NTNU
Det skapende universitet

CAM-programvare for CNC-programmering

Alexander John Andfossen

Master i produktutvikling og produksjon
Innlevert: juni 2017
Hovedveileder: Knut Sørby, MTP

Norges teknisk-naturvitenskapelige universitet
Institutt for maskinteknikk og produksjon

Forord

Denne masteroppgaven er skrevet av stud. techn. Alexander John Andfossen og er utført vårsemesteret 2017. Oppgaven tilsvarer 30 av totalt 300 studiepoeng av en 5-årig sivilingeniørutdanning i produktutvikling og produksjon med retningen produksjonsteknologi, ved Institutt for Maskinteknikk og Produksjon, NTNU.

Jeg vil takke min veileder Professor Knut Sørby for utforming av oppgaven og god støtte underveis i prosjektet og hjelpen med å utvikle postprosessorer. Jeg vil også takke Arild Sæther for tilgang til verkstedet ved MTP Valgrinda. Til slutt vil jeg takke Eyvinn Høvring Mikkelsen som har vært hjelpsom med sin kunnskap i Autodesk Fusion 360 og for hjelpen med å operere CNC-maskinene brukt i dette prosjektet.

Trondheim, 11.06.2017

Alexander John Andfossen

Summary

Autodesk Fusion 360 was introduced in 2013 and has since then become the employee's preferred software for CAD/CAM at the workshop at MTP Valgrinda, NTNU. A postprocessor for Fusion 360 CAM that generates instructions for the 5-axis machine at the workshop has not been available. The employees have therefore preferred to use the advanced 3- and 4- axis machines, which can be controlled by machine code generated by Fusion 360 CAM.

A postprocessor connects the CAM to a CNC machine, enabling tool path information defined in the CAM software to be post-processed to CNC-code that can be interpreted be the CNC-machine's control system. Autodesk has developed a postprocessor that adapts to most control languages in the marked. However, since the specifications of the various CNC-machines differ, the postprocessors must often be customized to the specific CNC-machine.

Fusion 360 is available to students free of charge, and works on PC, Mac and mobile phone operating systems. This makes it well suited for educational purposes at NTNU. Professor Knut Sørby suggested to investigate how Autodesk Fusion 360 could be introduced to the students at NTNU, both as an introduction to Fusion 360 CAD and CAM with simple 3-axis milling, and also as a more advanced CAD/CAM exercise with 5-axis milling.

In this thesis, the source code of mach3mill.cps - a postprocessor developed by Autodesk, was improved and developed to generate CNC-code from Fusion 360 CAM that adapts to the Proxxon-FF 500 CNC machine with steering language Mach3. This CNC-machine is used as part of a student project in the course TPK4190 – Production Technology at NTHU, given by Professor Sørby. Also, as part of this thesis, a user manual (tutorial) meant for students was made on how to use Fusion 360 CAD and CAM to craft the workpiece used in the student project.

For a more advanced program, it was focused on developing a post processor to Fusion 360 CAM that can generate machine code to the 5-axis machine Deckel Maho DMU 50 eVolution. Autodesk has developed a post processor called Heidenhain iso.cps. This post processor was used as the base for further development into a postprocessor that calculates tool paths by equations from the inverse kinematics of Deckel Maho DMU 50 eVolution. In order for Heidenhain iso.cps to handle occurrences of singularities without the tool colliding, a singularity control was developed and coded into the postprocessor using equations from the direct- and inverse kinematics of Deckel Maho DMU 50 eVolution. Also, a linearization control was developed that utilizes the equations from direct kinematics to check whether the rotational axes of the machine will give a deviation between two points at the tool path larger than a given tolerance. If so, an interpolation will be made between the two points on the tool path.

The accuracy of Deckel Maho DMU 50 eVolution was improved by calibrating a new machine base coordinate system.

Several practical tests on various models with simultaneous 5-axis milling, with tool paths generated by the developed postprocessor for Fusion 360, were conducted with satisfactory re-

sults. It was also checked that the machine code generated by the postprocessor could handle a singularity without colliding with the workpiece. The Mobius ring, presumed to be a challenging model to mill, was made with a satisfactory result.

A user manual (tutorial) on how to model and program tool paths followed by post processing to machine code that adapts to Deckel Maho DMU 50 eVolution was developed. This is meant to give the Ph.D students under guidance of Professor Sørby an introduction to Autodesk Fusion 360 CAD and CAM with different simultaneous 5-axis functions.

Sammendrag

Autodesk Fusion 360 ble introdusert i 2013 og har siden utviklet seg til å bli den foretrukne programvaren for CAD/CAM blant de ansatte på verkstedet på MTP Valgrinda, NTNU. Det har frem til nå manglet en postprosessor til Fusion 360 CAM som genererer maskinkode for 5-aksemaskinen Deckel Maho DMU 50 eVolution på verkstedet. Dermed har de ansatte foretrukket å bruke de avanserte 3-og 4-aksemaskinen, som kan styres med maskinkode generert fra Fusion 360 CAM.

En postprosessor er et bindeledd mellom CAM og en CNC-maskin, der informasjonen om verktøybanene definert i CAM postprosesseres til CNC-kode som kan tolkes av styringssystemet til CNC-maskinen. Autodesk har utviklet postprosessorer som er tilpasset de fleste styringsspråkene på markedet, men ettersom CNC-maskiner har ulike egenskaper må ofte postprosessorene tilpasses den tiltenkte CNC-maskinen.

Fusion 360 kommer med gratis lisens for studenter og kan brukes med operativsystemene til PC, Mac og mobil, som gjør det godt egnet som CAD/CAM-programvare til undervisning ved NTNU. Professor Knut Sørby ønsket at det skulle undersøkes hvordan Autodesk Fusion 360 kunne innføres som en del av undervisningen ved NTNU, både et opplegg for introduksjon til Fusion 360 CAD og CAM med enkel 3-aksemaskinering, samt et mer avansert CAD/CAM-opplegg med 5-aksemaskinering.

I denne oppgaven ble det gjort tilpasninger i kildekoden til en postprosessor utviklet av Autodesk med navnet *mach3mill.cps*, slik at det kunne genereres CNC-kode fra Fusion 360 CAM, tilpasset maskinen *Proxxon-FF 500 CNC* med styringsspråket *Mach3*. CNC-maskinen brukes som en del av et studentprosjekt i emnet TPK4190 - Produksjonsteknologi på NTNU, der Sørby er fagansvarlig. Som en del av denne oppgaven ble det også utviklet en manual for hvordan studentene kan bruke Fusion 360 CAD og CAM til å maskinere ut komponenten som brukes i studentprosjektet.

For et mer avansert opplegg ble det fokusert på utvikling av en postprosessor til Fusion 360 CAM, som kan generere maskinkode til 5-aksemaskinen Deckel Maho DMU 50 eVolution. Autodesk har utviklet en postprosessor med navnet *Heidenhain iso.cps*. Denne postprosessoren

ble brukt som utgangspunkt for å programmere en postprosessor som kalkulerer verktøybaner med likninger fra inverskinematikken til Deckel Maho DMU 50 eVolution. For at *Heidenhain iso.cps* skulle håndere situasjoner der det oppstår en singularitet, uten at verktøyet kolliderer, ble en singularitetskontroll programmert inn i postprosessoren ved å bruke likningene fra direkte- og inverskinematikken til Deckel Maho DMU 50 eVolution. Det ble også utviklet en linearise-ringskontroll som bruker likningene fra direkte-kinematikken for å sjekke om rotasjonsaksene i maskinen fører til et større avvik mellom to punkt i verktøybanen enn det som tolereres og eventuelt interpolerer et midtpunkt mellom de to punktene i verktøybanen.

