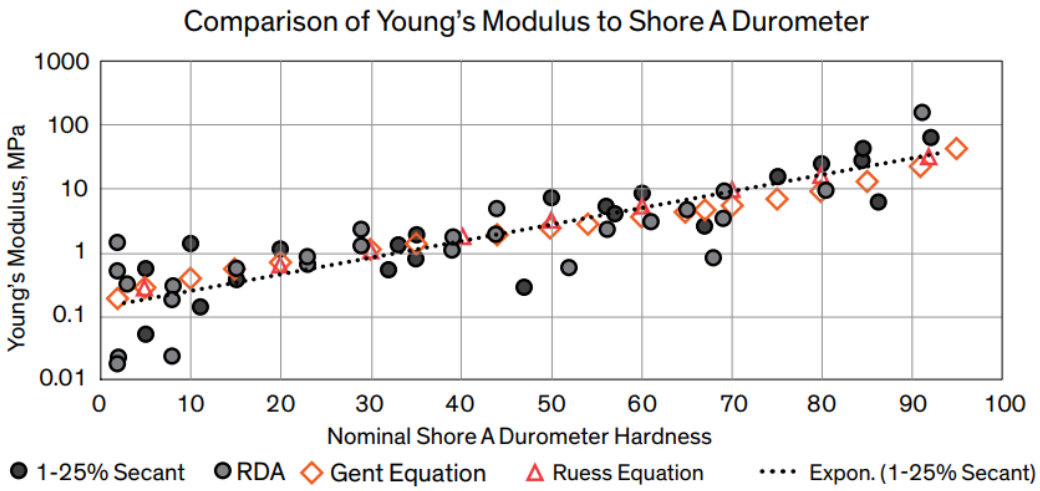


Figur 43: Utdrag fra forskningsartikkel [1]



Figur 44: Verifikasjon at hver demper tar opp lik vibrasjon [7]

For å verifisere at alle ben mottar like krefter er det satt opp en simulering med 3 vinduer som hvert representerer en demper. Figur 44, viser til at alle tre dempere tar opp like mye vibrasjon. Demper 1 og 3 er definert med kun toppsegment mens demper 2 har en oppdeling i 5. Den røde grafen viser til at alle segment har en maks utbøying lik hverandre.



Figur 45: Utdrag fra artikkel [9] E-modul av gummidemper fra hardhetstest for Shore A 55

Grafen i Figur 45, viser til at E-modul hvor gummien fra hardhetstesten i artikkel [9] ligger mellom 1 og 10 MPa. Grunnet at E-modulen ikke er oppgitt i Sperres dempere måtte dette regnes ut for hånd for å kunne sammenligne en utbøying av en utkrager mot modellen. Formelen for nedbøying kan snus ved å innføre stivheten til demperen.

Formel for nedbøying av en utkrager der kraften treffer i endepunktet:

$$\delta_{max} = \frac{FL^3}{3EI}$$

Benytter formel med hensyn på K.

$$K = \frac{3EI}{L^3}$$

- Der variablene i likningen er:
  - δ_{max} er nedbøying
  - K er fjærstivhet
  - L er lengden på utkrager
  - F er er kraften som virker ned på utkrager
  - E er E-modul for 55ShA
  - I er annet arealmoment
- Der alle variabler er kjent uten om E-modul

Snur formel til fordel for E:

$$E = K\frac{L^3}{3I}$$

$$E = 80\frac{55^3}{3\frac{\pi D^4}{64}}$$

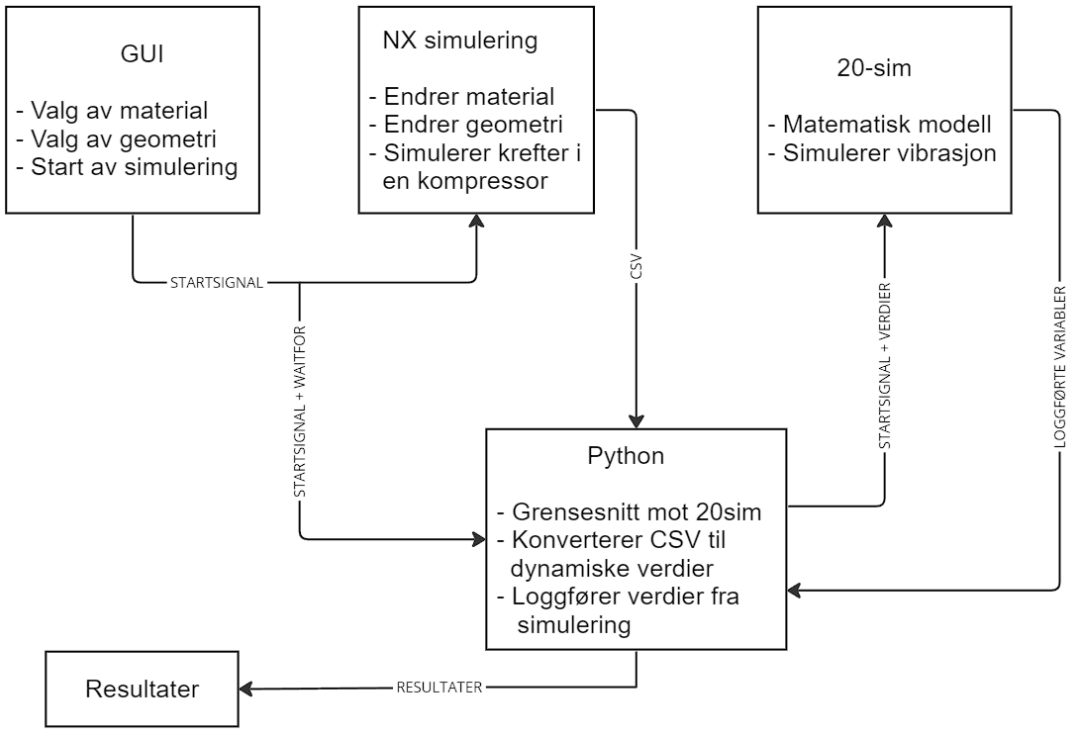
$$E = 80\frac{166375}{3 * 1553155}$$

$$E = 2.8565 \approx 2.86MPa$$

Som Figur 45 viser, treffer utregnet E-Modul på grafen.

6. RESULTATER

6.1. Resultat av utvikling



Figur 46: Flytskjema for utviklet system [7]

Som vist i Figur 46 inneholder systemet forskjellige prosesser der alle har sitt tilhørende grensesnitt. Dette krever flere programmeringsspråk for å knyte alle prosessene til et sammenhengende system. GUI og NX benytter Visual Basic siden dette samarbeider godt med grensesnittet til NX Open. For 20-sim er det gunstig å skrive i Python eller Octave, og Python har i dette tilfellet blitt valgt. Prosessforløpet i systemet er bygget opp med en GUI som styrer prosessene og simuleringen ved å gi brukeren et oversiktlig brukergrensesnitt. GUI fungerer som en styringsplattform for alle programmene der den sender startsignal som iverksetter programmenes respektive oppgaver. Gjennom dette forløpet vil programmene kommunisere og sende verdier videre til neste prosess i systemet. Alle verdier fra NX konverteres fra CSV til lister. Disse verdiene konverteres i Python og sendes til 20-sim. 20-sim tar for seg vibrasjonssimuleringen og Python loggfører dataene fra simuleringen. Denne dataen konverteres tilbake til CSV før den presenteres for brukeren.

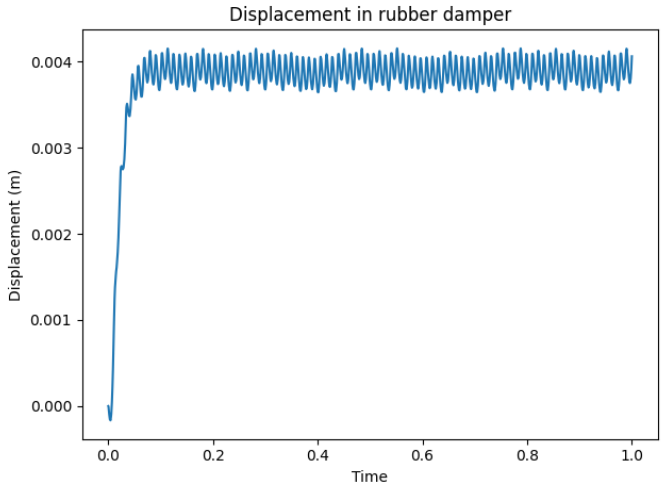
Programvaren fungerer ved at bruker først må importere NXs lokale materialliste, "Physicalmateriallibrary.xml". Listen med material sendes videre til en rullegardinsliste hvor bruker kan velge ønsket material. Deretter kan bruker velge geometrien i utfresingen på kontravekt, dette blir valgt ved å enten bruke glidere eller ved å skrive det inn manuelt. Dersom det ønskes å tilbakestille verdiene til standardmål kan knappen "Reset dimensions" benyttes. For driftsikkerhet er det satt en maksimal- og minimalverdi på utfresingens parameteriserte mål. Etter valgt materiale og geometri kan simuleringen kjøres. Simuleringen starter med å kjøre et skript som endrer materialet på stemplene, deretter et skript som endrer geometrien på kontravekten. Når disse skriptene er kjørt så startes en lastsimulering av krefter under kompressordrift i Simcenter3D. Verdiene som ble lagret fra NX simuleringen blir da oversatt til lister, og sendt til 20-sim hvor det skjer en vibrasjonssimulering gjennom den matematiske modellen med de nye verdiene. Resultatet blir så presentert grafisk og lagret numerisk.

I dette prosjektet har det blitt utviklet et verktøy som automatiserer forskjellige operasjoner for å vise hvilken påvirkning materialendring eller geometriendring har for vibrasjoner i gummidemperene til XA-150. Programforløpet inneholder parameterisering av verdier, endring av material, et simuleringsscenario for kompressoren, en whitebox som samler alle skript og kjører alle simuleringer og den matematiske modellen for simulering av vibrasjon. Programmet visualiseres via en GUI som gir brukeren mulighet til å kjøre alle simuleringer fra samme plattform. Programmet mottar og sender filer på tvers av av de ulike programmene.

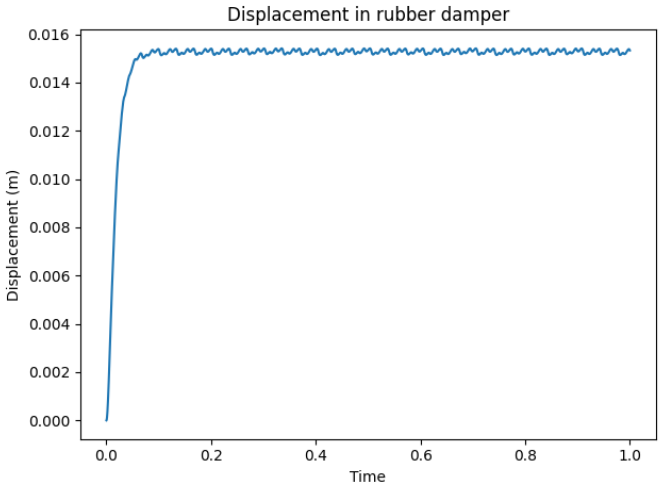
6.2. Resultat fra simulering

Resultatdelen gir innblikk i resultat på tvers av forskjellige konfigurasjoner i kompressoren. Det fremgår simuleringer der materiale i stemplene endres, endring i massesenter i kontravekten og en simulering som representerer en normal XA-150. Vibrasjoner simuleres i toppen av gummidemperene for å gi en indikasjon på vibrasjon i innfestningen av kompressoren, og presenteres i grafene som vibrasjon i x-retning.

6.2.1. Materialendring av stempel



Figur 47: Simulering av vibrasjon ved referansemateriale, Aluminum 2014

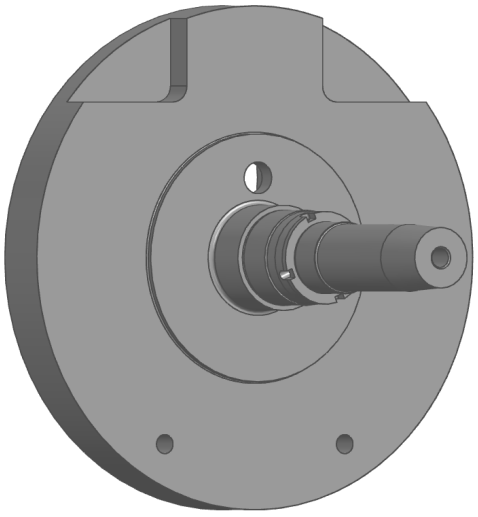


Figur 48: Simulering av vibrasjon ved endring av stempelmateriale til AISI Steel 410

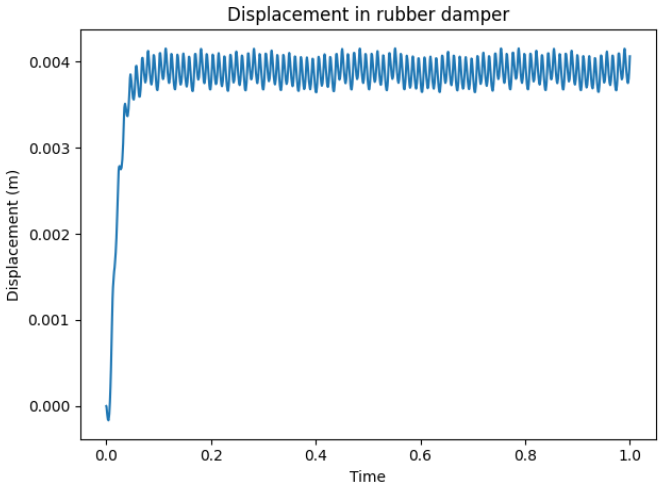
Figur 47 og 48 viser til en simulering der stemplene er aluminium og en simulering der stemplene er stål. Stål er her valgt for å illustrere forskjellene i vibrasjon ved materialendring.

6.2.2. Geometriendring av kontravekt

Geometriendringen tar for seg 3 forskjellige scenario der materiale står utrørt fra det opprinnelige, det vil si at stemplene er i aluminium for alle simuleringene, men at geometri av kontravekten varierer. De forskjellige scenarioene som presenteres er en normal kontravekt for referanse, en kontravekt der materiale er fjernet og en der materiale er lagt til. I tillegg er det inkludert et scenario hvor deler av kontravekten er fjernet.

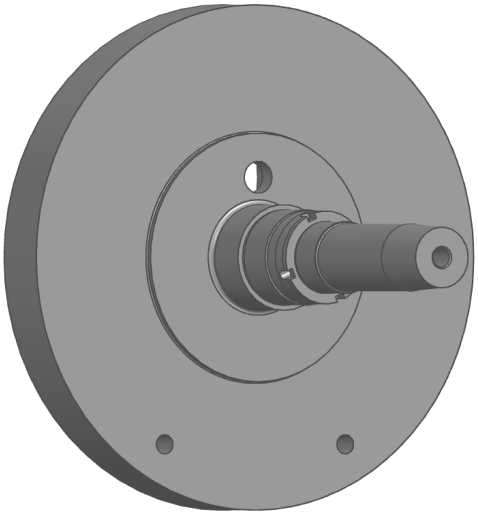


Figur 49: Referansegeometri

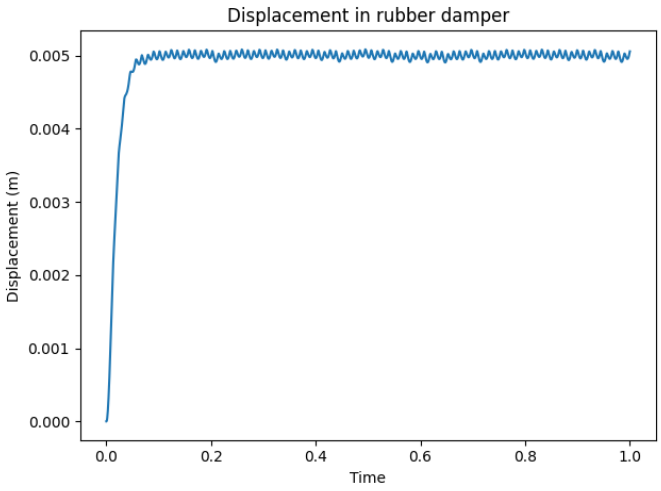


Figur 50: Simulering av vibrasjon tilhørende Figur 49

Figur 49 viser til referansegeometri og den tilhørende vibrasjonskurven.

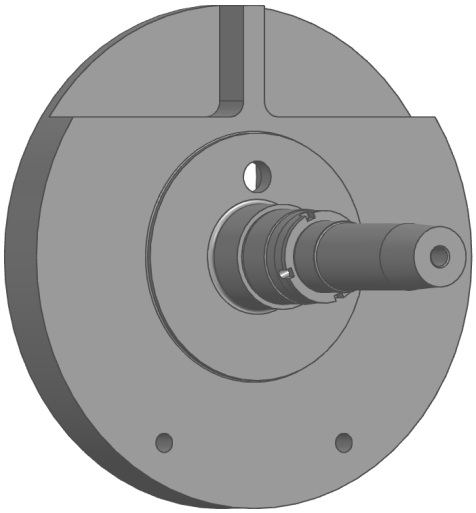


Figur 51: Geometri med lagt til materiale

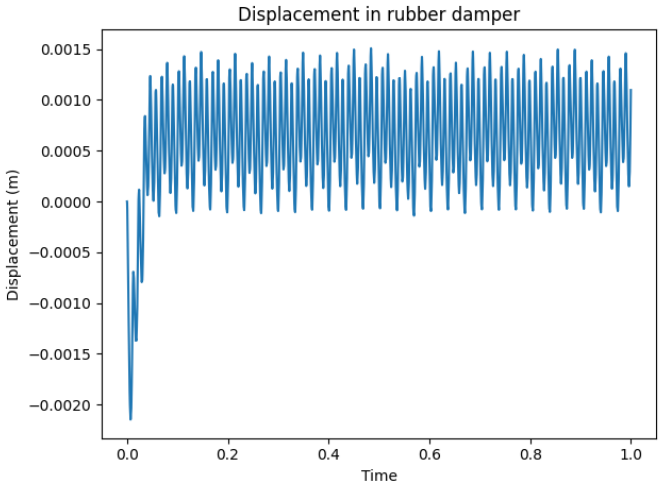


Figur 52: Simulering av vibrasjon tilhørende Figur 51

Figur 51 viser et scenario hvor materiale legges til. I denne geometrien, viser grafen at det vil forekomme en mindre vibrasjon men høyere utbøying enn i referansegeometri.

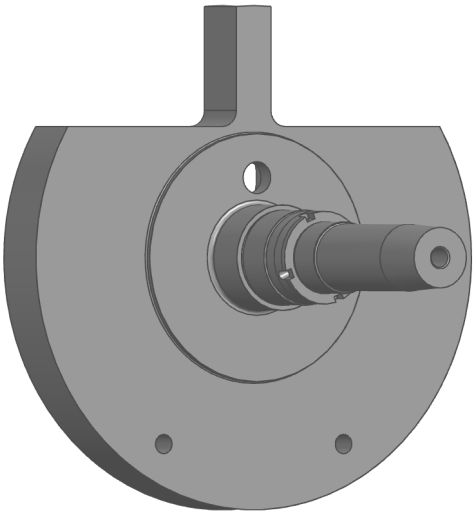


Figur 53: Geometri med fjernet materiale

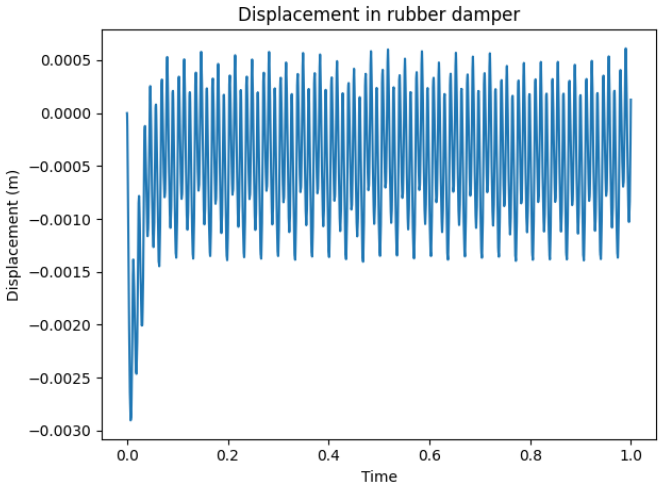


Figur 54: Simulering av vibrasjon tilhørende Figur 53

Figur 53 viser et scenario hvor utfresingen er økt, og at materiale er fjernet. Dette senker massesenteret på kontravekten. I dette scenariet forekommer det lite utbøying, men mye vibrasjon.



Figur 55: Scenario med maksimal geometriendring



Figur 56: Simulering av vibrasjon tilhørende Figur 55

Den siste kontravekten som får geometriendring tar for seg et scenario for å vise det ytre spekteret av parameteriseringen. I denne modellen er hele bakplaten fra toppen fjernet og det står kun en tapp igjen. Grafen viser at utbøying vil skje begge retninger i det horisontale plan, og at vibrasjonen øker betraktelig.

7. DISKUSJON

7.1. Diskusjon av resultat

For prosessens første resultat, materialendring, ble materialet i stemplene endret fra Aluminium 2014 til AISI Steel 410. Grunnet at AISI Steel 410 har høyere egenvekt enn Aluminium 2014 er det her forventet en høyere utbøying, og en høyere vibrasjonsamplitude. Dette da som et resultat av mer vekt som beveger seg hver gang kompressorens veiv roterer. Da denne vekten ligger langt fra massesenteret er det her da naturlig å anta at dette vil forårsake mer vibrasjon. I resultatene kommer det tydelig frem at materialendring har en reell effekt på vibrasjonen som kompressoren produserer. I Figur 48 vises det til en betydelig økning i utbøying, men reduksjon i vibrasjon. Figur 47 viser vibrasjonen ved materialvalg av slik kompressorene til Sperre er oppbygd. Grafen for aluminium viser her en urytmisk svingning med korte perioder og høy amplitude. I grafen for stål, er det en høyere utbøying med mindre amplitude og lengre perioder. For å definere hvilke resultat som er ettertraktet i en Sperre kompressor må en rekke andre faktorer her vurderes, som resonans og eventuell endring av dempingskoeffeisient.

En viktig faktor når vibrasjon skal analyseres, er omgivelsenes egenfrekvens. Hvis frekvensen fra kompressorens vibrasjon sammenfaller med de nærliggende systemenes egenfrekvens, kan omgivelsene ta til seg energi fra disse vibrasjonene. Dette kan igjen føre til økt vibrasjon, og stor slitasje på omgivelsene.

Da det i resultatet for materialendring viser en reduksjon i vibrasjon når egenvekten til stemplene øker, kan man da anta at kontravektens bidrag til vibrasjon også vil reduseres når dets egenvekt øker på samme side som stemplene. Hvis det antas at massesenteret til systemet ligger mellom veivtappens senter og stempler, vil en reduksjon av masse i dette området flytte massesenteret vekk fra reduksjonsområdet, som igjen vil øke vibrasjonen.

I resultatene for geometriendring viser Figur 50 kompressorens opprinnelige kontravekt. Her ligger forskyvningen på ca. 3.8mm, og bølgehøyden på ca 0.5mm. Videre på figur 52 ser man derimot at material er lagt til, og utbøyingen blir høyere, men vibrasjonen blir vesentlig mindre. Ved å legge til materiale på kontravekten gir resultatene ca. 5mm utbøying i positiv x-retning. Grunnet at systemet kun tar for seg de mekaniske forholdene er det naturlig å anta at størrelsen på lavtrykk- og høytrykkstempel har en innvirkning på den initielle utbøyingen. Det kommer her frem en tydelig trend i Figur 54 og Figur 56, der et resultat av fjerning av materiale gir større bølgehøyde for vibrasjonen i simuleringsgrafene.

Ved utvikling av nye eller eksisterende deler til kompressoren vil andre faktorer påvirke valget utenom material og geometriendring. Den mest anvendelige av disse metodene kan derfor være geometriendring, da det vil føre til en relativt liten endring i Sperres bestillinger samt produksjonslinje. En geometriendring vil kunne maskineres i allerede eksisterende deler, som ikke vil utføre noe endring på systemets holdbarhet eller duktilitet. Skal det utføres en materialendring i stemplene vil en rekke andre faktorer enn egenvekt stå ovenfor forandring som hardhet, duktilitet og varmeoverføring. Dette vil påvirke kompressorens andre egenskaper som varmeutvikling, levetid, og støyforurensing. Ekstrakostnader fra leverandør kan også forekomme om stempler skal bestilles i et ukonvensjonelt material og det vil derfor anbefales av gruppen å velge geometriendring om en av metodene skal implementeres i Sperres nåværende produkter.

Ved å endre material i stempler eller geometri på kontravekt kan kompressorens indre krefter endres. Disse nye kreftene kan danne et nytt vibrasjonsscenario og vil gi en annen belastning på gummidemperene. Denne belastningen kan føre til at gummidemperene må byttes ut til et hardere material eller medføre behov for alternative vibrasjonsisolatorer. Hardere material kan være nødvendig for å oppnå ønsket demping, men kan også medføre en dårligere isoleringsevne. Med dårligere isoleringsevne vil mer av vibrasjonen bli absorbert av underlaget kompressoren står på. Dette kan over tid føre til skader på nærliggende maskiner eller annet utstyr. Gummidempere vil også bli hardere over tid, både gjennom slitasje og gjennom varmeutvikling under bruk. Gummidempere er derimot billige, effektive isolatorer, som med en varierende hardhetskatalog gjør de til et godt valg for vibrasjonsisolering. Et alternativ til gummidempere, kan være fjærisolatorer. Ved bruk av denne type isolator er kompressoren adskilt fra underlaget ved bruk av en samling stive fjær. Levetiden til en slik isolator vil også overgå en konvensjonell gummidemper, men prisen vil være vesentlig dyrere enn gummidempere. Det kan her da vurderes om reduksjonen i vibrasjon på omgivelsene kan føre til økonomisk gevinst for kunden over tid.

Resultatene gir svar på undersøkelsene prosjektet baserer seg på, men inneholder et par svakheter. Grunnet tiden avsatt til prosjektet er det ikke nok tid til å verifisere alle prosesser. Dette betyr at den matematiske modellen

som er svært kompleks, ikke er verifisert for alle scenario eller for dynamisk kraft. Simuleringen gir derimot et resultat per simulering som avviker fra hverandre og som kan gi en pekepinn på om justeringen er hensiktsmessig.

7.2. Diskusjon av prosess

Prosjektet startet med å sette opp en struktur for rapporten, sette seg inn i nødvendige standarder og lage en plan for utviklingen. I dette prosjektet startet studentene tidlig med å bruke store deler av tidsrammen på å utvikle programvaren før studentene begynte å skrive. Det ble brukt mye tid på å simplifisere kodene slik at de enkelt kan endres i fremtiden ved nye behov eller forbedringer. Programmet ble utviklet etter metoden MVP, hvor det tidlig i prosjektet ble fremstilt flere utkast av programvaren for å effektivt starte testing og identifisere potensielle fallgruver.

I starten av arbeidet ble det tidlig konkludert med at skriptene, samt GUIen skulle skrives i Visual Basic. Dette viste seg svært effektivt da dette kodespråket enkelt kommuniserer med NX, og ressurser er godt tilgjengelig. Har man likevel god forståelse for NX og et annet kodespråk, er det likevel fullt mulig å bruke skript skrevet i Python, C# eller andre mer konvensjonelle språk. Om programmet senere skal videreutvikles krever det derfor at man her er inneforstått med språket Visual Basic. Hadde det her vært utviklet i et annet språk, ville videreutvikling vært enklere, men oppsettet i denne oppgaven ville blitt vesentlig vanskeligere. Alle skript ble derfor skrevet i Visual Basic, med unntak av skriptet for 20-sim som ble skrevet i Python da det kun var dette og Octave som støttet Controllab.

Siemens NX har vært en viktig brikke i utvikling av denne oppgaven. Både for utvikling av journals og skript, til de inkluderte materialbibliotekene, og simuleringen i Simcenter3D. Det finnes lignende programmer, som Inventor og Solidworks som innehar mange av de samme funksjonene, men da Sperre og NTNU bruker Siemens NX ble det konkludert med at dette var det beste valget for oppgaven.

Simcenter3D har fungert godt og effektivt for simulering av krefter innad i kompressoren. Det ble likevel brukt mye tid på definering av simuleringsmodellen da ble bygget, da NXs egne ressurser på temaet er begrenset. Med korrekt modell, og effektive skript arbeider NX både raskt og nøyaktig. Simcenter3D har derimot også en egen funksjon for simulering av vibrasjon. Denne funksjonen ble ikke utforsket i dette prosjektet, da det i premissene for oppgavene var bestemt at vibrasjonssimuleringen skulle utvikles i 20-sim.

20-sim har også fungert godt i oppgaven selvom det er et snevert program, med få tilgjengelige manualer og tilhørende informasjon. Programmet baserer seg på matematiske modeller, og lar brukeren i stor grad definere med verdier og begrensninger. På denne måten kan det bygges et komplett system tilnærmet virkeligheten.

Prosjektet møtte sin motstand i form av feil i forskningsartikkel, rekonstruering av joints i 3D modeller og et generelt stadie av verifikasjon av utviklede deler. Simuleringene er kjørt utallige ganger for å kontrollere at programmet fungerte som det skulle. Under disse eksperimentene ble det bemerket at koden var treg og hadde behov for optimalisering. Kodene ble derfor snevret inn og optimalisert. Den matematiske modellen er oppgavens største del da Bond Graph modeller tilhører fag på master-nivå. Studentene har likevel satt seg inn i dette for å utvikle et program som tilfredsstiller Sperres ønsker. Studentene har under sitt 3 årige bachelorstudium heller ikke hatt mye koding, men så seg nødt til å sette seg inn i dette da mesteparten av oppgaven omhandler programmering.

Det ble kortet inn på koden i skriptene som gjorde at programmet kjørte raskere. Det ble det lagt inn at programmet skal kjøre alt i silent mode noe som gjør at det ikke kommer opp kommandovinduer for hver prosess. Dette var midlertidig ikke mulig å gjøre med den matematiske modellen, selv etter kontakt med utviklerene til 20-sim. Spesielt var det modellen i 20-sim som ble forenklet med å fjerne 2 Inertia element i hvert segment for å bli kvitt 10 algebraiske løkker. De andre skriptene ble også kortet inn med ca. 150 linjer per skript, noe som gjør at programmet slipper å lese og kjøre unødvendige linjer. Ved å fjerne disse kodelinjene blir hele koden forenklet og optimalisert for fremtidig utvikling.

Det naturlige skrittet å ta videre, er å verifisere alle prosesser og implementere nye og ønskede funksjoner. Programmet tar kun for seg de roterende delene i kompressoren, og derfor burde resterende deler av kompressoren introduseres i fremtiden. Videre funksjoner som da kan introduseres i simuleringen er kompresjon, termodynamikk, moment fra motor, respektiv plassering av gummidempere og lager.

8. KONKLUSJON

Oppgaven fokuserer på utvikling av et verktøy som evner å kjøre flere simuleringsoperasjoner i samme program for å effektivisere prosesser og korte ned tidsbruk innen videreutvikling av kompressorer. Programmet er automatisert, intuitivt og kan integreres med eksisterende programvare. Det er lagt stort fokus på å bygge et robust program med et ukomplisert og oversiktlig brukergrensesnitt. Ved å bruke dette programmet har bedriften mulighet til å spare tid og kostnader ved utvikling av nye deler. Prosjektet er også utviklet spesielt med tanke om at programmet enkelt skal kunne bygges videre på, og fungere som en solid grunnmur for videreutvikling om dette skulle ønskes.

Oppgavetekstens hovedpunkter har gjennom hele prosjektet vært i fokus, og alle funksjoner som er utviklet er basert på disse punktene. Programmet, de autonome oppgavene, overføringene av filer, simuleringene, og resultatene lagres, kjøres, og fremstår i tråd med de premisser og ønsker som ble uttrykket for programmet av bedriften. Prosjektet innfrir alle punkter slik de presiseres i oppgaveteksten, og prosjektet konkluderes som godt gjennomført i henhold til de kravene som ble satt for gruppen og prosjektet av skolen og bedriften.

8.1. Conclusion in english

The task focuses on the development of a tool capable of running multiple simulation operations within the same program to streamline processes and reduce time spent on the further development of compressors. The program is automated, intuitive, and can be integrated with existing software. Great emphasis has been placed on building a robust program with a straightforward and clear user interface. By using this program, the company has the opportunity to save time and costs in the development of new parts. The project has also been specifically designed to allow for easy expansion and serve as a solid foundation for further development, if desired.

Throughout the project, the main points of the task description have been the central focus, and all the developed functions are based on these points. The program, the autonomous tasks, file transfers, simulations, and results are stored, executed, and presented in line with the premises and desires expressed by the company for the program. The project fulfills all the specified points as outlined in the task description, and it is concluded that the project has been well-executed according to the requirements set by the school and the company.

Referanser

[1] Khalil. Abdelrahman, Khaled F. Aljanaideh, Geoff Rideout, and Mohammad Al Janaideh. «Fault Detection in Flexible Beams Based on Output Only Measurements». In: *2020 American Control Conference (ACC)*. 2020, pp. 5034–5039. DOI: 10.23919/ACC45564.2020.9147301.

[2] A. Terry Bahill, B. Bentz, and F. Dean. «Discovering System Requirements». In: (Jan. 2009).

[3] Det Norske Veritas. *Vibration Class side 7*. 2020.

[4] Adeel Ehsan, Mohammed Ahmad M. E. Abuhaliqa, Cagatay Catal, and Deepti Mishra. «RESTful API Testing Methodologies: Rationale, Challenges, and Solution Directions». In: *Applied Sciences* 12.9 (2022). ISSN: 2076-3417. DOI: 10.3390/app12094369. URL: https://www.mdpi.com/2076-3417/12/9/4369.

[5] *Getting Started with NX Open*. URL: https://docs.plm.automation.siemens.com/data_services/resources/nx/1872/nx_api/common/en_US/graphics/fileLibrary/nx/nxopen/NXOpen_Getting_Started.pdf (visited on 03/21/2023).

[6] Jeffrey O. Grady. «Proving the Design Solution Satisfies the Requirements». In: (May 2016), p. 414.

[7] *Intern data*. URL: N/A.

[8] Hameed Lafta and Hazzi Akram. «Nonlinear Torsional Vibration Analysis Of Variable Inertia Reciprocating Engines». In: *Journal of Multidisciplinary Engineering Science and Technology* 3 (Jan. 2016), pp. 3159–3199.

[9] Kent Larson. *Can You Estimate Modulus From Durometer Hardness for Silicones? Yes, but only roughly ... and you must choose your modulus carefully!* Sept. 2017.

[10] *Learn About Damped Oscillations | Chegg.com*. URL: https://www.chegg.com/learn/physics/introduction-to-physics/damped-oscillations (visited on 01/12/2023).

[11] *Logo - NTNU*. URL: https://i.ntnu.no/logo-og-maler (visited on 01/11/2023).

[12] Fengxia Lyu, Caiqian Xie, Fengfeng Bie, Xinting Miao, Yifan Wu, and Ying Zhang. «Nonlinear Vibration Feature Recognition Method for Reciprocating Compressor Cylinder Based on VMD-Multifractal Spectrum». In: *Shock and Vibration* 2023 (Jan. 2023), pp. 1–15. DOI: 10.1155/2023/2504170.