Nøyaktigheten til Deckel Maho DMU 50 eVolution ble utbedret ved å kalibrere et nytt mask-innullpunkt.

Flere fysiske tester med simultan 5-aksemaskinering på ulike modeller med verktøybaner generert av den utviklede postprosessoren fra Fusion 360 ble utført med tilfredstillende resul-tater, blant annet en test der maskinkoden generert av postprosessoren håndterte en singu-laritet uten å kollidere med arbeidsstykket. Mobius-ringen, som er antatt å være en krevende modell å frese, ble maskinert med et tilfredstillende resultat.

For å gi doktorstipendiatene til Sørby innføring i Autodesk Fusion 360 CAD og CAM med ulike simultan 5-aksefunksjoner ble en manual for å modellere og programmere verktøybaner som så postprosesseres til maskinkode tilpasset Deckel Maho DMU 50 eVolution utviklet.

Innhold

Figurliste

Tabelliste

Kapittel 1

Innledning

1.1 Bakgrunn

På verkstedet ved ved MTP Valgrinda, NTNU produseres forskjellige komponenter til ulike formål innenfor undervisning og forskning. Blant annet bruker studentene verkstedet til å produsere en Stirlingmotor som en del av et studentprosjekt i faget TPK 4190-Produksjonsteknologi ved NTNU. Motorbasen maskineres av studentene ved å programmere verktøybaner manuelt basert på en maskintegning, for så å overføre programmet til en enkel CNC-maskin som freser med 3 akser.

De tradisjonelle CAD/CAM programvarene som brukes i forbindelse med undervisning ved NTNU krever dyre lisenser og er ofte kun for Windows datamaskiner, slik at programvarene kun er installert på datamaskiner på spesifikke datasaler på NTNU. Det er således bruk for rimeligere løsninger, der studentene kan bruke programvaren på egne datamaskiner.

Autodesk Fusion 360 er et brukervennlig CAD/CAM programvare som kan brukes gratis med studentlisens og kan installeres på plattformene Mac og PC, slik at studentene kan installere programmet på sine egne datamaskiner og modellere på egenhånd i CAD, og programmere automatiske verktøybaner i CAM.

De ansatte ved verkstedet på MTP Valgrinda foretrekker å bruke Autodesk Fusion 360 som programvare for CAD/CAM, og for å overføre de programmerte verktøybanene til CNC-maskinen må de gjennom en postprosessor som genererer maskinkode tilpasset den tiltenkte CNC- maskinen. Det manglet en postprosessor til Fusion 360 CAM som genererer NC-data for 5-aksemaskinen Deckel Maho DMU 50 eVolution, slik at de ansatte foretrekker å bruke de avanserte 3-og 4-aksemaskinene på verkstedet, som kan styres med maskinkode generert fra Fusion 360 CAM. Det er således et behov for å utvikle en postprosessor som kan generere verktøybaner fra Fusion 360 på 5-aksemaskinen.

1.2 Problemstilling

Masteroppgaven går ut på å undersøke funksjonaliteten til ulike CAM-systemer sammenliknet med Autodesk Fusion 360 med tanke på innføring i undervisning ved instituttet. Videre skal det utvikles et undervisningsopplegg med introduksjon til grunnleggende CNC-programmering, og et mer avansert opplegg for multiaksemaskinering. Det skal også utarbeides veiledninger for generering av CNC-kode (3-akse og 5-akse) ved hjelp av Autodesk Fusion 360.

For å generere CNC-kode må det utvikles en postprosessor som fungerer med Autodesk Fusion 360 på en enkel 3-aksemaskin som brukes i forbindelse med et undervisningsopplegg ved MTP. Det skal også utvikles en postprosessor for å generere maskinkode med Fusion 360 CAM som kan brukes på en avansert 5-aksemaskin ved MTP Valgrinda, det er også ønskelig å undersøke muligheten for å håndtere en singularitet og linearisere der det er nødvendig.

Videre skal nøyaktigheten til 5-aksemaskinen undersøkes ved hjelp av nullpunktstester, og muligheten for eventuelle forbedringer skal undersøkes.

Til slutt skal det gjennomføres praktiske maskineringsforsøk i 3-akse og 5-akse for å verifisere funksjonaliteten til de utviklede postprosessorene.

1.3 Rapportens oppbygning

Rapporten innleder med relevant teori for oppgaven i kapittel 2. Deretter blir prosessen med utvikling av ulike postprosessorer for Autodesk Fusion 360 forklart i kapittel 3. Kapitellet inneholder koden til funksjoner som ble programmert inn i kildekoden til postprosessorene og forklaring av kodens funksjonalitet, samt prosessen med å testkjøre de ulike endingene i kildekoden til postprosessorene på CNC-maskiner. Kapittelet avsluttes med utviklingen av en postprosessor for å generere NC-kode fra Fusion 360 til 5-aksemaskinen Deckel Maho DMU 50 eVolution. Kapittel 4 beskriver hvordan postprosessoren for Deckel Maho DMU 50 eVolution ble utviklet for å håndtere en singularitet, samt hvordan en lineariseringskontroll ble implementert. Prosedyren for å kalibrere og definere et nullpunkt er forklart i kapittel 5. Kapittel 6 beskriver hvordan verktøybanene i testmodellene ble programmert med Fusion 360-CAM og presenterer resultatet fra de fysiske maskineringstestene. Til slutt oppsummeres arbeidet i en konklusjon og en anbefaling for videre arbeid i kapittel 7.

I vedlegg bakerst i rapporten er det lagt ved to manualer utviklet av kandidaten for å bruke Fusion 360 til å modellere og programmere verktøybaner. Den første manualen er for Sterling-motorbasen som er tiltenkt å brukes i et undervisningsopplegg med introduksjon til 3-aksemaskinering (Vedlegg C). Den andre manualen er for simultan 5-aksemaskinering med ulike funksjoner i Fusion 360 som er tiltenkt doktorstipendiater (Vedlegg E).