[13] Azad Madni and Michael Sievers. «System of Systems Integration: Key Considerations and Challenges». In: *Systems Engineering* 17 (Sept. 2014). DOI: 10.1002/sys.21272.

[14] Ehsan Mollasalehi, Seyed Jahromi, Mario Forcinito, Wei Hu, and Qiao Sun. «Structural Fault Diagnosis of a Reciprocating Compressor Using Bond Graph Approach». In: vol. 1. Aug. 2012. DOI: 10.1115/DETC2012-71191.

[15] Alfredo Morillo, Javã Pedreira, Paulo Kurka, and Marco Bittencourt. *Modeling and Simulation of Torsional Vibrations in Two-Stage Reciprocating Compressors Driven by Electrical Motor*. Mar. 2017.

[16] Norelem. *Technical information for rubber bufers, Guide values for static load (excerpt from 26100, 26102, 26104 and 26106)*. 2020.

[17] *NX Journaling*. URL: https://www.nxjournaling.com/ (visited on 03/21/2023).

[18] Enaiyat Ghani Ovy. «Machine Dynamics Modeling by a Bond Graph Approach for Condition Monitoring». PhD thesis. Jan. 2017.

[19] Shengshan Pan, Muzhou Zhao, Bassem Andrawes, Hang Zhao, and Lian Li. «Compressive behavior of cylindrical rubber buffer confined with fiber reinforced polymer». In: *Journal of Low Frequency Noise, Vibration and Active Control* 39 (July 2018), p. 146134841878357. DOI: 10.1177/1461348418783570.

[20] E. Pedersen and H. Engja. *Mathematical Modelling and Simulation of Physical Systems*. Trondheim: Norwegian University of Science and Technology, 2014.

[21] M.Ravi Reddy, Jayakumar Vijayarangan, D. Santhoshkumar, and Muniappan Appu Samy. «Vibration Testing of Novel Engine Mount: Technical Note». In: *International Journal of Vehicle Structures and Systems* 10 (Dec. 2018). DOI: 10.4273/ijvss.10.6.09.

[22] Prabhu Shankar, Joshua Summers, and Keith Phelan. «A verification and validation planning method to address change propagation effects in engineering design and manufacturing». In: *Concurrent Engineering* 25 (Feb. 2017), p. 1063293X1667177. DOI: 10.1177/1063293X16671771.

[23] *SIDOPS+*. 20sim. URL: https://www.20sim.com/overview/introduction/ (visited on 03/27/2023).

[24] Liu Siyuan, Wanyou Li, Zhijun Shuai, and Meilong Chen. «Vibration Analysis of a Single-Cylinder Reciprocating Compressor considering the Coupling Effects of Torsional Vibration». In: *Shock and Vibration* 2019 (Apr. 2019), pp. 1–9. DOI: 10.1155/2019/3904595.

[25] *Sperre Air Power – Norwegian Maritime Exporters*. URL: https://www.nme.no/member/sperre/ (visited on 01/10/2023).

[26] *Sperre Classic*. Sperre. URL: https://sperre.com/products/sperre-classic (visited on 01/10/2023).

[27] *Sperre Skruekompressor*. Sperre. URL: https://sperre.com/products/screw (visited on 03/21/2023).

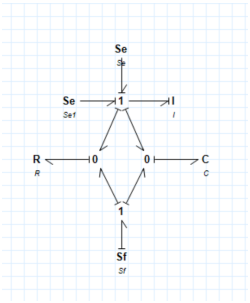
[28] *Sperre X-Range*. Sperre. URL: https://sperre.com/products/sperre-x-range (visited on 01/12/2023).

Vedlegg A. Utvikling

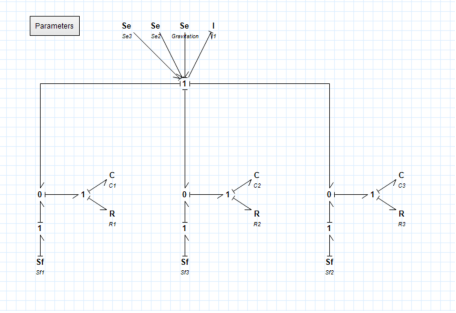
Vedlegg A.1. Utvikling matermatisk modell

UTVIKLING AV MATEMATISK MODELL

Dette vedlegget viser og forklarer utviklingen av den matematiske modellen og hvilke endringer som er gjort som et resultat av tidligere utvikling.

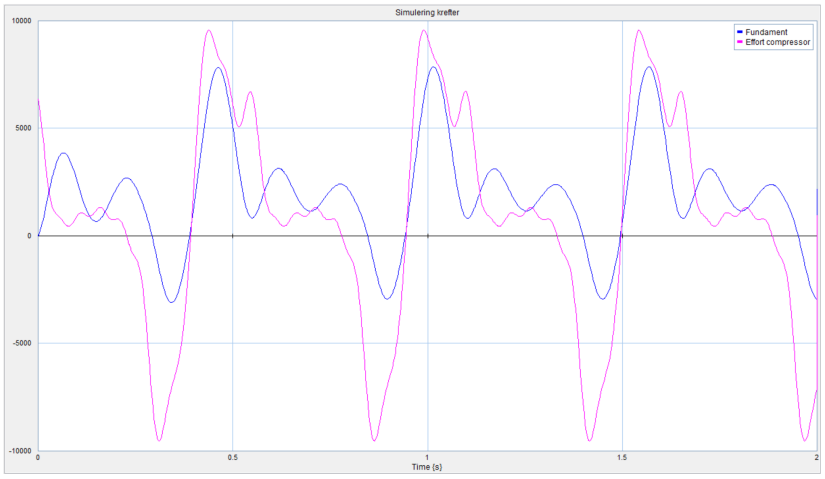


Figur 1: Mass-Spring-Damper i bond-graph.



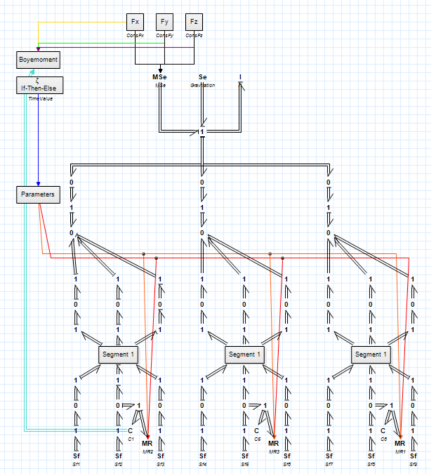
Figur 2: Tidlig modell av en kompressor med 3 dempere.

Modelleringen startet med et Mass-Spring-Damper system da dette appellerte godt mot problemstillingen. Grunnet at kompressoren har tre gummidempere ble modellen utvidet til dette.

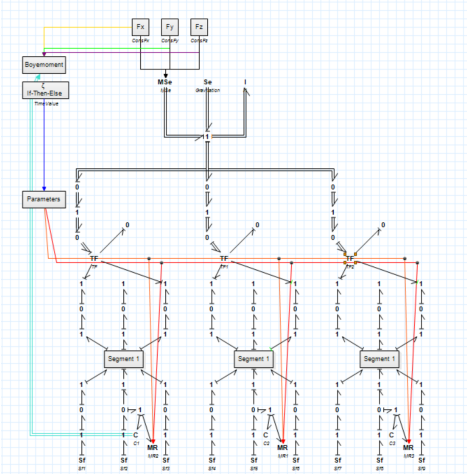


Figur 3: Tap av energi mellom kraft og fundament

Ved bruk av denne modellen i Figur 2 og eksempelkrefter mottatt fra Sperre ble det mulig å simulere krefter som virker opp og ned i systemet. Figur 3 viser en rosa graf som er krefter som virker ned i systemet og der blå representerer fundamentet. Arealet mellom er tapt energi.

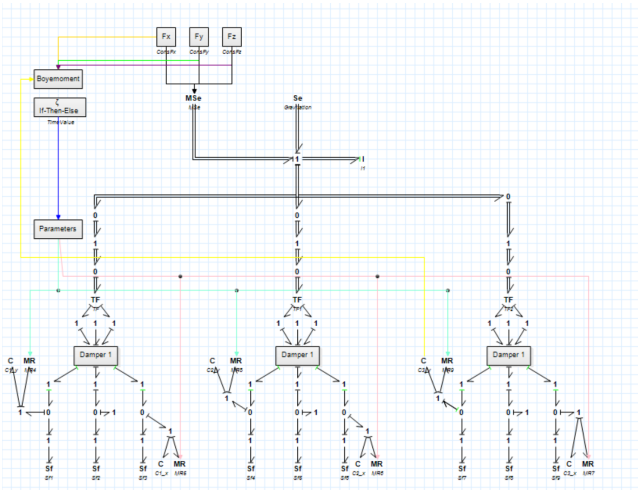


Figur 4: Ny modell. Matrise til enkelt bånd.



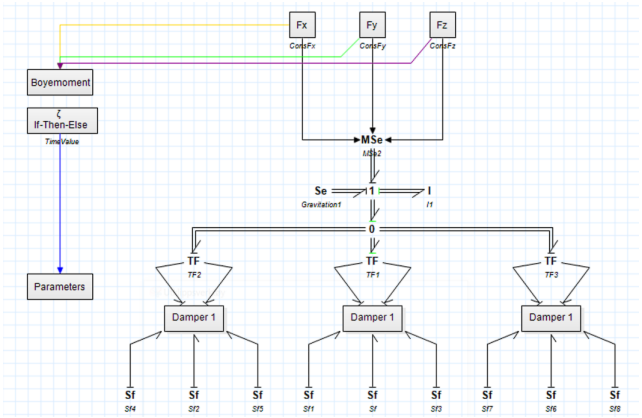
Figur 5: Ny modell. Matrise til matrise bånd.

For videreutvikling av modellen var nødvendig at flere komponenter ble introdusert. For å koble sammen krefter som virker inn i systemet, demping og fast innespenning der ulinearitet er introdusert ble Figur 4 Modellert. Figur 5 viser det sammensatte systemet etter en opprydding. Doble linjer gir uttrykk for at en tredimensjonal kraft trer gjennom. I dette tilfellet en [3,1] matrise av [x,y,z], MR er fortsatt en R men er nå blitt modulær (M for modulated).



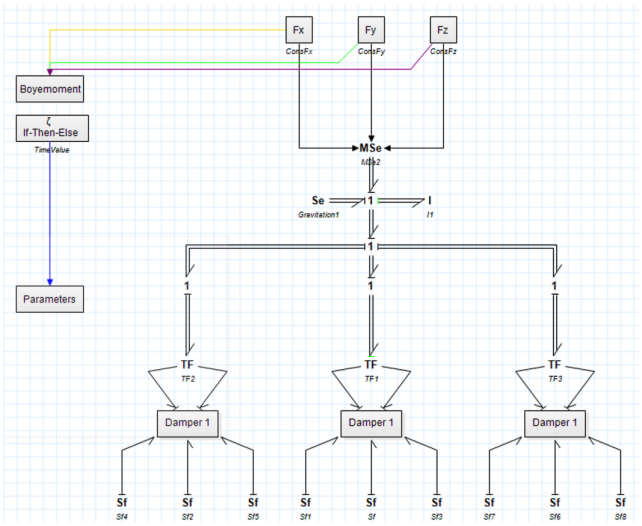
Figur 6: Modell hvor demping er utenfor MTF omforming

Figur 6 er en videreutvikling av Figur 4 og 5, der mye av koblingene er innbundet i dempersegmentet. Det er forstatt synlig i modellen er at demping ligger i x og z planet og ikke i selve θ, noe som ikke representerte modellen korrekt.



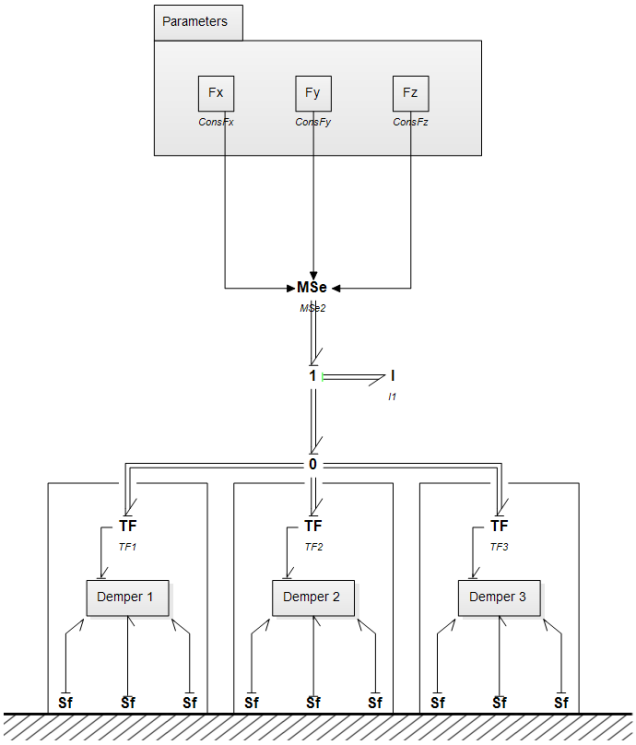
Figur 7: Modell med sammenkobling av dempere i 0-Junction

Grunnet at 0-Junction distribuerer lik kraft fra det som kommer inn til de andre bond ble dette naturligvis en feil fremgangsmåte da alle ben fikk like mye krefter inn som virket inn i 0-Junction. Demping der derimot innbundet i demperen på dette stadiet.



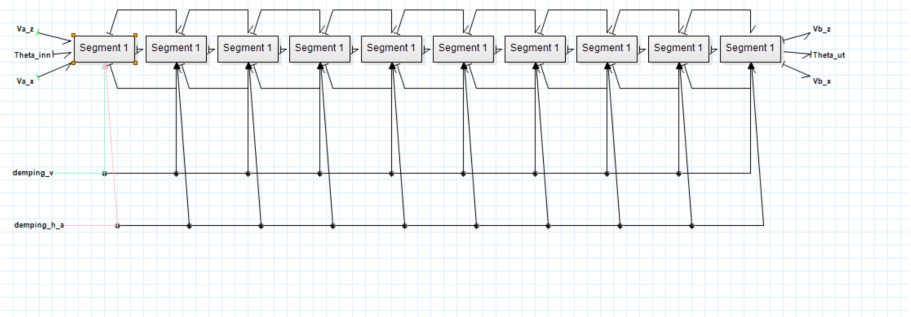
Figur 8: Modell med sammenkobling av dempere i 1-Junction

For å sikre at alle dempere ikke mottar den kraften som virker inn men likt respektivt til kraften kommer inn er 1-Junction benyttet isteden.



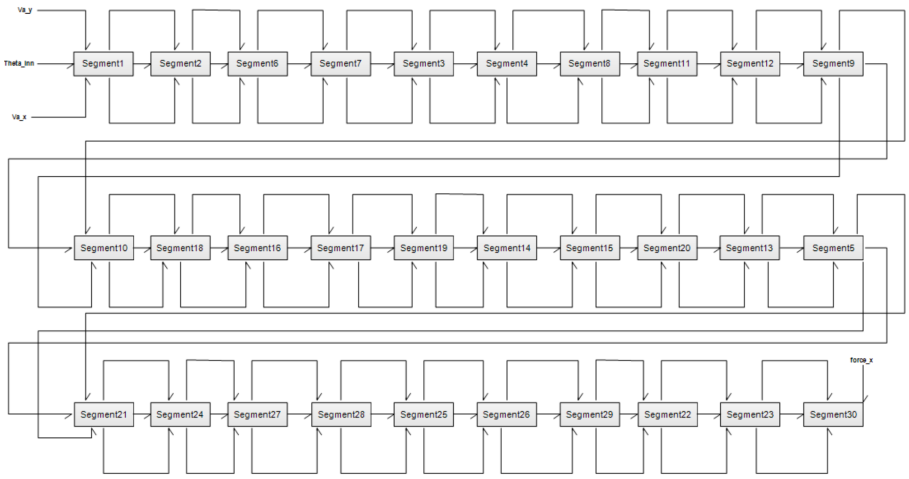
Figur 9: Ferdigutviklet matematisk modell

Ved den ferdigutviklede modellen er gravitasjonskreftene fjernet da modellen kun tar for seg kraft i x-retning. Modellen er også reversert tilbake til 0-Junction for å støtte kausaliteten av den ferdige modellen.



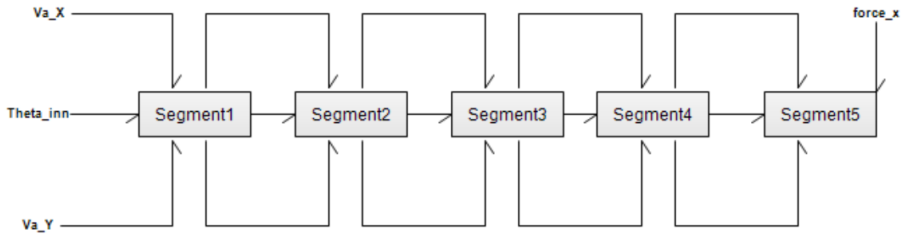
Figur 10: Gummidemper oppdelt i 10 segment

Ved å dele opp demperen i fler segment ble det mulig å hente ut verdier i spesifikke deler av demperen under last og ikke bare på enden. Figur 12 viser til en oppdeling av 10 segment.



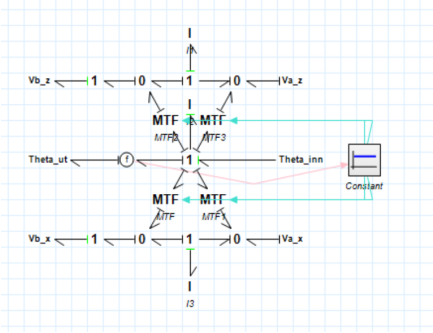
Figur 11: Gummidemper oppdelt i 30 segment

Det ble modellert en modell med 30 segment for å se utslaget dette ga for simuleringen. Grunnet at dette ga relativt lite utslag ble den endelige modellen modellert med 5 segment for å øke simuleringshastigheten.

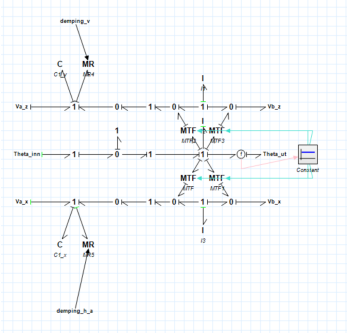


Figur 12: Gummidemper oppdelt i 5 segment

Den ferdige modellen inneholder 5 segment.

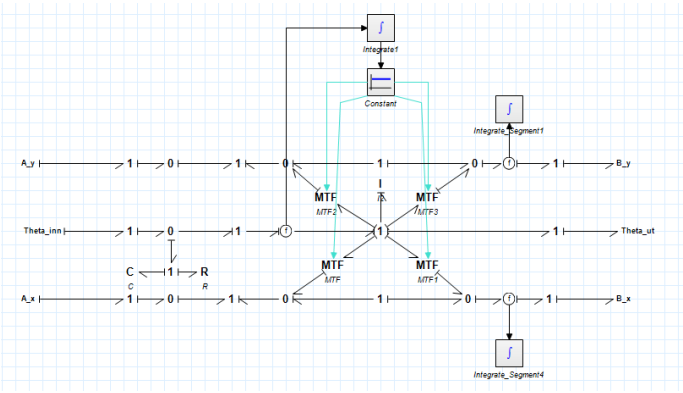


Figur 13: Segment uten demping.



Figur 14: Segment med demping.

Figur 13 og 14 Viser til segment både med og uten demping. Dempingen i Figur 14 er feilplassert da denne ikke skal dempe x og y bane.



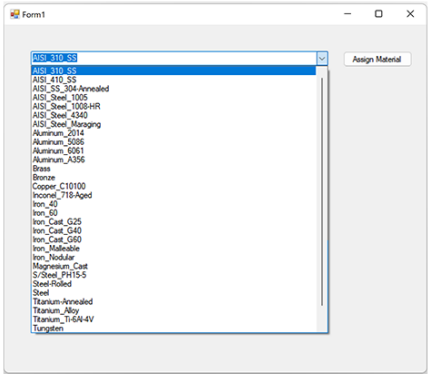
Figur 15: Ferdigutviklet segment.

Figur 15 er det siste segmentet og er ferdigutviklet. 2/3 inertia element er fjernet som reduserer simuleringstiden på grunn av algebraiske løkker. Demping er også flyttet til riktig plass da denne demper bøying en i gummidemperene. Det er også lagt på flowsensorer som integrerer signalet for å finne utbøyingen.

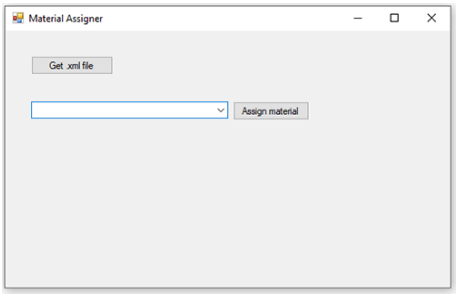
Vedlegg A.2. Utvikling GUI

UTVIKLING AV GUI

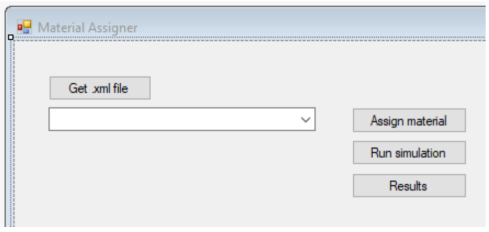
Dette vedlegget viser og forklarer utviklingen av brukergrensesnittet og hvilke endringer som er gjort visuelt og i kode.



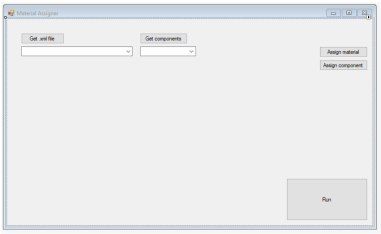
Figur 1: Rullegardinsliste av material



Figur 2: Tidlig utkast av GUI



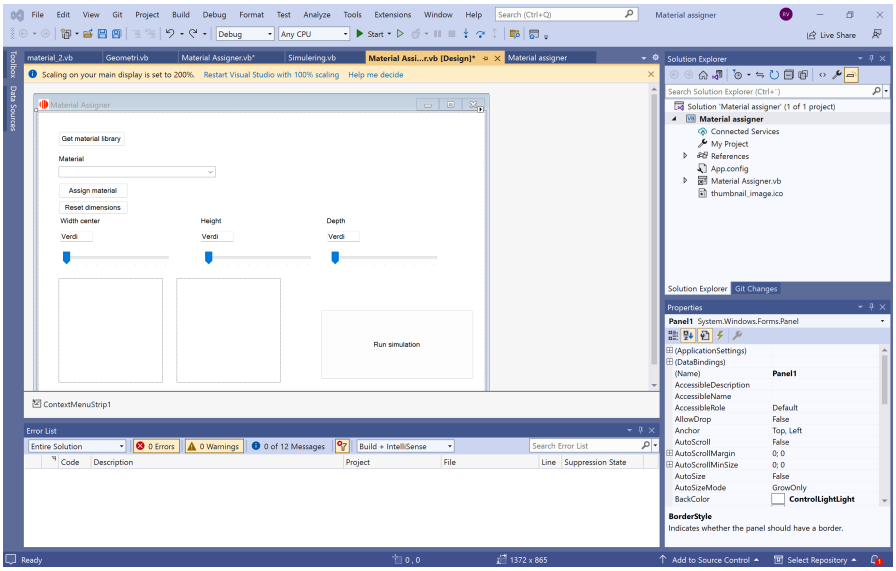
Figur 3: Utkast utviklet for å kunne kjøre simuleringen



Figur 4: Utkast med komponentvalg

GUI for oppgaven ble utviklet gjennom Visual Studio med den innebygde GUI-designer som vist i Figur 5. Det er her mulig å plassere ulike knapper, rullegardinslister og glidere. Utviklingen begynte ved å sette opp knapper og tildele funksjoner til de ulike prosessene systemet tar for seg. Figur 1 viser til det første utkastet av GUIen, denne har bare valg av material og lagring av materialvalget. Dette var for å teste ut programmet og hvordan det fungerte. Det ble oppdaget at materiallisten fra NX er lokalisert på forskjellige lokasjoner for forskjellige versjoner av NX, derfor er det laget en knapp som åpner et fil-søkevindu hvor brukeren selv importerer materiallisten. Materiallisten inneholder mange forskjellige material som metall, glass, gass og gummi og det er derfor laget en kode som vist i Figur 6 som kun henter ut de metalliske materialene og setter dem i rullegardinslisten. Figur 3 viser til en av de tidlige utviklingene av GUI, hvor det er laget en knapp for å kjøre simuleringen og for å vise resultater.

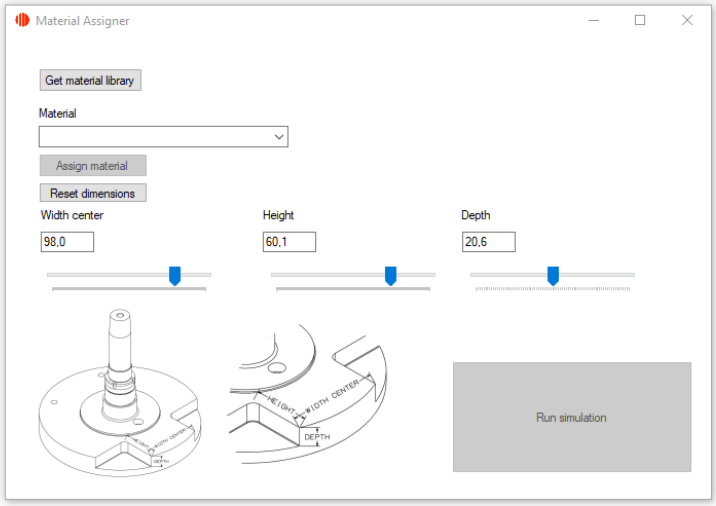
I Figur 4 er det implementert en boks hvor det velges hvilken komponent som skal tildeles en materialendring. Her er det også forandret på plasseringene av knapper, for å sikre et mer stilrent program. Denne versjonen har kun en knapp som kjører alle skript.



Figur 5: Bilde av GUI designer

```
44
45          For Each materialNode As XmlNode In materialNodeList
46              Dim classNode As XmlNode = materialNode.SelectSingleNode("BulkDetails/Class/Name")
47              If classNode.InnerText = "METAL" Then
48                  Dim materialNameNode As XmlNode = materialNode.SelectSingleNode("BulkDetails/Name")
49                  ComboBox1.Items.Add(materialNameNode.InnerText)
50              End If
51          Next
```

Figur 6: Kode som skiller ut metalliske material



Figur 7: Ferdigutviklet GUI

Etter at materialendring var implementert ble neste steg i prosessen å implementere geometriendring. Dette blir gjort ved å bruke funksjoner som gir brukeren mulighet til å endre verdier i sketch på kontravekten. Dette er blitt gjort ved å sette inn tre glidebarer og tre tekstbokser som vist i Figur 7. Det er her satt opp at bruker kan flytte glidebaren eller skrive inn ønsket verdi for bredde, høyde og dybde i tekstboksene. I denne GUI ble det også fjernet mye unødvendig kode for å gjøre at programmet kjører raskere. I Figur 7 er det også lagt inn to bilder som visualiserer hvilken geometriendring som kan gjøres. For at bruker ikke skal kunne bruke programmet feil, så er det lagt inn i koden at alle knapper er avslått før de kan brukes. Ved gjennomgang av GUI og oppgavens system, ble det gjort et valg om å fjerne komponentvalg da det kun er realistisk å gjøre en materialendring på stempelene.

I glidebaren ble det oppdaget at ved bruk av resetknapp ble verdiene ikke tilbakestilt til verdier med desimaler, derfor måtte koden for glidebaren endres fra som vist i Figur 8 til Figur 9. Metoden for å tvinge tekstboksen til å inkludere desimaler er det gjort med å først gange med 100 for å så dele på 100 igjen. I tillegg måtte det lages en kode (Figur 10) som endrer "," til ".", da tall med komma ikke ble forstått av NX.

```
138     Private Sub TrackBar3_slide(sender As Object, e As EventArgs) Handles MyBase.Load
139         ' Setter minimums- og maksimumsverdiene for TrackBar-kontrollen.
140         TrackBar3.Minimum = 1
141         TrackBar3.Maximum = 30
142
143         ' Setter standardverdi i trackbar.
144         TrackBar3.Value = 20.6
145
146         ' Setter standardverdi i textbox.
147         TextBox3.Text = TrackBar3.Value.ToString("N1")
148     End Sub
```

Figur 8: Gammel kode for glidebar

```
144     Private Sub TrackBar3_slide(sender As Object, e As EventArgs)
          Handles MyBase.Load
145         ' Setter maksimal og minimal verdi i trackbar og hvor store
              steg glidebaren har.
146         TrackBar3.Minimum = 100
147         TrackBar3.Maximum = 4000
148         TrackBar3.TickFrequency = 50
149         TrackBar3.LargeChange = 250
150         TrackBar3.SmallChange = 50
151
152         ' Setter standardverdi i trackbar.
153         TrackBar3.Value = 2060
154
155         ' Setter standerdverdi i textbox.
156         TextBox3.Text = (TrackBar3.Value / 100).ToString("N1")
157     End Sub
```

Figur 9: Ny kode for glidebar

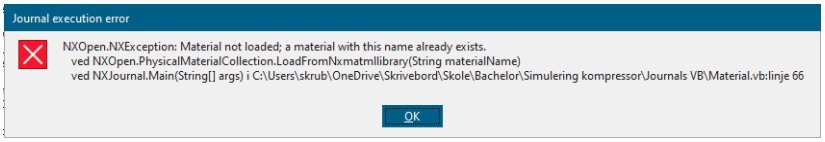
```
104             Dim lines As New List(Of String)(File.ReadAllLines
                  (filePath))
105             lines(0) = value.ToString("N1").Replace(",", ".")
106             File.WriteAllLines(filePath, lines)
```

Figur 10: Kode som endrer "," til "."

Vedlegg A.3. Feilmelding NX

FEILMELDINGER I NX

Det blir i dette vedlegget gjennomgått feilmeldinger møtt i Siemens NX.



Figur 1: Feilmelding journal materialskifte

Utfordringen forekom under utvikling av materialendrings journalen. Som vist i Figur 2 klarer ikke programmet laste inn et material som tidligere har vært brukt i filen. Dette ligger i en "loaded state", da det har blitt lastet inn men ikke er i bruk. For å løse dette må det inkluderes kode i journalen som sletter alle ubrukte materialer før det nye materialet påføres. Dette er et godt eksempel på hvordan journals bare lager en "dummy" kode, som må redigeres før den kan fungere som et komplett skript.

```
56          '-----------------------------------------------------
57          'Her slettes ubrukte lastede materialer, for å forhindre bugs med dobbellasting.
58          '-----------------------------------------------------
59
60          Dim physicalMaterial As NXOpen.PhysicalMaterial = Nothing
61
62          For Each tempMaterial As PhysicalMaterial In workPart.MaterialManager.PhysicalMaterials '.GetUsedMaterials
63          physicalMaterial = tempMaterial
64          physicalMaterial.Delete()
65      Next
```

Figur 2: Kode for fjerning av ubrukte, lastede materialer

Vedlegg B. Brukermanual

Instruksjonsmanual

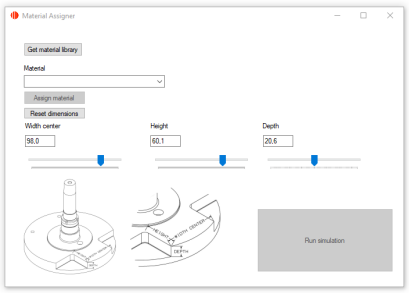
Innledende

Brukermanualen er tiltenkt en installasjonsinstruksjon for bruker slik at programmet skal kunne settes opp av utenforstående. For enkelhetes skyld anbefales det at programmet blir liggende direkte i:

('C:\menu')

Dersom bruker ønsker mappen lagret et annet sted er det nødvendig å endre alle paths i kodene til respektiv lokasjon på egen datamaskin. Manualen tar for seg operasjoner rundt programmet som forklarer hvilke funksjoner det innehar og hvordan det fungerer.

1. GUI



Figur 1: GUI

1.1. Installasjon

Før programmet anvendes må filene: Simulation.vb, material.vb og geometri.vb, som er først lokalisert under

('C:\Menu\Visual Basic\')

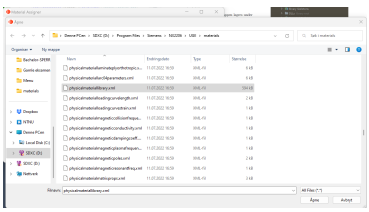
lagres i

('C:\...\Siemens\NX"VERSION"\NXBIN\')

For å kunne få frem materialer må .xml fil i "Get material library" hentes, denne filen ligger normalt på:

('C:\...\Siemens\NX"VERSION"\UGII\materials\physicalmateriallibrary.xml')

(også vist i Figur 2.)



Figur 2: Physicalmateriallibrary.xml lokasjon

1.2. Bruk

Start ved å åpne "Material Assigner.exe" for å åpne GUI. Deretter importere siemens lokale materialliste ved å trykke på knappen "Get material library", når materiallisten er valgt kan ønsket material velges i rullegardinlisten. Når ønsket material er valgt skal "Assign material" trykkes for å lagre verdien. Dersom en ønsker en geometriendring kan dette gjøres ved å enten bruke glideren eller ved å skrive inn dimensjonene som ønskes inn i tekstboksen. Dersom geometrien skal tilbakestilles til de orginale verdiene kan knappen "Reset dimensions" brukes. Når ønsket material og geometri er valgt kan simuleringen kjøres, simuleringen kjøres ved å trykke på "Run simulation". Simuleringen iverksettes og et vindu med "Simulation is running" vil vises, og et vindu med "Simulation finished" vil vises når simuleringen er ferdig. Resultatene fra simuleringen vil bli lagret under

```
('C:\Menu\Results\...csv').
```

.

2. Siemens NX og grensesnitt

For bruk av programmet anbefales NX2206 eller nyere. For funksjon av programmet i relasjon til NX, behøves alle modeller inkludert i mappen Kompressor, under C:\Menu. Skal modellene flyttes kan dette defineres i skriptene Simulation, Material, og Geometri. Det er i koden markert hvor nye lokasjoner kan plasseres. Det er likevel viktig å flytte alle filer, da assemblyfiler krever underliggende modellfiler i samme mappe.

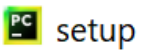
NX kan selv lese .vb kode, så ingen kodeplatform kreves av bruker. Skal .vb filer redigeres kan dette gjøres gjennom funksjonen Edit Journal.

3. Python og Controllab

For å kjøre pythonskriptet som er interpreterbart er det nødvendig at Pandas, NumPy, MatplotLib og Controllab er installert, Pandas, NumPy og Matplotlib kan alle installeres direkte i Pycharm men contrallab må installeres fra skriptingfiler som følger med 20-sim.

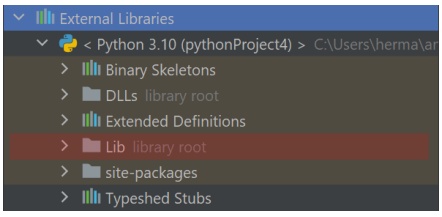
3.1. Automatisk installering

For automatisk installering kjøres setup.py som ligger lokalisert der 20-sim er installert.