Kapittel 2

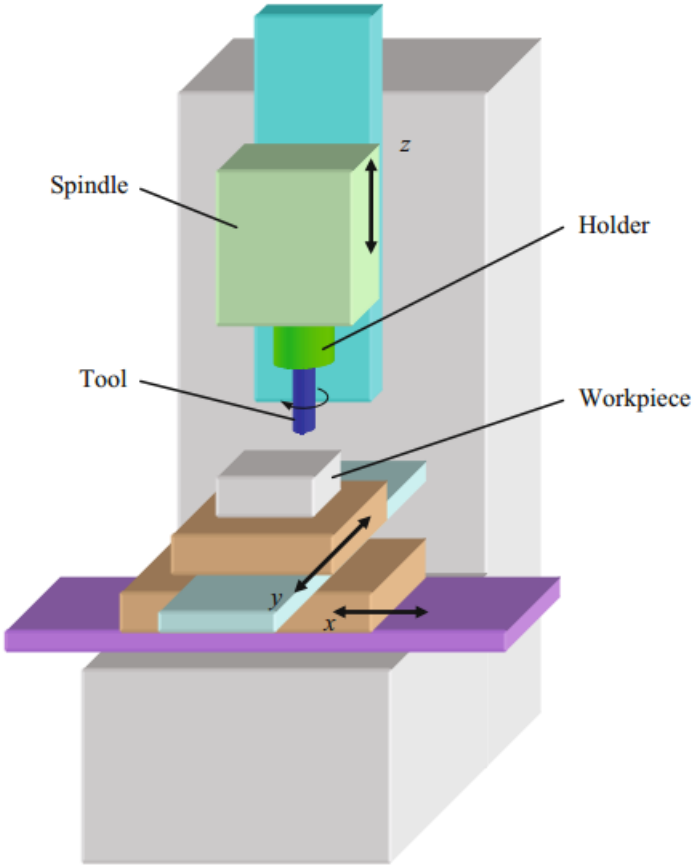
Teori

2.1 CNC maskiner

En numerisk styrt maskin, også kjent som CNC-maskin, er en sponfraskillende verktøymaskin som elektronisk styrer et roterende kuttverktøy, relativt til et oppspent arbeidstykke, for å avvirke materiale fra et råemne. Verktøyet styres etter et forhåndsutviklet program, slik at man får en ferdig modell med korrekt dimensjon og geometri. Freseverktøyet, også bare kalt fres, blir spent fast i en verktøyholder, som igjen er festet til en spindel som roterer verktøyet. Spindelen er enten vertikal eller horisontal, men det foretrukne i industrien er at spindelen er montert vertikalt mot verktøyet slik at sponet faller ned fra arbeidstykket, slik figur 2.1 illustrerer. Bevegelsene til verktøyet styres av motorer som beveger verktøyet om maskinaksene. De enkleste CNC(Computer Numerically Controlled)-maskinene kommer med 3 lineære akser, der verktøyet beveger seg om X, Y og Z-aksene(1). Akseretningene bestemmes av høyrehåndsregelen, der Z-aksen er orientert i samme retning som spindelen hvor skjærkraften blir overført. Z-aksen indikerer høyden vertikalt, der bevegelse i positiv retning gir større avstand til arbeidsstykket. X-aksen er horisontal og er parallell til arbeidsbordet, og Y-aksen bestemmes av høyrehåndsregelen. For mer avanserte CNC-maskiner kan det legges til en eller flere rotasjonsakser, der rotasjonsaksene roterer om aksene A, B eller C, som henholdsvis indikerer om rotasjonen er om X, Y eller Z-aksen(2).

I mars 1952 ble den første NC(Numerically Controlled)-maskinen med 3-akser introdusert(3). Siden den tid har industrier som fly og bilindustrien stilt krav om maskiner med stor nøyaktighet og kortere omløpstid.

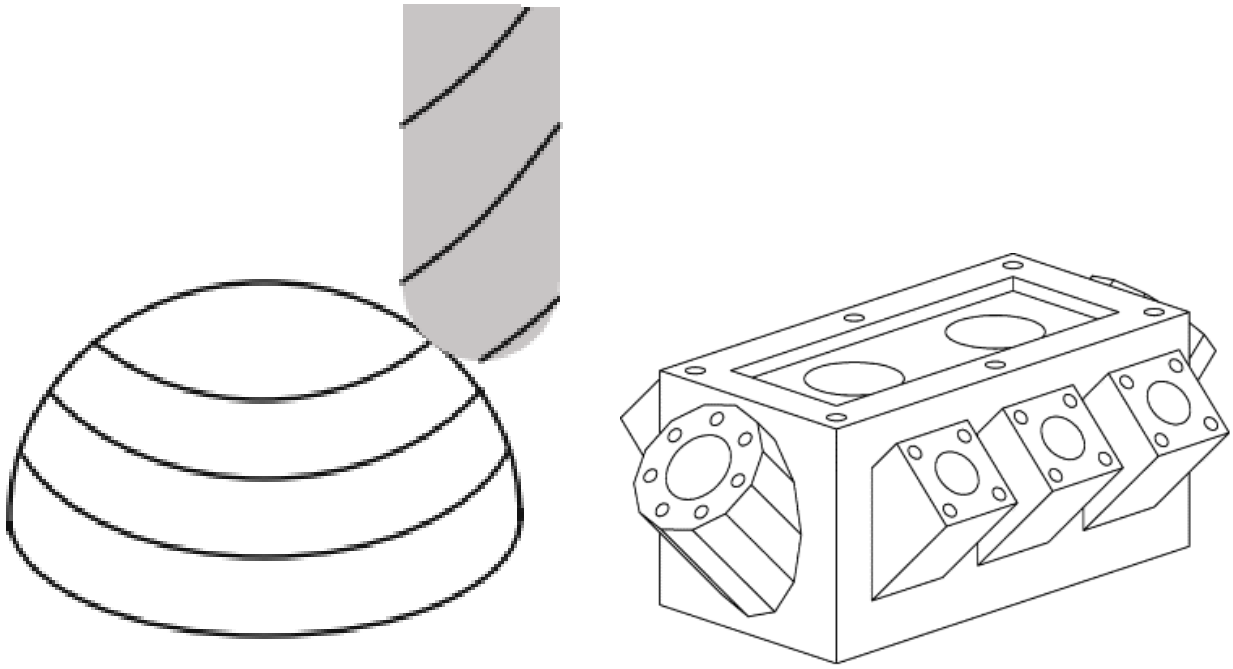
I 1979 ble den første CNC-maskinen, på norsk betegnet som databasert numerisk styringssystem, introdusert. Maskinen hadde en mikroprosessor inkludert i styringen, som gjorde det mulig å bruke styringen mer aktivt(2). Tradisjonelt ble komplekse flater maskinert med 3-akser og en radiefres slik figur 2.2a illustrerer, men med et økende behov for variabel verktøyorientering ble maskiner med rotasjonsakser i tillegg til de tre lineære aksene introdusert i 1984 (4).



Figur 2.1: 3-akse maskin med vertikal spindel(1).