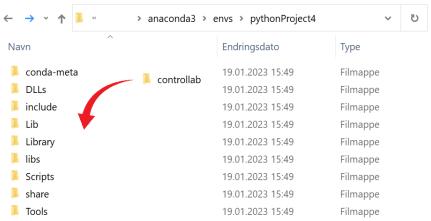
3.2. Manuell installering

Før bruk av skriptet som inneholder controllab-pluginen må "controllab" installeres lokalt i PyCharm For å gjøre dette er det av nødvendighet å finne frem til Lib mappen som ligger der prosjektfilen i PyCharm er installert. "controllab" mappen ligger vedlagt i "C:\Menu\Python_skripting_20sim" i tilsendt mappe.



Lib inneholder fra før plugins som følger med PyCharm. I tilfellet av utvikling av systemet er mappen for lib lokalisert i prosjektfilens bibliotek (Lib).

```
('C:\Users\name\Anaconda3\envs\pythonProjectX\Lib')
```



4. 20-sim

20-sim blir stående som urørt under prosessen, for å endre denne må dette gjøres manuelt. 20-sim støtter ikke batchmode og må derfor åpnes under prosessen. 20-sim tar automatisk ved hjelp av Pythonscript inn krefter fra NX i form av CSV og konvererer disse til variabler. Skriptet loggfører variablene og lager et plot i matplotlib. Denne DataFramen lagres i:

```
('C:\Menu\Results' som 'Materialnavn' + 'DisplacementDamper')
```

Vedlegg C. Vibration Class - DNV Rules side 7

Table 11 Turbine driven generators

Velocity
4 – 1000 Hz
7 mm/s
To be measured in any direction on the bearings. Applies to both fixed and resilient mounted.

Table 12 Gears

Velocity
4 – 1000 Hz
7 mm/s
To be measured in any direction on the foundation and on the input shaft bearing.

Table 13 Electric motors, separators, motor driven hydraulic pumps, fans not installed on reciprocating engines

	Velocity
	4.0 – 200 Hz [1)]
Internal excited	7 mm/s [2)]
External excited	12 mm/s [2)]
To be measured in any direction on the bearings. 1) The upper frequency limit shall be at least 200 Hz and above 2 x rpm. 2) For vertically mounted motors the vibration level may be increased by 50% for the top of the motor.	

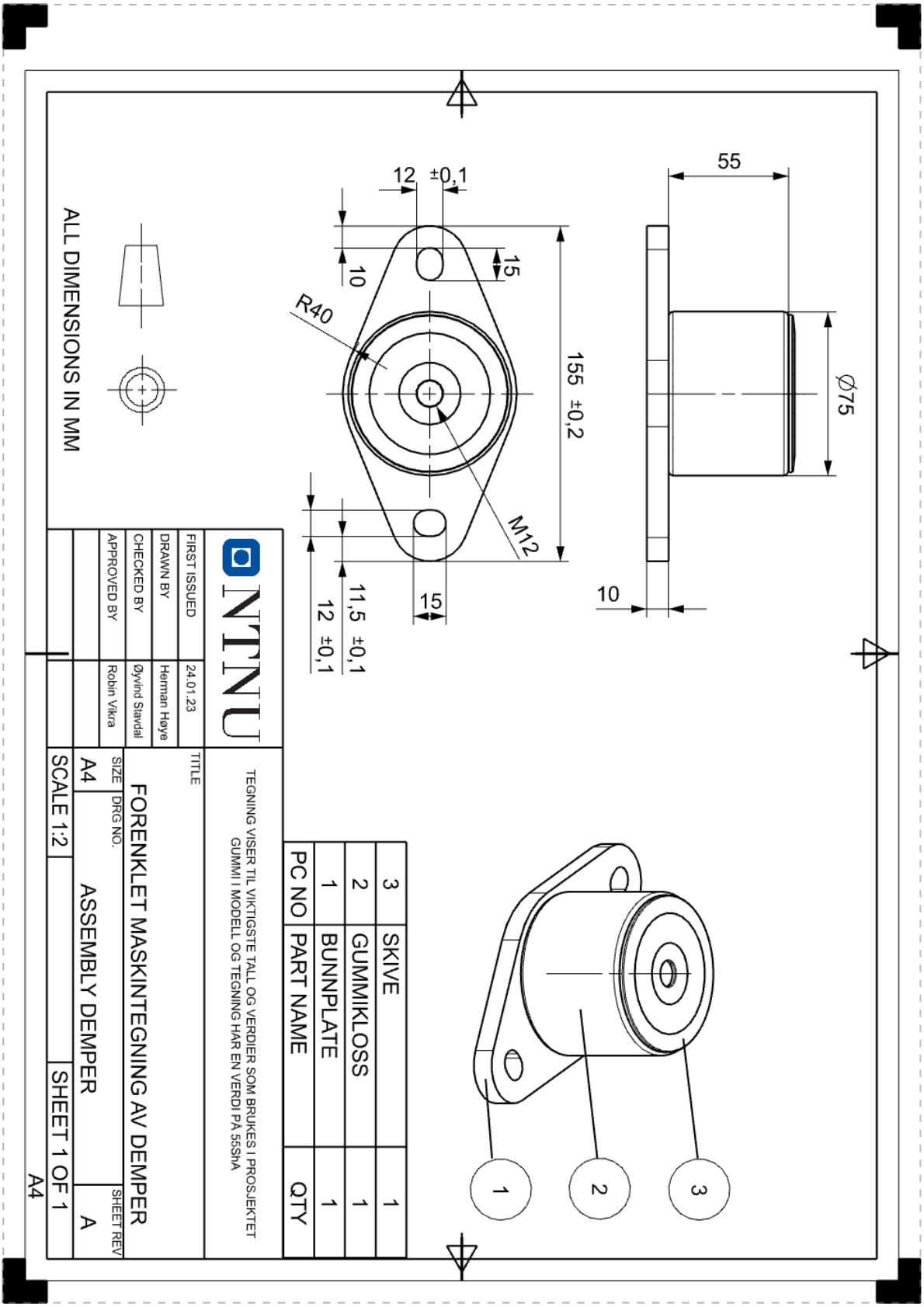
Table 14 Compressors (screw or centrifugal)

	Velocity
	4 – 200 Hz [1)]
Elastically mounted	10 mm/s
Fixed mounted	7 mm/s
To be measured in any direction on the bearings. 1) The upper frequency limit shall be at least 200 Hz and above 2x rpm.	

Table 15 Reciprocating compressors and reciprocating pumps

Velocity
4 – 200 Hz
30 mm/s
To be measured in any direction on the bearings. Applies for both resilient and fixed mounted.

Vedlegg D. Forenklet maskintegning av demper



Vedlegg E. Technical information for rubber buffers

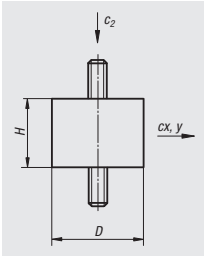
Technical information for rubber buffers

Note:
Our rubber buffers are simple and cost-efficient standard units for elastic mounting. They are ideally suited for compressive and axial loads of a diverse range of applications. With shear stress however, they are substantially less resilient than with compressive stress. The adjacent tables provide a guide to the static load values. By high dynamic alternating loads or high frequencies the loading should be reduced proportionately.

Guide values for static load (excerpt from 26100, 26102, 26104 and 26106)

Type	D	H	Compressive loads						Shear stresses					
			Spring rate c2 in N/mm			Permissible load F in N			Spring rate cx, y in N/mm			Permissible load F in N		
			hard	medium	soft	hard	medium	soft	hard	medium	soft	hard	medium	soft
A	20	15	300	190	120	500	320	200	60	40	30	190	120	70
A	30	15	670	410	250	1100	700	400	90	60	40	350	210	130
A	30	30	240	150	100	900	570	340	50	30	20	430	280	170
A	40	30	480	300	170	1800	1110	670	90	60	30	770	500	250
A	50	20	2400	1500	900	5000	3190	1870	240	160	100	1200	770	460
A	50	40	600	380	220	2800	1750	1050	120	80	50	1280	800	460
A	75	25	5000	1655	1700	8000	5000	3300	410	260	160	2800	1750	1030
A	75	55	650	400	240	4700	3000	1750	130	80	50	2100	1300	800
B	25	20	320	160	120	490	320	190	70	45	25	230	160	90
B	30	20	660	430	260	830	520	310	100	75	50	330	210	130
B	30	30	350	220	130	750	450	280	70	50	30	350	220	130
B	40	30	550	350	210	1250	750	450	110	70	40	520	330	200
B	50	40	560	370	220	2100	1270	760	120	80	45	930	580	350
B	50	50	350	220	130	1750	1100	650	80	50	30	800	510	310
B	75	50	950	630	330	4700	2910	1720	180	120	80	1900	1200	710
C	20	25	200	130	80	300	190	120	50	30	20	150	90	60
C	30	30	590	380	220	720	450	270	90	60	50	260	170	110
C	40	30	900	570	340	1080	680	410	150	90	60	380	240	140
C	50	30	1700	1090	650	2500	1750	950	210	150	70	470	290	170
C	50	50	360	220	140	1390	870	520	80	40	30	610	390	230
C	75	50	1010	630	370	3650	2050	1200	200	130	80	1560	980	580

Type	D	H	Compressive loads	
			Spring rate c2 in N/mm	Permissible load F in N
			medium	medium
D	25	20	150	260
D	30	20	330	730
D	40	30	250	950
D	50	20	660	1750
D	75	25	1430	4650



Rubber hardness:
hard = 70 Shore medium = 55 Shore soft = 45 Shore
For general guidance natural rubber is ca. 55 Shore.
static compression load: F (max.) = ca. 6.5 kg/cm² (63.77 N/cm²)
static axial load: F (max.) = ca. 1.5 kg/cm² (14.72 N/cm²)
by 10 % spring displacement, or transverse travel during axial load.
Naturally, much higher loads are possible without damage, but these considerably effect the rubber buffer in its primary purpose. Tensile loads are possible but should be avoided on account of the peak stress at the contact edges and the notch sensitivity of rubber.

Tolerances for rubber buffers
Permissible dimensional deviations per DIN 7751 Part 2. Permissible hardness deviation ±5 Shore A.

Synoptical Table - Properties of the Individual Material

Rubber material		Main Characteristics - Resistance to									
Abb.	Polymer	Temperature	Tensile strength	Fracture strain	Aging	Ozone	Petrol	Oil	Acid	Alkalis	Tensile strain
NR (NK)	Natural rubber	-30 °C – +80 °C	1	1	3	4	6	6	3	3	600%
SBR	Styrene-butadiene rubber	-30 °C – +80 °C	5	2	3	4	4	5	3	3	450%
CR	Chloroprene rubber	-20 °C – +110 °C	3	2	2	2	2	2	2	2	450%
NBR	Acrylonitrile-butadiene rubber	-30 °C – +120 °C	5	2	3	3	1	1	4	3	450%
EPDM	Ethylene propylene terpolymer	-30 °C – +130 °C	5	3	1	1	5	4	1	2	450%
SI	Silicone rubber	-60 °C – +200 °C	6	4	1	1	5	4	5	5	500%

1 = excellent 2 = very good 3 = good 4 = moderate 5 = low 6 = insufficient

Vedlegg F. Målinger fra CMM

Vedlegg F.1. Rapport innenfor mål

ZEISS **ZEISS CALYPSO**

SPERRE
Dependable Air Power

Zeiss prog name			
Part name / Item no			
Drawing number			
Drawing index		Run	All Characteristics
Order number		Last 1 measurements	
Part ident		► Approval ≠ Blocked	
Casting batch no.		No. measured values	23
Time/Date		No. values: red	0
Text			

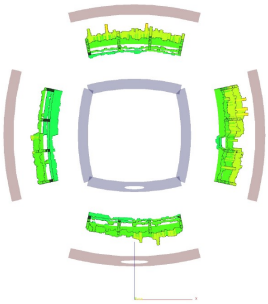
Name	Measured value	Nominal value	Upper allowance	Lower allowance	Deviation	+/-
Diameter						
Diameter_Cylinder Veivtapp						
Diameter_Cylinder rammelager						
Diameter_Circle innvendig						
Diameter_Circle tetningsring						
Roundness Veivtapp						
Slaglengde veivtapp						
Perpendicularity veivtapp vs ref.plan						
For CNC						
True Position antatt lagerleie kontravekt						
True Position antatt lagerleie kontrave…						
True Position antatt lagerleie kontrave…						
True Position skive						
True Position skive .Z						
True Position skive .X						
Profile kon vs BA						
Profile kon isolert						
Concentricity mellom spindler						
Roundness skive						
Distance vekt side1_Y						
Distance vekt side2_Y						
Distance dybde skive_Y						

Grå bokser sensurerer verdier og toleranser

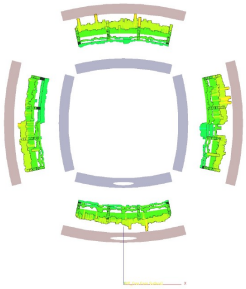
ZEISS CALYPSO

Zeiss prog name
Part name / Item no
Order number
Part ident
Time/Date

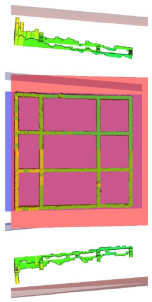
Name	Measured value	Nominal value	Upper allowance	Lower allowance	Deviation	+/-
CAD-Kon vs BA						



CAD-Kon isolert view1



CAD-Kon isolert view2



Vedlegg F.2. Rapport med forskyvning i horisontalt plan

ZEISS **ZEISS CALYPSO**

SPERRE Dependable Air Power

Zeiss prog name			
Part name / Item no			
Drawing number			
Drawing index	Run	All Characteristics	
Order number	Last 1 measurements		
Part ident	► Approval ≠ Blocked		
Casting batch no.	No. measured values	23	
Time/Date	No. values: red	7	
Text			

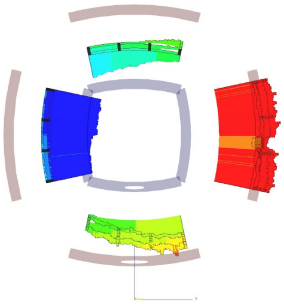
Name	Measured value	Nominal value	Upper allowance	Lower allowance	Deviation	+/-	
Diameter							
Diameter_Cylinder Veivtapp							
Diameter_Cylinder rammelager							
Diameter_Circle innvendig							
Diameter_Circle tetningsring							
Roundness Veivtapp							
Slaglengde veivtapp							
Perpendicularity veivtapp vs ref.plan							
For CNC							
True Position antatt lagerleie kontravekt							
True Position antatt lagerleie kontrave…							
True Position antatt lagerleie kontrave…							
True Position skive							
True Position skive .Z							
True Position skive .X							
Profile kon vs BA							
Profile kon isolert							
Concentricity mellom spindler							
Roundness skive							
Distance vekt side1_Y							
Distance vekt side2_Y							
Distance dybde skive_Y							

Grå bokser sensurerer verdier og toleranser

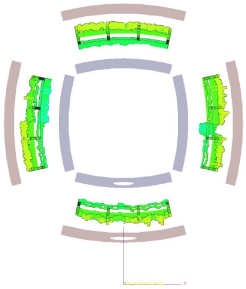
ZEISS CALYPSO

Zeiss prog name
Part name / Item no
Order number
Part ident
Time/Date

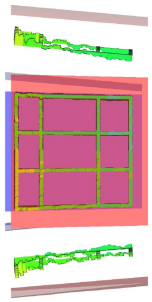
Name	Measured value	Nominal value	Upper allowance	Lower allowance	Deviation	+/-
CAD-Kon vs BA						



CAD-Kon isolert view1



CAD-Kon isolert view2



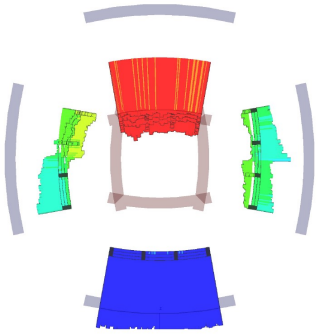
Vedlegg F.3. Rapport med forskyvning i vertikalt plan

ZEISS **ZEISS CALYPSO**

SPERRE
Dependable Air Power

Zeiss prog name			
Part name / Item no	CMM Type		
Drawing number			
Drawing index	Run	All Characteristics	
Order number	Last 1 measurements		
Part ident	► Approval ≠ Blocked		
Casting batch no.	No. measured values	13	
Time/Date	No. values: red	7	
Text			

Name	Measured value	Nominal value	Upper allowance	Lower allowance	Deviation	+/-	
Diameter Cylinder							
Diameter Tetning							
Slaglengde							
Perpendicularity aksel vs referanseplan							
For CNC							
Concentricity mellom spindlene							
Radial Runout							
Profil kon vs BA							
Profil kon isolert							
X-Avvik Circle							
True Position antatt lagerleie veivtapp							
CAD-Kon vs BA							

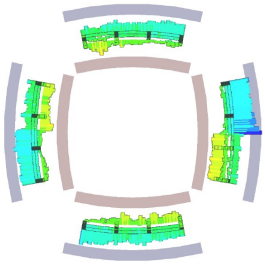


Grå bokser sensurerer verdier og toleranser

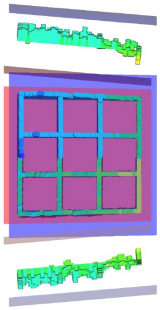
ZEISS CALYPSO

Zeiss prog name
Part name / Item no
Order number
Part ident
Time/Date

Name	Measured value	Nominal value	Upper allowance	Lower allowance	Deviation	+/-
CAD-Kon isolert						



CAD-Kon isolert 2						



Vedlegg G. Predefinerte koder 20-Sim

I Element (I - (Inertia)

```
parameters
        real i = 1;
equations
    state = int(p.e);
    p.f = state / i;
```

I-elementet er tregheten av massen for systemet som begever seg. I tilfelle av oppgaven er det massen av kompressoren uten ben.