Tidligere måtte arbeidsstykket spennes om manuelt for å få ny orientering i en 3-akse maskin, men med 5 akser kan et råemne avvirkes til en ferdig komponent i en oppspenning

Verktøymaskiner med 5 akser har en stor fordel i maskinering av komplekse strukturer, de to ekstra rotasjonsaksene tillater verktøyet å følge arbeidsstykkets overflatekrumning, som gir en glattere overflate med høyere nøyaktighet og reduserer behovet for sliping. En 5-aksemaskin kan bygges på flere måter. For eksempel kan en maskin ha tre lineære akser om X, Y og Z, og to rotasjonsakser i verktøyet, slik at arbeidsstykket står i ro når det maskineres.
En annen oppbygging av en 5-aksemaskin, slik som den som brukes i dette prosjektet, kan sammenliknes med to roboter, en som holder arbeidsstykket og en som holder verktøyet. Det kreves minimum fem frihetsgrader for å orientere verktøyet og arbeidstykket relativt til hverandre slik at alle orienteringer vinkelrett på en halvkule oppspent på arbeidsbordet kan håndteres (5). Figur 2.2b viser en modell med hull og flate plan i forskjellige vinkler, det er ikke mulig å maskinere denne modellen i en operasjon med en 3-aksemaskin. Med en 5-aksemaskin kan arbeidsstykket roteres relativt til verktøyet i alle retninger. Når ønsket orientering er oppnådd kan hullene og flatene maskineres ved å holde rotasjonsaksene i ro, og bevege kun X-,Y-og Z-

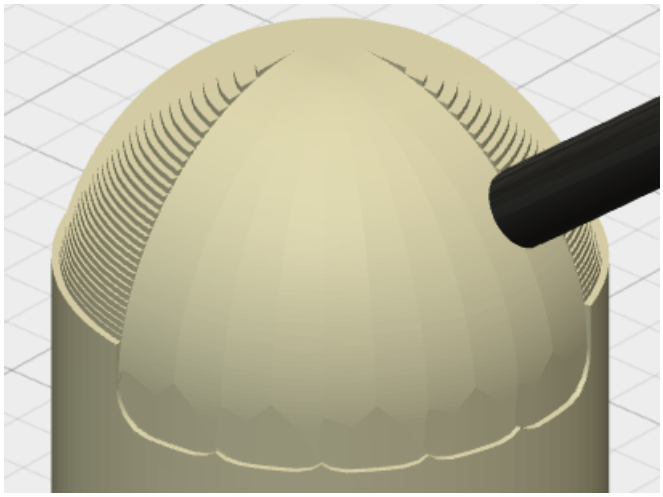


(a) 3-akse med radiefres(1). (b) 5-akse posisjonering(5).

Figur 2.2

aksene(5).

Figur 2.3 viser en modell som krever fem akser for å maskinere overflaten. Dersom orienteringen av verktøyet relativt til arbeidsstykket skal endres for hvert steg, må verktøymaskinen kontrollere alle fem akser simultant for å avvirke materiale (5).



Figur 2.3: 5-akse kontur.

2.2 CAD/CAM

CNC-maskiner styres av data generert fra en programvare kjent som CAM(Computer-Aided Manufacturing). CAM er et verktøy for å automatisk generere verktøybaner fra en CAD(Computer-Aided Design)-modell, som igjen kan genereres til maskinspråk for både dreiing og fresing. Det finnes mange varianter av programvare for CAD og CAM, både frittstående CAM eller CAD programvare og programvare som har CAM integrert i CAD. Det som tilsynelatende er gjennomgående i industrien er at programvarene har dyre lisenser og lav brukervennlighet som krever stor ekspertise å håndtere.

2.2.1 Autodesk Fusion 360

I 2013 lanserte Autodesk Fusion 360 den første skybaserte CAD plattformen, som siden har fått integrert CAM. Fusion 360 er tilgjengelig på plattformene PC, Mac og mobil og har stor fokus på brukervennlighet. Ettersom det er skybasert er det enkelt å samarbeide om prosjekter, der man kan endre, dele ideer og gi tilbakemelding på prosjekter over skyen fra ulike plattformer. Brukervennligheten til Fusion 360 har gjort det til den foretrukne CAD/CAM programvaren blant de ansatte på verkstedet til NTNU Valgrinda. Selv med tilgang til programvarer med mer avansert funksjonalitet og dyre liseser, veier brukervennligheten til Fusion 360 opp.

I CAM seksjonen i Fusion 360 utarbeides et program på grunnlag av en CAD-modell. Først defineres et råemne, deretter defineres koordinatene for nullpunktet til råemnet. Videre blir funksjoner som valg av verktøy og mating-og spindelhastighet lagt inn. Brukeren må så definere bevegelsen til verktøyet og dersom det er maskinering med rotasjonsakser kan orienteringen til verktøyet spesifiseres. Når alt er definert genererer CAM verktøybaner for å avvirke materiale fra råemne til ferdig maskinert komponent. Disse verktøybanene genereres automatisk basert på innstillingene definert av brukeren, og det kan derfor oppstå uønskede kollisjoner mellom verktøyet og arbeidsstykket. For å forhindre kollisjoner kan prosessen simuleres med et simuleringsverktøy innebygd i CAM-programvaren. Simuleringen viser verktøybanen relativt til arbeidsstykket for hele operasjonen fra råemne til ferdig maskinert komponent. Når simuleringen er ferdig kan den maskinerte virtuelle komponenten kontrolleres for eventuelle feil.

2.2.2 CL-data

Når simuleringen stemmer overens med ønsket modell kan verktøybanene lagres som CL(Cutter Location)-data. I en CL-fil beskrives verktøybanene av senterpunktet til verktøyet med posisjoner og orienteringer relativt til nullpunktet definert i CAM-programmet. Koordinatene til nullpunktet er gitt av [x, y, z, i, j, k] der i,j,k er verktøyets orienteringsvektor og x,y,z er koordinatene til verktøyets senterpunkt. CL-data er uavhengig av verktøymaskiner fordi verktøybanene beskrives

relativt til maskinens nullpunkt, og tar ikke hensyn til konfigurering av maskinakser(6).

2.2.3 NC-data

Der CL data beskriver lokasjonen av verktøyet relativt til arbeidsstykkets definerte nullpunkt, er verktøyets posisjon og orientering beskrevet av maskinkoordinater fra nullpunktet i en verktøymaskin. En verktøymaskin med flere akser vil følge kodede instrukser gitt av NC-data(Numerical Control). NC-data er gitt av punktkoordinater og rotasjonsakser om maskinens koordinatsystem [X, Y, Z, A, B, C] der [X, Y, Z] er koordinater til maskinens tre lineære akser, også kjent som sleider, og [A, B, C] er koordinater til rotasjonsaksene. NC-data gir instrukser i form av tall og bokstaver, linje for linje, der en linje blir referert til som en blokk. Styringssystemet til en CNC-maskin tolker blokkene og konverterer det til styresignaler. Eksempler på styringssignaler kan være å rotere spindelen verktøyet er festet i og/eller bevege verktøyet i en bestemt hastighet langs de ulike aksene(7).

2.2.4 Programmeringsformat

Programmeringsformatet beskriver regler for at styringen til en verktøymaskin skal anvende programmert informasjon. Programmeringsformatet er regulert av internasjonal standard (ISO 6983/1 og 6983/2). I en NC-styrt maskin blir et program skrevet med nødvendig antall blokker ut ifra hva som skal utføres, der hver blokk inneholder informasjonen som er nødvendig for å utføre en del av programmet(2). G, M, F, S og T-koder er vanlige adresser i maskinkoden. Koder med G adressen brukes for å kontrollere sleidene og CNC systemet. G-koder kan deles inn i fire kategorier:

- Posisjon og orientering til verktøyet relativt til arbeidsstykket
- Sette et lokalt koordinatsystem
- Interpolasjon for å maskinere konturer som sirkel og linje
- Diverse ting som å kompansere for lengden av verktøyet.

M-adresser er kommandoer for skru av og på funksjoner i maskinen, som for eksempel å starte et program, slå av og på kjøleveske, rotere spindelen med eller mot klokken, klargjøre for verktøybytte osv.
F-adressen benyttes for å sette matingshastigheten som vil si hastigheten verktøyet og arbeidsstykket, og programmeres i mm/rev eller mm/min.
S-adressen benyttes for å sette spindelhastigheten i RPM.