C Element (Fjær)

```
parameters
        real c = 1;
equations
        state = int(p.f);
        p.e = state / c;
```

C-elementet representerer en fjær, der potensiell energi lagres i form av elastisk deformasjon når den belastes.

R Element (Demper)

```
parameters
        real r = 1.0;
equations
        p.e = r * p.f;
```

R-elementet representerer en demper, som bidrar til å redusere bevegelsesenergien i et systemet. Alle komponenter bevarer energien i systemet unntatt R-elementet.

0-Junction

```
equations
        sum (direct (p.f)) = 0;
        equal (collect (p.e));
        effort = first (p.e);
```

0-junction representerer en kobling der alle efforts fra bånd er like. Summen av all flow er lik null, og orienteringen av båndene bestemmer summen av flow. all flow fra bånd som peker mot 0 skal legges til og all flow som peker vekk skal trekkes fra. Kun ett bånd kan ha kausaliteten effort-inn.

1-Junction

```
equations
        sum(direct (p.e)) = 0;
        equal (collect (p.f));
        flow = first (p.f);
```

1-junction er det motsatte koblingspunktet av 0-junction, der lall flow er lik men alle efforts skal summeres til null. kausalitet spiller også en rolle her men er motsatt av i 0-junction, der alle bånd skal ha en effort-inn men kun ett bånd er tillatt som effort-ut.

Se Element (Effort)

```
parameters
        real effort = 1;
variables
        real flow;
equations
        p.e = effort;
        flow = p.f;
```

Se-elementet er kilden til kraft. Den settes som standard til å være en konstant men kan endres til å være dynamisk.

Sf Element (Flow)

```
parameters
        real flow = 1;
variables
        real effort;
equations
        p.f = flow;
        effort = p.e;
```

Sf-elementet er kilten til hastighet og settes som standard til en konstant. Sf kan settes til 0 og da vil systemet oppføre seg som fast innspent.

TF (Transformator)

```
parameters
        real r = 1;
equations
        p1.e = r * p2.e;
        p2.f = r * p1.f;
```

-elementet representerer en kraftkontinuelig relasjon mellom flow og effort for begge porter (innput og output) For kausualitetens del er alltid en port relatert til effort mens den andre har en flow kausualitet.

MSe (Modulated SE)

```
variables
        real flow[3, 1];
equations
        p.e = [forcex; forcey; forcez];
```

MSe-elementet fungerer på samme måte som et Se-element med forbehold om at koden er predefinert til å være modulær. Denne funksjonen forenkeler kun for bruker og oversikten i det globale systemet da et Se-element kan skrives om til et MSe-element.

Vedlegg H. Python-kode

File - C:\Menu\20simINIT.py

```python
import controllab
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

logvariables = ['time', 'Submodel2.Segment5.Integrate1.output']

# X-CSV
FX = pd.read_csv("C:\\Menu\\FX.csv", skiprows=2, header=None)
FX = FX.rename(columns={0: 'Time(sec)', 1: 'Force(N)_Real'})
# Definerer max tid for 360 grader
max_time = 0.0337
# Definerer antall vinkler 72
num_vinkler = 72
# Tidsintervall per vinkel
vinkel_size = max_time / num_vinkler
# Opprette ny liste for vinkler
force_list = [None] * num_vinkler
for i in range(len(FX)):
    # Bestemmer vinkelen i grader for korresponderende tid
    angle = (FX.loc[i, 'Time(sec)'] / max_time) * 360
    vinkel_index = int(angle / vinkel_size) % num_vinkler
    if force_list[vinkel_index] is None or FX.loc[i, 'Time(sec)'] < force_list[vinkel_index][0]:
        force_list[vinkel_index] = (FX.loc[i, 'Time(sec)'], FX.loc[i, 'Force(N)_Real'])
# Iterer for hver vinkel og print force evt ingen force'

angles = []
force_x = []
for i in range(num_vinkler):
    if force_list[i]:
        angle = (i / num_vinkler) * 360
        print(f"Angle {angle:.2f} degrees: force_x = {force_list[i][1]:.3f}")
    else:
        angle = (i / num_vinkler) * 360
        print(f"Angle {angle:.2f} degrees: no force_x value found")
    angles.append(round(angle))
    force_x.append(force_list[i][1])
print(angles, force_x)

force_x = np.array(force_x)
angles = np.array(angles)
force_values = [force[1] if force is not None else None for force in force_list]
# angles = [(i / num_vinkler) * 360 for i in range(num_vinkler)]

# Y-CSV
FY = pd.read_csv("C:\\Menu\\FY.csv", skiprows=2, header=None)
FY = FY.rename(columns={0: 'Time(sec)', 1: 'Force(N)_Real'})
# Definerer max tid for 360 grader
max_time_y = 0.0337
# Definerer antall vinkler 72
num_vinkler_y = 72
# Tidsintervall per vinkel
vinkel_size_y = max_time_y / num_vinkler_y
# Opprette ny liste for vinkler
force_list_y = [None] * num_vinkler_y
for i in range(len(FY)):
    # Bestemmer vinkelen i grader for korresponderende tid
    angle_y = (FY.loc[i, 'Time(sec)'] / max_time_y) * 360
    vinkel_index_y = int(angle_y / vinkel_size_y) % num_vinkler_y
    if force_list_y[vinkel_index_y] is None or FY.loc[i, 'Time(sec)'] < force_list_y[vinkel_index_y][0
]:
        force_list_y[vinkel_index_y] = (FY.loc[i, 'Time(sec)'], FY.loc[i, 'Force(N)_Real'])
# Iterer for hver vinkel og print force evt ingen force'

angles_y = []
force_y = []
for i in range(num_vinkler_y):
    if force_list_y[i]:
```

File - C:\Menu\20simINIT.py

```
        angle_y = (i / num_vinkler_y) * 360
        print(f"Angle {angle_y:.2f} degrees: force_y = {force_list_y[i][1]:.3f}")
    else:
        angle_y = (i / num_vinkler) * 360
        print(f"Angle {angle_y:.2f} degrees: no force_y value found")
    angles_y.append(round(angle_y))
    force_y.append(force_list_y[i][1])
print(angles_y, force_y)

force_y = np.array(force_y)
angles_y = np.array(angles_y)
force_values_y = [force[1] if force is not None else None for force in force_list_y]
# angles = [(i / num_vinkler) * 360 for i in range(num_vinkler)]

ports = ['5580']
xxsim = controllab.XXSim()
xxsim.connect()
# if not xxsim.connect():
#     print('Operation failed! Aborting.')
#     exit(1)
xxsim.open_model('C:\\Menu\\Bachelorstart.emx')
xxsim.set_log_variables(logvariables)
xxsim.set_parameters(['ConsFx.y', 'ConsFx.alpha'], [force_x, angles])
xxsim.set_parameters(['ConsFy.o', 'ConsFy.beta'], [force_y, angles_y])
xxsim.process_model()
xxsim.set_scriptmode(False)
xxsim.run()
logged_data = xxsim.get_log_values(logvariables)
temp = logged_data['time']['Submodel2.Segment5.Integrate1.output']
xxsim.close_model()
xxsim.disconnect()
plt.plot(tid,displacement)
plt.xlabel('Time')
plt.ylabel('Displacement (m)')
plt.title('Displacement in rubber damper')
plt.show()

```

Vedlegg I. Visual Basic kode

Vedlegg I.1. GUI-kode

C:\Menu\Material assigner.vb 1

```
1  ' NX 2206
2  ' Skript for Material assigner GUI.
3
4  Imports System.IO
5  Imports System.Windows.Forms.VisualStyles.VisualStyleElement
6  Imports System.Windows.Forms.VisualStyles.VisualStyleElement.ToolBar
7  Imports System.Xml
8  Imports NXOpen
9
10
11 Public Class Form1
12     Private selectedMaterial As String = ""
13     Private selectedValue As String = ""
14
15     Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
         MyBase.Load
16         'Gjør at alle Buttons ikke er aktiv før de skal brukes
17         Button2.Enabled = False
18         Button3.Enabled = False
19
20     End Sub
21
22     Private Sub picturebox(sender As Object, e As EventArgs) Handles
         MyBase.Load
23         ' Laster inn bilde for PictureBox1.
24         Dim image1 As Image = Image.FromFile("C:\menu
             \kontravektgui.png")
25         PictureBox1.Image = image1
26         PictureBox1.SizeMode = PictureBoxSizeMode.Zoom
27
28         ' Laster inn bilde for PictureBox2.
29         Dim image2 As Image = Image.FromFile("C:\menu
             \kontravektgui2.png")
30         PictureBox2.Image = image2
31         PictureBox2.SizeMode = PictureBoxSizeMode.Zoom
32     End Sub
33
34
35
36     Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
         Button1.Click
37         ' Laster inn materialer fra XML-fil.
38         Dim openFileDialog As New OpenFileDialog()
39         openFileDialog.Filter = "All Files (*.*)|*.*"
40         openFileDialog.InitialDirectory = "D:\Siemens\NX2206\UGII
             \materials"
41
42         If openFileDialog.ShowDialog() = DialogResult.OK Then
43             Dim xmlDoc As XmlDocument = New XmlDocument()
44             xmlDoc.Load(openFileDialog.FileName)
45
46             Dim materialNodeList As XmlNodeList =
                 xmlDoc.GetElementsByTagName("Material")
```

C:\Menu\Material assigner.vb 2

```
47
48              For Each materialNode As XmlNode In materialNodeList
49                  Dim classNode As XmlNode =
                      materialNode.SelectSingleNode("BulkDetails/Class/
                      Name")
50                  If classNode.InnerText = "METAL" Then
51                      Dim materialNameNode As XmlNode =
                      materialNode.SelectSingleNode("BulkDetails/Name")
52                      ComboBox1.Items.Add(materialNameNode.InnerText)
53                  End If
54              Next
55
56              Button2.Enabled = True
57          End If
58      End Sub
59
60
61      Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
           Button2.Click
62          ' Tilordner valgt materiale.
63          If ComboBox1.SelectedItem IsNot Nothing Then
64              selectedMaterial = ComboBox1.SelectedItem.ToString()
65              Dim fileName As String = "C:\Menu\selected_material.txt"
66
67              Dim streamWriter As System.IO.StreamWriter = New
                  System.IO.StreamWriter(fileName, False)
68              streamWriter.WriteLine(selectedMaterial)
69              streamWriter.Close()
70              MessageBox.Show("Material Assigned.")
71              Button3.Enabled = True ' Enable Button3
72          Else
73              MessageBox.Show("Please select a material from the
                  ComboBox.")
74          End If
75      End Sub
76
77
78
79      Private Sub TrackBar1_slide(sender As Object, e As EventArgs)
          Handles MyBase.Load
80          ' Setter maksimal og minimal verdi i trackbar og hvor store
              steg glidebaren har.
81          TrackBar1.Minimum = 1000
82          TrackBar1.Maximum = 12000
83          TrackBar1.TickFrequency = 50
84          TrackBar1.LargeChange = 250
85          TrackBar1.SmallChange = 50
86
87          ' Setter standardverdi i trackbar.
88          TrackBar1.Value = 9800
89
90          ' Setter standerdverdi i textbox.
91          TextBox1.Text = (TrackBar1.Value / 100).ToString("N1")
```

C:\Menu\Material assigner.vb 3

```
 92      End Sub
 93
 94      Private Sub TrackBar1_ValueChanged(sender As Object, e As
           EventArgs) Handles TrackBar1.ValueChanged
 95          ' Oppdater Tekst-egenskapen til TextBox-kontrollen med
               gjeldende verdi fra TrackBar-kontrollen.
 96          TextBox1.Text = (TrackBar1.Value / 100).ToString("N1")
 97      End Sub
 98
 99      Private Sub TextBox1_TextChanged(sender As Object, e As EventArgs)
           Handles TextBox1.TextChanged
100          ' Ser på teksten i TextBox-kontrollen og setter verdien til
               egenskapen Value på TrackBar-kontrollen.
101          Dim value As Double
102          If Double.TryParse(TextBox1.Text, value) AndAlso value >=
               TrackBar1.Minimum / 100 AndAlso value <= TrackBar1.Maximum /
               100 Then
103              TrackBar1.Value = CInt(value * 100)
104
105              ' Skriver den valgte verdien til en tekstfil.
106              Dim filePath As String = "C:\Menu\Geometry\Width.txt"
107              Dim lines As New List(Of String)(File.ReadAllLines
                   (filePath))
108              lines(0) = value.ToString("N1").Replace(",", ".")
109              File.WriteAllLines(filePath, lines)
110          End If
111      End Sub
112
113      Private Sub TrackBar2_slide(sender As Object, e As EventArgs)
           Handles MyBase.Load
114          ' Setter minimums- og maksimumsverdiene for TrackBar-kontrollen
                og hvor store steg glidebaren har.
115          TrackBar2.Minimum = 1500
116          TrackBar2.Maximum = 7550
117          TrackBar2.TickFrequency = 25
118          TrackBar2.LargeChange = 50
119          TrackBar2.SmallChange = 25
120
121          ' Setter standardverdi i trackbar
122          TrackBar2.Value = 6010
123
124          ' Setter standardverdi i textbox
125          TextBox2.Text = (TrackBar2.Value / 100).ToString("N1")
126      End Sub
127
128      Private Sub TrackBar2_ValueChanged(sender As Object, e As
           EventArgs) Handles TrackBar2.ValueChanged
129          ' Oppdaterer Text-egenskapen til TextBox-kontrollen med
               gjeldende verdi av TrackBar-kontrollen.
130          TextBox2.Text = (TrackBar2.Value / 100).ToString("N1")
131      End Sub
132
133      Private Sub TextBox2_TextChanged(sender As Object, e As EventArgs)
```

C:\Menu\Material assigner.vb 4

```
        Handles TextBox2.TextChanged
134       ' Ser på teksten i TextBox-kontrollen og setter verdien til
            egenskapen Value på TrackBar-kontrollen.
135       Dim value As Double
136       If Double.TryParse(TextBox2.Text, value) AndAlso value >=
            TrackBar2.Minimum / 100 AndAlso value <= TrackBar2.Maximum /
            100 Then
137         TrackBar2.Value = CInt(value * 100)
138
139         ' Lagrer verdien i .txt dokument.
140         Dim filePath As String = "C:\Menu\Geometry\Height.txt"
141         Dim lines As New List(Of String)(File.ReadAllLines
              (filePath))
142         lines(0) = value.ToString("N1").Replace(",", ".")
143         File.WriteAllLines(filePath, lines)
144       End If
145     End Sub
146
147     Private Sub TrackBar3_slide(sender As Object, e As EventArgs)
        Handles MyBase.Load
148       ' Setter maksimal og minimal verdi i trackbar og hvor store
            steg glidebaren har.
149       TrackBar3.Minimum = 100
150       TrackBar3.Maximum = 4000
151       TrackBar3.TickFrequency = 50
152       TrackBar3.LargeChange = 250
153       TrackBar3.SmallChange = 50
154
155       ' Setter standardverdi i trackbar.
156       TrackBar3.Value = 2060
157
158       ' Setter standerdverdi i textbox.
159       TextBox3.Text = (TrackBar3.Value / 100).ToString("N1")
160     End Sub
161
162     Private Sub TrackBar3_ValueChanged(sender As Object, e As
        EventArgs) Handles TrackBar3.ValueChanged
163       ' Oppdater Tekst-egenskapen til TextBox-kontrollen med
            gjeldende verdi fra TrackBar-kontrollen.
164       TextBox3.Text = (TrackBar3.Value / 100).ToString("N1")
165     End Sub
166
167     Private Sub TextBox3_TextChanged(sender As Object, e As EventArgs)
        Handles TextBox3.TextChanged
168       ' Ser på teksten i TextBox-kontrollen og setter verdien til
            egenskapen Value på TrackBar-kontrollen.
169       Dim value As Double
170       If Double.TryParse(TextBox3.Text, value) AndAlso value >=
            TrackBar3.Minimum / 100 AndAlso value <= TrackBar3.Maximum /
            100 Then
171         TrackBar3.Value = CInt(value * 100)
172
173         ' Skriver den valgte verdien til en tekstfil.
```

C:\Menu\Material assigner.vb 5

```
174             Dim filePath As String = "C:\Menu\Geometry\Depth.txt"
175             Dim lines As New List(Of String)(File.ReadAllLines
                  (filePath))
176             lines(0) = value.ToString("N1").Replace(",", ".")
177             File.WriteAllLines(filePath, lines)
178         End If
179     End Sub
180
181     Private Sub button4_Click(sender As Object, e As EventArgs) Handles
           Button4.Click
182         ' Nullstiller verdiene i glidebar og tekstboksen.
183         TrackBar1.Value = 9800
184         TextBox1.Text = (TrackBar1.Value / 100).ToString("N1")
185
186         TrackBar2.Value = 6010
187         TextBox2.Text = (TrackBar2.Value / 100).ToString("N1")
188
189         TrackBar3.Value = 2060
190         TextBox3.Text = (TrackBar3.Value / 100).ToString("N1")
191     End Sub
192
193
194
195
196
197
198     Private Sub Button3_Click(sender As Object, e As EventArgs) Handles
           Button3.Click
199         ' Viser en melding om at skriptet kjøres.
200         MessageBox.Show("Simulation have been started, please wait..")
201
202         ' Utfør Button3-operasjoner.
203         Dim journalFile As String = "D:\Program Files\Siemens\NX2206
              \NXBIN\run_journal.exe"
204         Dim openArgs As String = """D:\Program Files\Siemens\NX2206
              \NXBIN\material.vb"""
205         Dim geoArgs As String = """D:\Program Files\Siemens\NX2206
              \NXBIN\geometri.vb"""
206         Dim simArgs As String = """D:\Program Files\Siemens\NX2206
              \NXBIN\simulering.vb"""
207
208         Dim openInfo As New ProcessStartInfo(journalFile, openArgs)
209         openInfo.CreateNoWindow = True
210         openInfo.UseShellExecute = False
211
212         Dim geoInfo As New ProcessStartInfo(journalFile, geoArgs)
213         geoInfo.CreateNoWindow = True
214         geoInfo.UseShellExecute = False
215
216         Dim simInfo As New ProcessStartInfo(journalFile, simArgs)
217         simInfo.CreateNoWindow = True
218         simInfo.UseShellExecute = False
219
```