T-adressen benyttes for verktøybytte, der maskinen bytter til verktøyet som er på plassen spesifisert i nummeret etter bokstaven *T*. T05 vil for eksempel hente verktøyet som er på plass nummer 5 i maskinen(3).

2.2.5 Postprosessor

Ethvert CAM-program trenger en postprosessor for å generere verktøybaner. De første CAM systemene genererte G-og M-koder i form av linje-og banebevegelser, men behovet for en postprosessor med et eget grensesnitt ble større da G-og M-koder ble mer avanserte med komplekse funksjoner som automatisk verktøybytte, og radius-og verktøykompensering. Selv om programformatet til en styring blir regulert av ISO-standardene, blir ikke alltid ISO-standardene fulgt av alle CNC-maskinene(2), dermed kan betydningen av G-og M-koder variere for forskjellige styringer, og det må være mulig å gjøre tilpasninger i grensesnittet til postprosessorene slik at en postprosessor kan tilpasses styringen til en CNC-maskin.

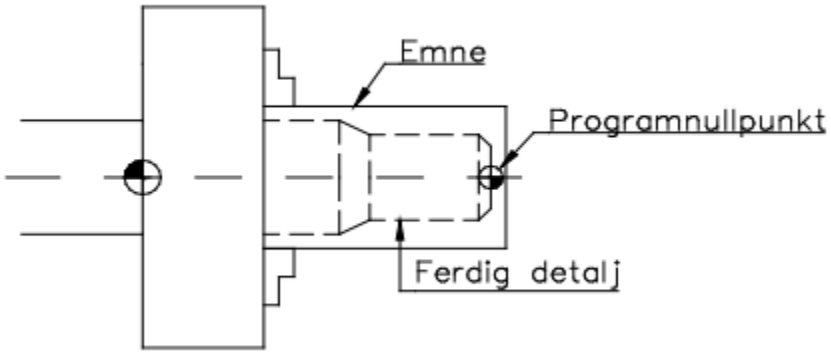
CAM programmet genererer en midlertidig fil med instrukser som er skjult for brukeren, denne filen er uten et spesielt filformat, og inneholder data med informasjon som posisjonering av verktøyet relativt til arbeidsstykket, verktøylengde og hastigheter. En postprosessor tolker denne datafilen og genererer et program med blokker av maskinspesifikke NC-data(8), slik figur 2.4 illustrerer.

Figur 2.4: Flytskjema fra CAM til NC-data

Verktøybaner definert i et CAM-program er gitt av CL-data, for å konvertere verktøybaner gitt av CL-data til maskinlesbar NC-data må det gjennom en postprosessor. I CL-data finnes det informasjon om freseverktøyets lengde og bevegelser for å kutte materiale, men inneholder ikke informasjon om vinkler på rotasjonsaksene. En postprosessor vil bruke CL-dataen til å regne ut lokasjonen til sleidene og rotasjonen om rotasjonsaksene. For at verktøyets orientering og posisjon relativt til arbeidsstykket skal være lik i CAM og i maskinstyringen må oppspenningen av arbeidsstykket være lik i CNC-maskinen og i CAM og for at oppspenningen skal bli lik må et nullpunkt defineres(9).

Når et arbeidsstykke skal bearbeides må CNC-styringen vite hvor emnet befinner seg. For å vite orienteringen til verktøyet relativt til arbeidsstykket brukes det tre forhåndsdefinerte faste punkt; maskinnullpunkt, referansepunkt og programnullpunkt:(2)

- Maskinnullpunktet er et fast punkt, fastlagt av maskinfabrikanten, som angir nullpunktet for CNC-maskinen sitt koordinatsystem. Alle andre referansepunkt og nullpunkt bruker maskinnullpunktet som utgangspunkt.
- Referansepunktet er fastlagt av maskinfabrikanten, med en bestemt avstand til maskinnullpunktet, og er til for å oppdatere styringens målesystem i forhold til maskinnullpunktets koordinatsystem. Når maskinens sleider(akser) styres til referansepunktet stopper maskinen og målesystemet oppdateres.
- Programnullpunktet(figur 2.5) er nullpunktet til arbeidsstykket, som plasseres av programmereren der det er mest hensiktsmessig i forhold til programmering og oppspenning. For å gjøre programmeringen enklere flyttes maskinnullpunktet til programnullpunktet.



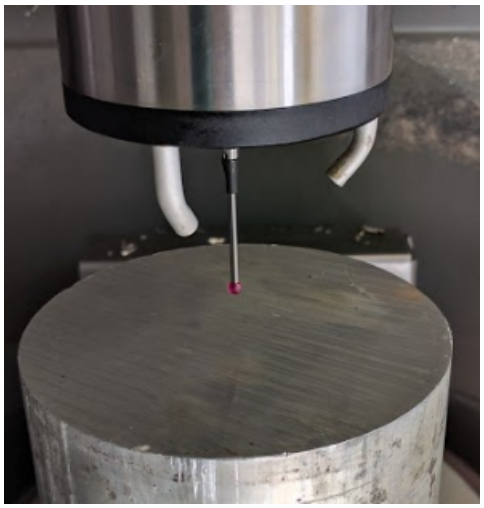
Figur 2.5: Plasseringen av programnullpunktet på et oppspent emne, plasseringen bestemmes ut fra emnets utseende, kvalitetskrav og oppspenning(2)

.

I en 3-aksemaskin kan programnullpunktet defineres av operatøren på et oppspent arbeidsstykke, for så å implementere nullpunktet på samme sted på modellen i CAM, og postprosessoren vil kalkulere verktøybanene om programnullpunktet.

2.2.6 Nullpunkt

For maskiner med flere akser vil rotasjonsaksene gjøre at nullpunktet vil være definert av maskinens kinematikk på et bestemt sted. Operatøren spenner fast arbeidsstykket i en skrustikke montert på rotasjonsbordet i verktøymaskinen for så å bruke en måleprobe(se figur 2.6) til å

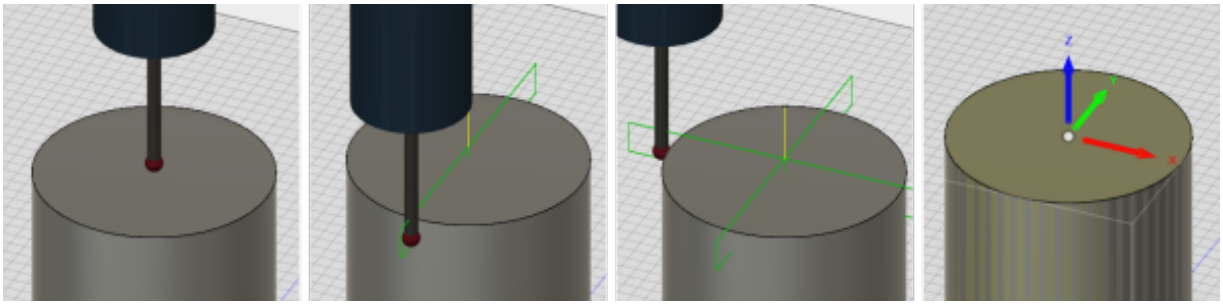


Figur 2.6: Måleproben montert i DMU 50 eVolution. En måleprobe er en elektronsik kantsøker som måler verdiene for plasseringen av emnet i forhold til maskinnullpunktet.

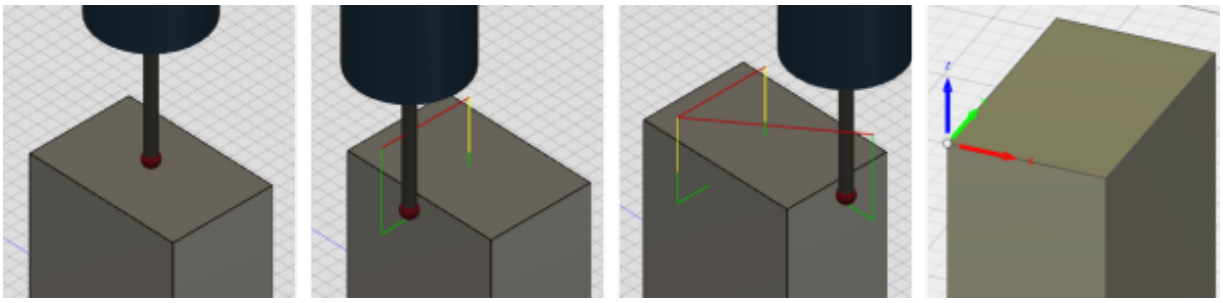
måle posisjonen og orientering til arbeidsstykket relativt til maskinens nullpunkt. Resultatet av målingene gir ut distansen i millimeter mellom sleidene fra det oppmålte punktet til maskinnullpunktet i form av X, Y og Z-verdier, slik at det kan defineres et programnullpunkt på arbeidstykket. Programnullpunktet er da gitt med posisjon og orientering relativt til maskinnullpunktet. For at translasjonen fra maskinnullpunktet til programnullpunktet i verktøymaskinen skal samsvare med programnullpunktet definert i oppsetet til emne i CAM, må verdien av målingene implementeres i CAM, ofte via brukerinnstillingene til postprosessoren.

For at koordinatsystemene til de to nullpunktene plasseres samme sted på arbeidsstykket, er det hensiktsmessig å plassere programnullpunktet for en sylinder i toppsentrum av delen slik figur 2.7 illustrerer, og for en kube i et hjørne slik figur 2.8 illustrerer.

Grunnet rotasjonsaksene til fem-aksemaskiner vil kinematikken til maskinen gjøre post-



Figur 2.7: Simulering av oppmåling av koordinatsystemet til en sylinder. En måleprobe måler positiv og negativ X og Y verdi, samt høyden til sylinderen på Z-aksen. Dersom verktøymaskinen har et probeprogram for å måle en sylinder installert, vil resultatet av målingen være avstanden fra nullpunktet definert i rotasjonsbordet og opp til det oppmålte koordinatsystemet, gitt i X-, Y-og Z verdier. X-, Y-og Z-verdiene implementeres så inn i postprosessoren som offsetverdier.



Figur 2.8: Simulering av en oppmåling av koordinatsystemet til en kube. En måleprobe måler X og Y verdien fra kantene, samt høyden til kuben Z aksen, og det dannes et koordinatsystem i toppen av hjørnet. Dersom verktøymaskinen har et probeprogram for å måle en kube installert, vil resultatet av målingen være avstanden fra nullpunktet definert i rotasjonsbordet og opp til det oppmålte koordinatsystemet, gitt i X-, Y-og Z-verdier. X-, Y-og Z-verdiene implementeres så inn i postprosessoren som offsetverdier.

prosessering komplisert. Transformasjonen fra arbeidsstykket til maskinkoordinater kalles invers kinematikk. I NC-maskinering med fem akser, må CL-data på formen [x,y,z,i,j,k] transformeres til samsvarende maskinkoordinater på formen [X, Y, Z, A, B] (eller A og C, eller B og C ettersom hvilke to akser arbeidsbordet i maskinen roterer om), som kontrollerer bevegelsene i maskinen. I de fleste operasjonene er bevegelsene en kombinasjon av bevegelse om arbeidsstykket og verktøyet. Algoritmene fra inverskinematikken til maskinen vil transformere programnullpunktet til maskinnullpunkt(5). Denne transformeringen blir som oftest regnet ut i postprosessoren som en del av utregningen fra CL-data til CNC-data. Fordi hver modell av en verktøymaskin har et unikt geometrisk oppsett, vil rotering og posisjonering av de forskjellige aksene relativt til arbeidsstykket være ulik fra maskin til maskin. Det må derfor utvikles en egen postprosessor, med maskinspesifikk inverskinematikk tilpasset en spesifikk verktøymaskin.

2.3 Kinematikk

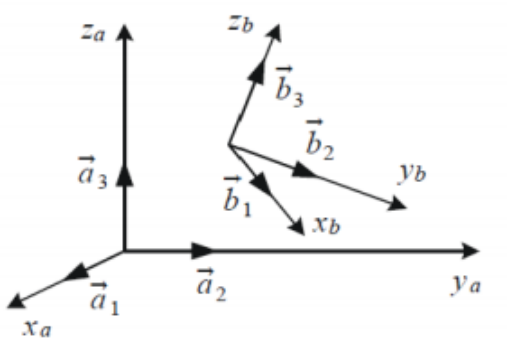
2.3.1 Posisjon og orientering

I verktøymaskiner kan posisjonen og orienteringen til et arbeidsstykke beskrives ut fra et kjent koordinatsystem (figur 2.9).

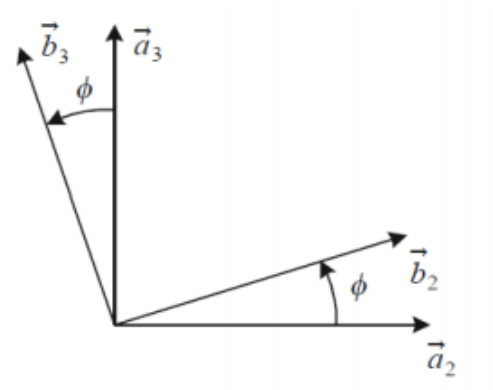
Rotasjonen fra a til b beskrives av likningen:

$$\mathbf{R}_b^a = \{\vec{a}_i \cdot \vec{b}_j\} \tag{2.1}$$

der $\mathbf{R}$ er rotasjonsmatrisen. For en rotasjon med en vinkel ϕ om x_a fra koordinatsystem a til b vil rotasjonsmatrisen beskrives som $\mathbf{R}_x(\phi)$. På samme måte vil da $\mathbf{R}_y(\theta)$ være rotasjonen om y-aksen med en vinkel ϕ, og $\mathbf{R}_z(\psi)$ er rotasjonen om z-aksen med en vinkel ψ.



Figur 2.9: Koordinatsystmene a og b (10).



Figur 2.10: En rotasjon om $\vec{a}_1$ med en vinkel ϕ (10).