C:\Menu\Material assigner.vb 6

```
220         Dim openProcess As Process = Process.Start(openInfo)
221         openProcess.WaitForExit() ' Venter for material.vb til å
              fullføre før den kjører geometri.vb
222
223         Dim geoProcess As Process = Process.Start(geoInfo)
224         geoProcess.WaitForExit() ' Venter for geometri.vb til å
              fullføre før simulering kan kjøres
225
226         Dim simProcess As Process = Process.Start(simInfo)
227         simProcess.WaitForExit() ' Venter for simulering.vb til å
              fullføre før Button4 kan kjøres
228
229         ' Utfører Button4-operasjoner.
230         Dim scriptToRun As String = "C:\Menu\20simINIT.py"
231
232         Dim pythonPath As String = Nothing
233         For Each userDir As String In Directory.GetDirectories("C:
              \Users")
234             Dim pythonExe As String = Path.Combine(userDir, "AppData
                  \Local\Programs\Python\Python39\python.exe") ' lokasjon
                  hvor python.exe bruker å være
235             If File.Exists(pythonExe) Then
236                 pythonPath = pythonExe
237                 Exit For
238             End If
239         Next
240
241         If pythonPath Is Nothing Then
242             MessageBox.Show("Could not find python.exe in C:\Users")
243             Return
244         End If
245
246         Dim p As New Process
247         p.StartInfo.FileName = pythonPath
248         p.StartInfo.Arguments = scriptToRun
249         p.StartInfo.UseShellExecute = False
250         p.StartInfo.CreateNoWindow = True
251         p.StartInfo.RedirectStandardOutput = True
252         p.Start()
253
254         Dim output As String = p.StandardOutput.ReadToEnd()
255         MessageBox.Show(output)
256
257         ' Viser en melding om at simulering er ferdig.
258         MessageBox.Show("Simulation finished.")
259     End Sub
260
261 End Class
262
263
264
265
266
```

Vedlegg I.2. Geometri-kode

D:\Program Files\Siemens\NX2206\NXBIN\Geometri.vb 1

```
1  ' NX 2206
2  ' Skript for forandring av utkapp i veivtapp tilhørende XA-150. Det
     leses 3 .txt filer fra C:\Menu\, og verdiene her settes som verdiene i
      veivtapp filen. Verdier i veivtapp filen må være parametrisert.
3
4  Option Strict Off
5  Imports System
6  Imports NXOpen
7  Imports System.IO
8
9  Module NXJournal
10 Sub Main (ByVal args() As String)
11
12         Dim theSession As NXOpen.Session = NXOpen.Session.GetSession()
13         '------------------------------------------------
14         ' Material lagres som streng fra tekstfil
15         '------------------------------------------------
16         Dim Depth As String
17         Dim FilePath1 As String = "C:\Menu\Depth.txt"
18         Using reader As StreamReader = New StreamReader(FilePath1)
19             Depth = reader.ReadLine
20
21         End Using
22
23         Dim Width As String
24         Dim FilePath2 As String = "C:\Menu\Width.txt"
25         Using reader As StreamReader = New StreamReader(FilePath2)
26             Width = reader.ReadLine
27
28         End Using
29
30         Dim Height As String
31         Dim FilePath3 As String = "C:\Menu\Height.txt"
32         Using reader As StreamReader = New StreamReader(FilePath3)
33             Height = reader.ReadLine
34
35         End Using
36
37         ' ---------------------------------------------
38         '   Fil åpnes fra lokasjon
39         ' ---------------------------------------------
40         Dim basePart1 As NXOpen.BasePart = Nothing
41         Dim partLoadStatus1 As NXOpen.PartLoadStatus = Nothing
42         basePart1 = theSession.Parts.OpenActiveDisplay("C:\Menu
             \Kompressor\20228_Kontravekt_for_akselpumpe_01_A.prt",
             NXOpen.DisplayPartOption.AllowAdditional, partLoadStatus1)
43
44         Dim workPart As NXOpen.Part = theSession.Parts.Work
45
46         Dim displayPart As NXOpen.Part = theSession.Parts.Display
47
48         partLoadStatus1.Dispose()
49         theSession.ApplicationSwitchImmediate("UG_APP_MODELING")
```

D:\Program Files\Siemens\NX2206\NXBIN\Geometri.vb 2

```
50
51          theSession.CleanUpFacetedFacesAndEdges()
52
53          ' ------------------------------------------------
54          '   Finner variabler i eksisterende mål
55          '------------------------------------------------
56
57          Dim expression1 As NXOpen.Expression = CType
              (workPart.Expressions.FindObject("Height"), NXOpen.Expression)
58          Dim unit1 As NXOpen.Unit = CType
              (workPart.UnitCollection.FindObject("MilliMeter"),
              NXOpen.Unit)
59          workPart.Expressions.EditExpressionWithUnits(expression1, unit1,
               Height)
60
61          Dim expression2 As NXOpen.Expression = CType
              (workPart.Expressions.FindObject("Depth"), NXOpen.Expression)
62          workPart.Expressions.EditExpressionWithUnits(expression2, unit1,
               Depth)
63
64          Dim expression3 As NXOpen.Expression = CType
              (workPart.Expressions.FindObject("Width"), NXOpen.Expression)
65          workPart.Expressions.EditExpressionWithUnits(expression3, unit1,
               Width)
66
67          '------------------------------------------------
68          '    Variabler deklareres som nye verdier
69          '------------------------------------------------
70
71          Dim objects1(2) As NXOpen.NXObject
72          objects1(0) = expression1
73          objects1(1) = expression2
74          objects1(2) = expression3
75
76          theSession.CleanUpFacetedFacesAndEdges()
77
78          ' ------------------------------------------------
79          '    Modell lagres
80          ' ------------------------------------------------
81          Dim partSaveStatus1 As NXOpen.PartSaveStatus = Nothing
82          partSaveStatus1 = workPart.Save
              (NXOpen.BasePart.SaveComponents.True,
              NXOpen.BasePart.CloseAfterSave.False)
83
84          partSaveStatus1.Dispose()
85
86      End Sub
87  End Module
88
```

Vedlegg I.3. Material-kode

D:\Program Files\Siemens\NX2206\NXBIN\Geometri.vb 1

```
1  ' NX 2206
2  ' Skript for forandring av utkapp i veivtapp tilhørende XA-150. Det
     leses 3 .txt filer fra C:\Menu\, og verdiene her settes som verdiene i
      veivtapp filen. Verdier i veivtapp filen må være parametrisert.
3
4  Option Strict Off
5  Imports System
6  Imports NXOpen
7  Imports System.IO
8
9  Module NXJournal
10 Sub Main (ByVal args() As String)
11
12         Dim theSession As NXOpen.Session = NXOpen.Session.GetSession()
13         '-------------------------------------------------
14         ' Material lagres som streng fra tekstfil
15         '-------------------------------------------------
16         Dim Depth As String
17         Dim FilePath1 As String = "C:\Menu\Depth.txt"
18         Using reader As StreamReader = New StreamReader(FilePath1)
19             Depth = reader.ReadLine
20
21         End Using
22
23         Dim Width As String
24         Dim FilePath2 As String = "C:\Menu\Width.txt"
25         Using reader As StreamReader = New StreamReader(FilePath2)
26             Width = reader.ReadLine
27
28         End Using
29
30         Dim Height As String
31         Dim FilePath3 As String = "C:\Menu\Height.txt"
32         Using reader As StreamReader = New StreamReader(FilePath3)
33             Height = reader.ReadLine
34
35         End Using
36
37         ' ------------------------------------------------
38         '   Fil åpnes fra lokasjon
39         ' ------------------------------------------------
40         Dim basePart1 As NXOpen.BasePart = Nothing
41         Dim partLoadStatus1 As NXOpen.PartLoadStatus = Nothing
42         basePart1 = theSession.Parts.OpenActiveDisplay("C:\Menu
             \Kompressor\20228_Kontravekt_for_akselpumpe_01_A.prt",
             NXOpen.DisplayPartOption.AllowAdditional, partLoadStatus1)
43
44         Dim workPart As NXOpen.Part = theSession.Parts.Work
45
46         Dim displayPart As NXOpen.Part = theSession.Parts.Display
47
48         partLoadStatus1.Dispose()
49         theSession.ApplicationSwitchImmediate("UG_APP_MODELING")
```

D:\Program Files\Siemens\NX2206\NXBIN\Geometri.vb 2

```
50
51          theSession.CleanUpFacetedFacesAndEdges()
52
53          ' ------------------------------------------------
54          '   Finner variabler i eksisterende mål
55          '------------------------------------------------
56
57          Dim expression1 As NXOpen.Expression = CType
              (workPart.Expressions.FindObject("Height"), NXOpen.Expression)
58          Dim unit1 As NXOpen.Unit = CType
              (workPart.UnitCollection.FindObject("MilliMeter"),
              NXOpen.Unit)
59          workPart.Expressions.EditExpressionWithUnits(expression1, unit1,
               Height)
60
61          Dim expression2 As NXOpen.Expression = CType
              (workPart.Expressions.FindObject("Depth"), NXOpen.Expression)
62          workPart.Expressions.EditExpressionWithUnits(expression2, unit1,
               Depth)
63
64          Dim expression3 As NXOpen.Expression = CType
              (workPart.Expressions.FindObject("Width"), NXOpen.Expression)
65          workPart.Expressions.EditExpressionWithUnits(expression3, unit1,
               Width)
66
67          '------------------------------------------------
68          '    Variabler deklareres som nye verdier
69          '------------------------------------------------
70
71          Dim objects1(2) As NXOpen.NXObject
72          objects1(0) = expression1
73          objects1(1) = expression2
74          objects1(2) = expression3
75
76          theSession.CleanUpFacetedFacesAndEdges()
77
78          ' ------------------------------------------------
79          '    Modell lagres
80          ' ------------------------------------------------
81          Dim partSaveStatus1 As NXOpen.PartSaveStatus = Nothing
82          partSaveStatus1 = workPart.Save
              (NXOpen.BasePart.SaveComponents.True,
              NXOpen.BasePart.CloseAfterSave.False)
83
84          partSaveStatus1.Dispose()
85
86      End Sub
87  End Module
88
```

Vedlegg I.4. Simulering-kode

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 1

```
1  ' NX 2206
2  ' Journal created by Øyvind on Tue Apr  4 12:47:38 2023 Vest-Europa
     (sommertid)
3
4  '
5  Imports System
6  Imports NXOpen
7
8  Module NXJournal
9  Sub Main (ByVal args() As String)
10
11 Dim theSession As NXOpen.Session = NXOpen.Session.GetSession()
12 ' ----------------------------------------------
13 'Åpner simulerings fil
14 ' ----------------------------------------------
15 Dim basePart1 As NXOpen.BasePart = Nothing
16 Dim partLoadStatus1 As NXOpen.PartLoadStatus = Nothing
17         basePart1 = theSession.Parts.OpenActiveDisplay("C:\Menu
             \Kompressor\Simulering_ny.sim",
             NXOpen.DisplayPartOption.AllowAdditional, partLoadStatus1)
18         Dim workPart As NXOpen.Part = theSession.Parts.Work
19 Dim displayPart As NXOpen.Part = theSession.Parts.Display
20 partLoadStatus1.Dispose()
21 theSession.ApplicationSwitchImmediate("UG_APP_MECHANISMS")
22         ' ----------------------------------------------
23         '   Starter Solve
24         ' ----------------------------------------------
25         theSession.MotionSession.InitializeSimulation(workPart)
26
27         Dim nullNXOpen_Motion_MotionSolution As
             NXOpen.Motion.MotionSolution = Nothing
28
29         Dim motionSolution1 As NXOpen.Motion.MotionSolution = CType
             (workPart.MotionManager.MotionSolutions.FindObject
             ("Solution001"), NXOpen.Motion.MotionSolution)
30
31 motionSolution1.SolveNormalRunSolution()
32
33 theSession.CleanUpFacetedFacesAndEdges()
34
35 Dim globalSelectionBuilder2 As NXOpen.Motion.GlobalSelectionBuilder =
     Nothing
36 globalSelectionBuilder2 =
     theSession.MotionSession.MotionMethods.GetGlobalSelectionBuilder
     (workPart)
37
38 Dim selectTaggedObjectList1 As NXOpen.SelectTaggedObjectList = Nothing
39 selectTaggedObjectList1 = globalSelectionBuilder2.Selection
40
41 Dim added1 As Boolean = Nothing
42 added1 = selectTaggedObjectList1.Add(motionSolution1)
43
44 Dim globalSelectionBuilder3 As NXOpen.Motion.GlobalSelectionBuilder =
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 2

```
    Nothing
45  globalSelectionBuilder3 =                                          ↵
      theSession.MotionSession.MotionMethods.GetGlobalSelectionBuilder ↵
      (workPart)
46
47  Dim selectTaggedObjectList2 As NXOpen.SelectTaggedObjectList = Nothing
48  selectTaggedObjectList2 = globalSelectionBuilder3.Selection
49
50  Dim removed1 As Boolean = Nothing
51  removed1 = selectTaggedObjectList2.Remove(motionSolution1)
52
53  Dim globalSelectionBuilder4 As NXOpen.Motion.GlobalSelectionBuilder = ↵
      Nothing
54  globalSelectionBuilder4 =                                          ↵
      theSession.MotionSession.MotionMethods.GetGlobalSelectionBuilder ↵
      (workPart)
55
56  Dim selectTaggedObjectList3 As NXOpen.SelectTaggedObjectList = Nothing
57  selectTaggedObjectList3 = globalSelectionBuilder4.Selection
58
59  Dim joint1 As NXOpen.Motion.Joint = CType                          ↵
      (workPart.MotionManager.Joints.FindObject("Drive"),               ↵
      NXOpen.Motion.Joint)
60
61  Dim added2 As Boolean = Nothing
62  added2 = selectTaggedObjectList3.Add(joint1)
63
64  Dim markId1 As NXOpen.Session.UndoMarkId = Nothing
65  markId1 = theSession.SetUndoMark                                   ↵
      (NXOpen.Session.MarkVisibility.Invisible, Nothing)
66
67
68  '''''''''''''''''''''''''
69  'Her lages FX graf
70  '''''''''''''''''''''''''
71
72  Dim nullNXOpen_Motion_Graph As NXOpen.Motion.Graph = Nothing
73
74  Dim graphObjectBuilder1 As NXOpen.Motion.GraphObjectBuilder = Nothing
75  graphObjectBuilder1 = motionSolution1.Graphs.CreateGraphObjectBuilder ↵
      (nullNXOpen_Motion_Graph)
76
77  graphObjectBuilder1.ReferenceObject = joint1
78
79  graphObjectBuilder1.SolutionObject = motionSolution1
80
81  graphObjectBuilder1.SetQuantityType("Component", "FX")
82
83  graphObjectBuilder1.SetQuantityType("CSYS", "abs")
84
85  graphObjectBuilder1.SetQuantityType("Request", "Force")
86
87  Dim nullNXOpen_CoordinateSystem As NXOpen.CoordinateSystem = Nothing
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 3

```
 88
 89 graphObjectBuilder1.ReferencedCsys = nullNXOpen_CoordinateSystem
 90
 91 Dim graph1 As NXOpen.Motion.Graph = Nothing
 92 graph1 = graphObjectBuilder1.FindEquivalent()
 93
 94 Dim nXObject1 As NXOpen.NXObject = Nothing
 95 nXObject1 = graphObjectBuilder1.Commit()
 96
 97 graphObjectBuilder1.Destroy()
 98
 99 Dim graph2 As NXOpen.Motion.Graph = CType(nXObject1,
      NXOpen.Motion.Graph)
100
101 graph2.SetIsTemporary(True)
102
103 Dim markId2 As NXOpen.Session.UndoMarkId = Nothing
104 markId2 = theSession.SetUndoMark(NXOpen.Session.MarkVisibility.Visible,
      "Start")
105
106 theSession.SetUndoMarkName(markId2, "Viewport Dialog")
107
108 Dim objects1(0) As NXOpen.TaggedObject
109 objects1(0) = joint1
110 Dim removed2 As Boolean = Nothing
111 removed2 = selectTaggedObjectList3.RemoveArray(objects1)
112
113 theSession.DeleteUndoMark(markId2, Nothing)
114
115 Dim ygraphs1(0) As NXOpen.Motion.Graph
116 ygraphs1(0) = graph2
117 Dim plot1 As NXOpen.CAE.Xyplot.Plot = Nothing
118 plot1 = motionSolution1.PlotGraphObjectsInViewport
      (nullNXOpen_Motion_Graph, ygraphs1, 1, 0, False)
119
120 Dim globalSelectionBuilder5 As NXOpen.Motion.GlobalSelectionBuilder =
      Nothing
121 globalSelectionBuilder5 =
      theSession.MotionSession.MotionMethods.GetGlobalSelectionBuilder
      (workPart)
122
123 Dim selectTaggedObjectList4 As NXOpen.SelectTaggedObjectList = Nothing
124 selectTaggedObjectList4 = globalSelectionBuilder5.Selection
125
126 Dim added3 As Boolean = Nothing
127 added3 = selectTaggedObjectList4.Add(joint1)
128
129 Dim id1 As NXOpen.Session.UndoMarkId = Nothing
130 id1 = theSession.NewestVisibleUndoMark
131
132 Dim nErrs1 As Integer = Nothing
133 nErrs1 = theSession.UpdateManager.AddToDeleteList(graph2)
134
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 4