For rotasjonen $\mathbf{R}_x(\phi)$ kan det fra figur 2.10 observeres at $\vec{a}_1 = \vec{b}_1$ slik at $\vec{a}_1 \cdot \vec{b}_1 = 1$, videre blir da

$$\vec{a}_1 \cdot \vec{b}_2 = \vec{a}_1 \cdot \vec{b}_3 = \vec{a}_2 \cdot \vec{b}_1 = \vec{a}_3 \cdot \vec{b}_1 = 0 \tag{2.2}$$

$$\vec{a}_1 \cdot \vec{b}_1 = cos(\phi), \quad \vec{a}_3 \cdot \vec{b}_3 = cos(\phi) \tag{2.3}$$

$$\vec{a}_3 \cdot \vec{b}_2 = sin(\phi), \quad \vec{a}_1 \cdot \vec{b}_1 = -sin(\phi) \tag{2.4}$$

Matrisene for $\mathbf{R}_y(\theta)$ og $\mathbf{R}_z(\psi)$ finnes på samme måte og de tre rotasjonsmatrisene blir dermed:

$$\mathbf{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{2.5}$$

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.6}$$

$$\mathbf{R}_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

Posisjonen og orienteringen fra koordinatsystemet b til a beskrives av en 4x4 homogen transformasjonsmatrise T på formen:

$$\mathbf{T}_b^a = \begin{bmatrix} \mathbf{R}_b^a & \mathbf{r}_{ab}^a \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.8}$$

$\mathbf{r}_{ab}^a$ representerer translasjonen fra koordinatsystem b til a, der $\mathbf{r}_{ab}^a$ beskriver posisjonen av origo til koordinatsystem b relativt til origo av koordinatsystem a, gitt i a koordinater(10). For en kjede med flere ledd kan transformasjonsmatrisen skrives som $\mathbf{T}_t^w = \mathbf{T}_1^w\mathbf{T}_2^1\mathbf{T}_3^2...\mathbf{T}_n^{n-1}\mathbf{T}_t^n$ der hvert ledd i kjeden er transponert og/eller rotert med referanse til koordinatsystemet for det forrige leddet i kjeden. $\mathbf{T}_t^w$ beskriver da verktøyets posisjon relativt til arbeidsstykket(6).

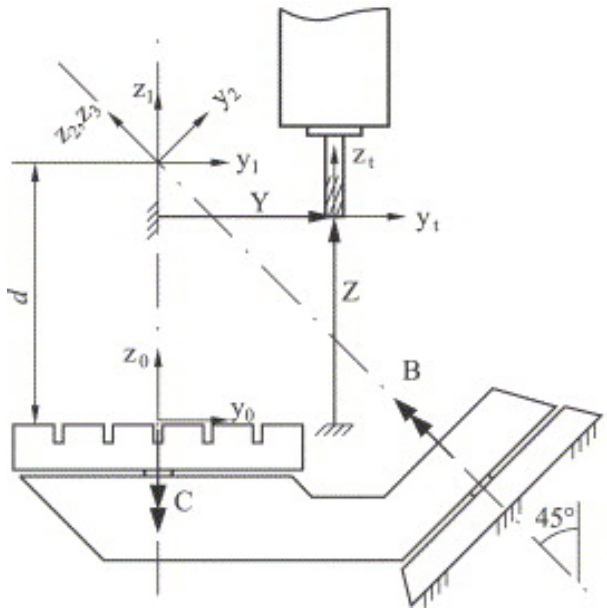
2.3.2 Deckel Maho DMU 50 eVolution

Verkstedet på NTNU Valgrinda har en Deckel Maho DMU 50 eVolution, som er en CNC-maskin med fem akser. Figur 2.11 illustrerer Deckel Maho DMU 50 eVolution tegnet fra siden med et rotasjonsbord som roterer om aksene B og C. Vinkelen mellom aksene er 45°, der krysningspunktet mellom aksene er i et punkt med en vertikal avstand d fra maskinkoordinatsystemet $x_0y_0z_0$. Arbeidsområde til B-aksen er [0, 180°] og C-aksen er $[-\infty,\infty]$. Rotasjonsaksene gjør at et verktøy kan posisjoneres vinkelrett på et tilfeldig punkt på overflaten til en halvkule oppspent på rotasjonsbordet (11).

2.3.3 Direkte kinematikk

Direkte kinematikk brukes for å kalkulere CL-data fra maskinaksevariablene gitt i NC-data. Direkte kinematikk kan brukes til å tette avviket som oppstår fra ønsket verktøybane og faktiske verktøybane i maskiner med rotasjonsakser ved hjelp av en lineariseringsalgoritme som blir forklart i detalj senere i kapittelet 2.6. Knut Sørby presentere i sin forskning (11) likninger for direkte kinematikk til Deckel Maho DMU 50 eVolution. Følgende koordinatssytem ble definert for å bestemme likningene for kinematikken til maskinen:

- $x_0y_0z_0$: Basekoordinatsystemet, eller nullpunktet er lokalisert i senter av rotasjonsbordet når vinklene B og C er 0°
- $x_1y_1z_1$: En translasjon av $x_0y_0z_0$ i en distanse d om z_0



Figur 2.11: DMU 50 eVo med B aksen i 0° sett fra siden (11).

- $x_2y_2z_2$: Rotasjon om $x_1y_1z_1$ i en vinkel på +45°om x_1. Koordinatsystemene $x_0y_0z_0$, $x_1y_1z_1$ og $x_2y_2z_2$ er faste og beveger seg ikke med maskinaksene.
- $x_3y_3z_3$: Rotasjon om $x_2y_2z_2$ i en vinkel B om z_2.
- $x_4y_4z_4$: Rotasjon om $x_3y_3z_3$ i en vinkel -45°om x_3
- $x_5y_5z_5$: Translasjon av $x_4y_4z_4$ i en distanse $-d$ langs z_4. Koordinatsystemet $x_5y_5z_5$ er alltid lokalisert i senteret til maskin bordet, selv etter B aksen er rotert.
- $x_wy_wz_w$: Arbeidstykket sitt koordinatsystem, finnes ved å rotere $x_5y_5z_5$ i en vinkel -C om z_5.
- $x_ty_tz_t$: Et fast koordinatsystem i senteret til verktøytuppen.

2.3.4 Denavit-Hartenberg parametre

Denavit Hartenberg bruker fire parametere for å beskrive posisjonen til hvert koordinatsystem relativt til det forrige koordinatsystemet(12). Ved å bruke tabell 2.1 og de definerte koordinatsystemene til Deckel Maho DMU 50 eVolution gitt i avsnitt 2.3.3, kan Denavit-Hartenberg parameterne fra maskinnullpunktet og opp til arbeidsstykket sitt koordinatsystem og utredes, og er gitt i tabell 2.2.

Tabell 2.1: Finne Denavit-Hartenberg parametere.