```
135 Dim nErrs2 As Integer = Nothing
136 nErrs2 = theSession.UpdateManager.DoUpdate(id1)
137
138 theSession.DeleteUndoMark(markId1, Nothing)
139
140 Dim plot2D1 As NXOpen.CAE.Xyplot.Plot2D = CType(plot1,
      NXOpen.CAE.Xyplot.Plot2D)
141
142 Dim basicModel1 As NXOpen.CAE.Xyplot.BasicModel = CType
      (plot2D1.FindObject("Graph[0][0]_DiagramContainer[0]"),
      NXOpen.CAE.Xyplot.BasicModel)
143
144 Dim iDisplayStyle1 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
145 iDisplayStyle1 = basicModel1.DisplayStyle
146
147 Dim basicModel2 As NXOpen.CAE.Xyplot.BasicModel = CType
      (plot2D1.FindObject("LegendItem[0]"), NXOpen.CAE.Xyplot.BasicModel)
148
149 Dim iDisplayStyle2 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
150 iDisplayStyle2 = basicModel2.DisplayStyle
151
152 Dim iDisplayStyle3 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
153 iDisplayStyle3 = basicModel1.DisplayStyle
154
155 Dim iDisplayStyle4 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
156 iDisplayStyle4 = basicModel1.DisplayStyle
157
158 Dim iDisplayStyle5 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
159 iDisplayStyle5 = basicModel2.DisplayStyle
160
161 Dim iDisplayStyle6 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
162 iDisplayStyle6 = basicModel1.DisplayStyle
163
164 Dim displayDataExportParameters1 As
      NXOpen.CAE.Xyplot.DisplayDataExportParameters = Nothing
165 displayDataExportParameters1 =
      theSession.XYPlotManager.DataExporter.NewDisplayDataExportParameters
      ()
166
167 Dim plots1(0) As NXOpen.CAE.Xyplot.Plot
168 plots1(0) = plot2D1
169 displayDataExportParameters1.SetPlots(plots1)
170
171 Dim plotdataindices1(0) As Integer
172 plotdataindices1(0) = 0
173 displayDataExportParameters1.SetRecordIndices(plotdataindices1)
174
175 Dim graphindices1(0) As Integer
176 graphindices1(0) = 0
177 displayDataExportParameters1.SetGraphIndices(graphindices1)
178
179 Dim exportFilesParameter1 As NXOpen.CAE.FTK.ExportFilesParameter =
      Nothing
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 5

```
180 exportFilesParameter1 =
      theSession.XYPlotManager.DataExporter.NewExportFilesParameters()
181
182 Dim markId3 As NXOpen.Session.UndoMarkId = Nothing
183 markId3 = theSession.SetUndoMark(NXOpen.Session.MarkVisibility.Visible,
      "Start")
184
185 Dim numberFormat1 As NXOpen.CAE.NumberFormat = Nothing
186 numberFormat1 = exportFilesParameter1.NumberFormat
187
188 theSession.SetUndoMarkName(markId3, "Export Data Dialog")
189
190 ' ------------------------------------------------
191 'Data exporteres til C:\Menu\CSV\FX.csv
192 ' ------------------------------------------------
193 Dim objects2(0) As NXOpen.TaggedObject
194 objects2(0) = joint1
195 Dim removed3 As Boolean = Nothing
196 removed3 = selectTaggedObjectList4.RemoveArray(objects2)
197
198 Dim markId5 As NXOpen.Session.UndoMarkId = Nothing
199 markId5 = theSession.SetUndoMark
      (NXOpen.Session.MarkVisibility.Invisible, "Export Data")
200
201 exportFilesParameter1.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.FileDescription,
      False)
202
203 exportFilesParameter1.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.General, False)
204
205 exportFilesParameter1.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.Abscissa, False)
206
207 exportFilesParameter1.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.Ordinate, False)
208
209 exportFilesParameter1.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.AbscissaZ, False)
210
211 exportFilesParameter1.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.ApplicationSpecifi
      c, False)
212
213 exportFilesParameter1.OverrideSetting =
      NXOpen.CAE.FTK.ExportFilesParameter.OverrideOption.Replace
214
215 ''''''''''''''''''''
216 'Lagringslokasjon defineres
217 ''''''''''''''''''''
218
219 Dim filenames1(0) As String
220 filenames1(0) = "C:\Menu\CSV\FX.csv"
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 6

```
221 exportFilesParameter1.SetFileNames(filenames1)
222
223 theSession.DeleteUndoMark(markId5, Nothing)
224
225 theSession.SetUndoMarkName(markId3, "Export Data")
226
227 theSession.XYPlotManager.DataExporter.ExportToFiles
      (displayDataExportParameters1, exportFilesParameter1)
228
229 exportFilesParameter1.Dispose()
230 displayDataExportParameters1.Dispose()
231 theSession.CleanUpFacetedFacesAndEdges()
232
233 Dim globalSelectionBuilder6 As NXOpen.Motion.GlobalSelectionBuilder =
      Nothing
234 globalSelectionBuilder6 =
      theSession.MotionSession.MotionMethods.GetGlobalSelectionBuilder
      (workPart)
235
236 Dim selectTaggedObjectList5 As NXOpen.SelectTaggedObjectList = Nothing
237 selectTaggedObjectList5 = globalSelectionBuilder6.Selection
238
239 Dim added4 As Boolean = Nothing
240 added4 = selectTaggedObjectList5.Add(joint1)
241
242 Dim graphObjectBuilder2 As NXOpen.Motion.GraphObjectBuilder = Nothing
243 graphObjectBuilder2 = motionSolution1.Graphs.CreateGraphObjectBuilder
      (nullNXOpen_Motion_Graph)
244
245 graphObjectBuilder2.ReferenceObject = joint1
246
247 graphObjectBuilder2.SolutionObject = motionSolution1
248
249 graphObjectBuilder2.SetQuantityType("Component", "FZ")
250
251 graphObjectBuilder2.SetQuantityType("CSYS", "abs")
252
253 graphObjectBuilder2.SetQuantityType("Request", "Force")
254
255 graphObjectBuilder2.ReferencedCsys = nullNXOpen_CoordinateSystem
256
257 Dim graph3 As NXOpen.Motion.Graph = Nothing
258 graph3 = graphObjectBuilder2.FindEquivalent()
259
260 Dim nXObject2 As NXOpen.NXObject = Nothing
261 nXObject2 = graphObjectBuilder2.Commit()
262
263 graphObjectBuilder2.Destroy()
264
265 Dim graph4 As NXOpen.Motion.Graph = CType(nXObject2,
      NXOpen.Motion.Graph)
266
267 graph4.SetIsTemporary(True)
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 7

```
268
269 Dim markId7 As NXOpen.Session.UndoMarkId = Nothing
270 markId7 = theSession.SetUndoMark(NXOpen.Session.MarkVisibility.Visible,
       "Start")
271
272 theSession.SetUndoMarkName(markId7, "Viewport Dialog")
273
274 ' ------------------------------------------------
275 '   Dialog Begin Viewport
276 ' ------------------------------------------------
277 Dim objects3(0) As NXOpen.TaggedObject
278 objects3(0) = joint1
279 Dim removed4 As Boolean = Nothing
280 removed4 = selectTaggedObjectList5.RemoveArray(objects3)
281
282 theSession.DeleteUndoMark(markId7, Nothing)
283
284 Dim ygraphs2(0) As NXOpen.Motion.Graph
285 ygraphs2(0) = graph4
286 Dim plot2 As NXOpen.CAE.Xyplot.Plot = Nothing
287 plot2 = motionSolution1.PlotGraphObjectsInViewport
      (nullNXOpen_Motion_Graph, ygraphs2, 1, 0, False)
288
289 Dim globalSelectionBuilder7 As NXOpen.Motion.GlobalSelectionBuilder =
      Nothing
290 globalSelectionBuilder7 =
      theSession.MotionSession.MotionMethods.GetGlobalSelectionBuilder
      (workPart)
291
292 Dim selectTaggedObjectList6 As NXOpen.SelectTaggedObjectList = Nothing
293 selectTaggedObjectList6 = globalSelectionBuilder7.Selection
294
295 Dim added5 As Boolean = Nothing
296 added5 = selectTaggedObjectList6.Add(joint1)
297
298 Dim id2 As NXOpen.Session.UndoMarkId = Nothing
299 id2 = theSession.NewestVisibleUndoMark
300
301 Dim nErrs3 As Integer = Nothing
302 nErrs3 = theSession.UpdateManager.AddToDeleteList(graph4)
303
304 Dim nErrs4 As Integer = Nothing
305 nErrs4 = theSession.UpdateManager.DoUpdate(id2)
306
307
308
309 Dim plot2D2 As NXOpen.CAE.Xyplot.Plot2D = CType(plot2,
      NXOpen.CAE.Xyplot.Plot2D)
310
311 Dim basicModel3 As NXOpen.CAE.Xyplot.BasicModel = CType
      (plot2D2.FindObject("Graph[0][0]_DiagramContainer[0]"),
      NXOpen.CAE.Xyplot.BasicModel)
312
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 8

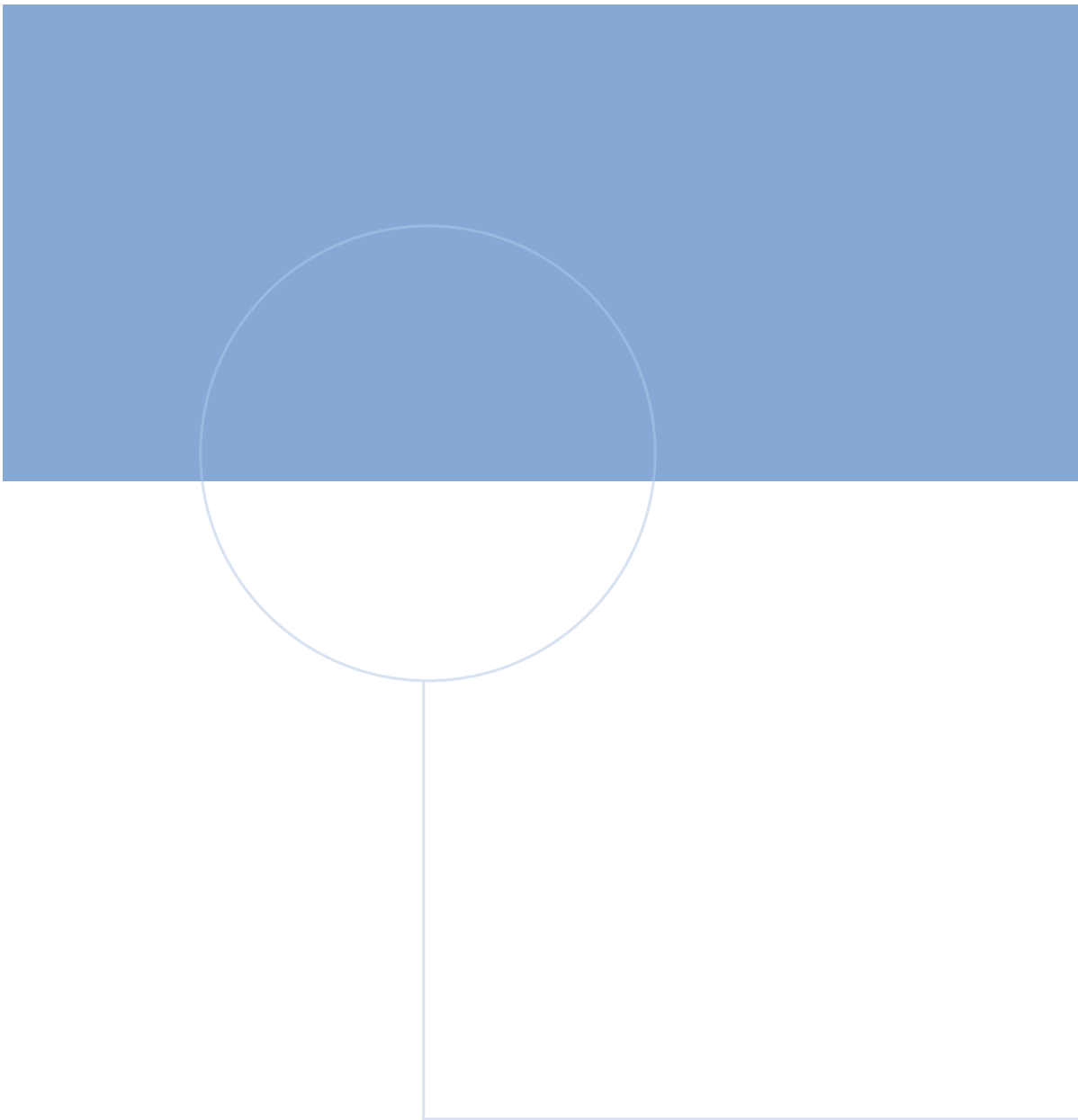
```
313 Dim iDisplayStyle7 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
314 iDisplayStyle7 = basicModel3.DisplayStyle
315
316 Dim basicModel4 As NXOpen.CAE.Xyplot.BasicModel = CType
      (plot2D2.FindObject("LegendItem[0]"), NXOpen.CAE.Xyplot.BasicModel)
317
318 Dim iDisplayStyle8 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
319 iDisplayStyle8 = basicModel4.DisplayStyle
320
321 Dim iDisplayStyle9 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
322 iDisplayStyle9 = basicModel3.DisplayStyle
323
324 Dim iDisplayStyle10 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
325 iDisplayStyle10 = basicModel3.DisplayStyle
326
327 Dim iDisplayStyle11 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
328 iDisplayStyle11 = basicModel4.DisplayStyle
329
330 Dim iDisplayStyle12 As NXOpen.CAE.Xyplot.IDisplayStyle = Nothing
331 iDisplayStyle12 = basicModel3.DisplayStyle
332
333 Dim displayDataExportParameters2 As
      NXOpen.CAE.Xyplot.DisplayDataExportParameters = Nothing
334 displayDataExportParameters2 =
      theSession.XYPlotManager.DataExporter.NewDisplayDataExportParameters
      ()
335
336 Dim plots2(0) As NXOpen.CAE.Xyplot.Plot
337 plots2(0) = plot2D2
338 displayDataExportParameters2.SetPlots(plots2)
339
340 Dim plotdataindices2(0) As Integer
341 plotdataindices2(0) = 0
342 displayDataExportParameters2.SetRecordIndices(plotdataindices2)
343
344 Dim graphindices2(0) As Integer
345 graphindices2(0) = 0
346 displayDataExportParameters2.SetGraphIndices(graphindices2)
347
348 Dim exportFilesParameter2 As NXOpen.CAE.FTK.ExportFilesParameter =
      Nothing
349 exportFilesParameter2 =
      theSession.XYPlotManager.DataExporter.NewExportFilesParameters()
350
351 Dim markId8 As NXOpen.Session.UndoMarkId = Nothing
352 markId8 = theSession.SetUndoMark(NXOpen.Session.MarkVisibility.Visible,
     "Start")
353
354 Dim numberFormat2 As NXOpen.CAE.NumberFormat = Nothing
355 numberFormat2 = exportFilesParameter2.NumberFormat
356
357 theSession.SetUndoMarkName(markId8, "Export Data Dialog")
358
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 9

```
359 ' ------------------------------------------------
360 'Eksportering CSV FY
361 ' ------------------------------------------------
362 Dim objects4(0) As NXOpen.TaggedObject
363 objects4(0) = joint1
364 Dim removed5 As Boolean = Nothing
365 removed5 = selectTaggedObjectList6.RemoveArray(objects4)
366
367 Dim markId9 As NXOpen.Session.UndoMarkId = Nothing
368 markId9 = theSession.SetUndoMark
      (NXOpen.Session.MarkVisibility.Invisible, "Export Data")
369
370 theSession.DeleteUndoMark(markId9, Nothing)
371
372 Dim markId10 As NXOpen.Session.UndoMarkId = Nothing
373 markId10 = theSession.SetUndoMark
      (NXOpen.Session.MarkVisibility.Invisible, "Export Data")
374
375 exportFilesParameter2.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.FileDescription,
      False)
376
377 exportFilesParameter2.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.General, False)
378
379 exportFilesParameter2.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.Abscissa, False)
380
381 exportFilesParameter2.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.Ordinate, False)
382
383 exportFilesParameter2.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.AbscissaZ, False)
384
385 exportFilesParameter2.SetIsCertainHeaderExportRequired
      (NXOpen.CAE.FTK.ExportFilesParameter.HeaderOptions.ApplicationSpecifi
      c, False)
386
387 exportFilesParameter2.OverrideSetting =
      NXOpen.CAE.FTK.ExportFilesParameter.OverrideOption.Replace
388
389 Dim filenames2(0) As String
390 filenames2(0) = "C:\Menu\CSV\FY.csv"
391 exportFilesParameter2.SetFileNames(filenames2)
392
393 theSession.DeleteUndoMark(markId10, Nothing)
394
395 theSession.SetUndoMarkName(markId8, "Export Data")
396
397 theSession.XYPlotManager.DataExporter.ExportToFiles
      (displayDataExportParameters2, exportFilesParameter2)
398
399 exportFilesParameter2.Dispose()
```

D:\Program Files\Siemens\NX2206\NXBIN\Simulering.vb 10

```
400  displayDataExportParameters2.Dispose()
401  theSession.CleanUpFacetedFacesAndEdges()
402
403
404  End Sub
405  End Module
406
```



NTNU

Kunnskap for en bedre verden