Joint Angle	θ_i	Vinkelen mellom x_{i-1} og x_i aksen om z_{j-1} aksen
Link offset	d_j	Distansen fra basen til koordinatsystemet $i-1$ til x_i aksen langs z_{i-1} aksen
Link length	a_i	Distansen mellom z_{i-1} aksen og z_i langs x_i aksen
Link twist	α_i	vinkelen fra z_{i-1} aksen til z_i aksen om x_i

Tabell 2.2: Denavit-Hartenberg parametere - DMU 50 eVolution

Link	a_i	α_i	d_i	θ_i
1	0	0	d	0
2	0	$\frac{\pi}{2}$	0	0
3	0	0	0	B
4	0	$-\frac{\pi}{2}$	0	0
5	0	0	$-d$	0
6	0	0	0	C

2.3.5 Finne transformasjonsmatrisen fra Denavit-Hartenberg parametere

Den homogene transformasjonsmatrisen fra koordinatsystem $i-1$ til i, er definert i rotasjoner og transformasjoner som

$$^{i-1}A_i(\theta_i, d_i, a_i, \alpha_i) = Rot_{z,\theta_i} \cdot Trans_{z,d_i} \cdot Trans_{z,d_i} \cdot Rot_{x,\alpha_i} \tag{2.9}$$

der:

$$Rot_{z,\theta_i} = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) & 0 & 0 \\ sin(\theta_i) & cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

$$Trans_{z,d_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

$$Trans_{z,d_i} = \begin{bmatrix} 1 & 0 & 0 & \alpha_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

$$Rot_{x,\alpha_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_i) & -sin(\alpha_i) & 0 \\ 0 & sin(\alpha_i) & cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

Transformasjonsmatrisen mellom to lokale koordinatsystem blir da:

$$\boldsymbol{A_i^{i-1}}(\theta_i, d_i, a_i, \alpha_i) = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i)cos(\alpha_i) & sin(\theta_i)sin(\alpha_i) & a_i cos(\theta_i) \\ sin(\theta_i) & cos(\theta_i)cos(\alpha_i) & -cos(\theta_i)sin(\alpha_i) & a_i sin(\theta_i) \\ 0 & sin(\alpha_i) & cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

Transformasjonen fra koordinatsystemet i senteret av verktøyenden til arbeidsstykket sitt koordinatsystem kan skrives:

$$\boldsymbol{T_w^t} = \boldsymbol{T_w^5 T_5^4 T_4^3 T_3^2 T_2^1 T_1^0 T_0^t} \tag{2.15}$$

Ved å bruke transformasjonsmatrisen 2.14 og tabell 2.2 blir verdiene til $\boldsymbol{T}$ for hvert ledd:

$$\boldsymbol{T_0^1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.16}$$

$$\boldsymbol{T_1^2} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\frac{\pi}{2}) & -sin(\frac{\pi}{2}) & 0 \\ 0 & sin(\frac{\pi}{2}) & cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.17}$$

$$\boldsymbol{T_2^3} = \begin{pmatrix} cos_B & -sin_B & 0 & 0 \\ sin_B & cos_B & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.18}$$

$$\boldsymbol{T_3^4} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\frac{\pi}{2}) & -sin(\frac{\pi}{2}) & 0 \\ 0 & sin(\frac{\pi}{2}) & cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.19}$$

$$\boldsymbol{T_4^5} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -d \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.20}$$

$$\boldsymbol{T_5^w} = \begin{pmatrix} -cos_C & sin_C & 0 & 0 \\ -sin_C & -cos_C & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.21}$$

$$\boldsymbol{T_0^t} = \begin{pmatrix} 1 & 0 & 0 & X \\ 0 & 1 & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.22}$$

$\boldsymbol{T_w^0}$ er gitt fra Denavit Hartenberg parametrene og er transformasjonen fra maskinnullpunktet til arbeidstykket sitt koordinatsystem. $\boldsymbol{T_0^t}$ er koordinatsystemet i senteret til verktøytuppen, slik at $\boldsymbol{T_w^t}$ er en funkson av aksevariablene til maskinen; X, Y, Z, B og C. Orienteringen av Z-aksen fra verktøykoordinatsystemet relativt til arbeidsstykket sitt koordinatsystem er i den tredje kolonnen av $\boldsymbol{T_w^t}$. Posisjonen til enden av verktøyet relativt til arbeidsstykket er i den fjerde kolonnen. CL-data fra aksevariabler kan derfor finnes med likingen:

$$\begin{bmatrix} i & x \\ j & y \\ k & z \\ 0 & 1 \end{bmatrix} = \boldsymbol{T_w^t} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.23}$$

Ved å bruke likning 2.23 blir direkte kinematikken for DMU 50 eVolution (11)

$$i = \frac{1}{2}(\sqrt{2}cos_C sin_B - sin_C cos_B + sin_C) \tag{2.24}$$

$$j = \frac{1}{2}(\sqrt{2}sin_C sin_B + cos_C cos_B - cos_C) \tag{2.25}$$

$$k = \frac{1}{2} + \frac{1}{2}cos_B \tag{2.26}$$

$$x = \frac{\sqrt{2}}{2}[Xsin_C + (Y+Z-d)cos_C]sin_B + \frac{1}{2}[-Y+Z-d-(Y+Z-d)cos_B]sin_C + Xcos_Ccos_B \tag{2.27}$$

$$y = \frac{\sqrt{2}}{2}[(Y+Z-d)sin_C - Xcos_C]sin_B + Xsin_Ccos_B + \frac{1}{2}[(Y+Z-d)cos_Ccos_B + (Y-Z+d)cos_C] \tag{2.28}$$

$$z = \frac{1}{2}[-\sqrt{2}Xsin_B + (Y+Z-d)cos_B - Y + Z + d] \tag{2.29}$$

2.3.6 Inverskinematikk

I denne oppgaven har inverskinematikken til DMU 50 eVolution blitt implementert som en del av postprosessoren for å generere maskinkode med verktøyets posisjon og orientering gitt av maskinaksenes posisjon og vinkler relativt til maskinens nullpunkt. Ved å bruke 2.15 og relasjonen

$$(\boldsymbol{T}_{\boldsymbol{w}}^{\boldsymbol{t}})^{-1} = \boldsymbol{T}_{\boldsymbol{t}}^{\boldsymbol{w}} = \boldsymbol{T}_{\boldsymbol{t}}^{\boldsymbol{0}}\boldsymbol{T}_{\boldsymbol{0}}^{\boldsymbol{1}}\boldsymbol{T}_{\boldsymbol{1}}^{\boldsymbol{2}}\boldsymbol{T}_{\boldsymbol{2}}^{\boldsymbol{3}}\boldsymbol{T}_{\boldsymbol{3}}^{\boldsymbol{4}}\boldsymbol{T}_{\boldsymbol{4}}^{\boldsymbol{5}}\boldsymbol{T}_{\boldsymbol{5}}^{\boldsymbol{w}} \tag{2.30}$$

kan man utarbeide et sett med likninger for å finne en analytisk løsning til inverskinematikken:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \\ a_{41} & a_{42} \end{bmatrix} = \boldsymbol{T}_{\boldsymbol{t}}^{\boldsymbol{w}} \begin{bmatrix} i & x \\ j & y \\ k & z \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.31}$$

Ved å bruke 2.31 presenterte Sørby likningene 2.32-2.36 for å finne vinklene B og C samt posisjonene X, Y og Z(11).

$$B = arccos(2k-1),\ for\ 0° \leqslant B \leqslant 180° \tag{2.32}$$

$$C = arctan[(1-k)i + \sqrt{2(k-k^2)}j, \sqrt{2(k-k^2)}i + (k-1)j],\ for\ -180° < C \leqslant 180° \tag{2.33}$$

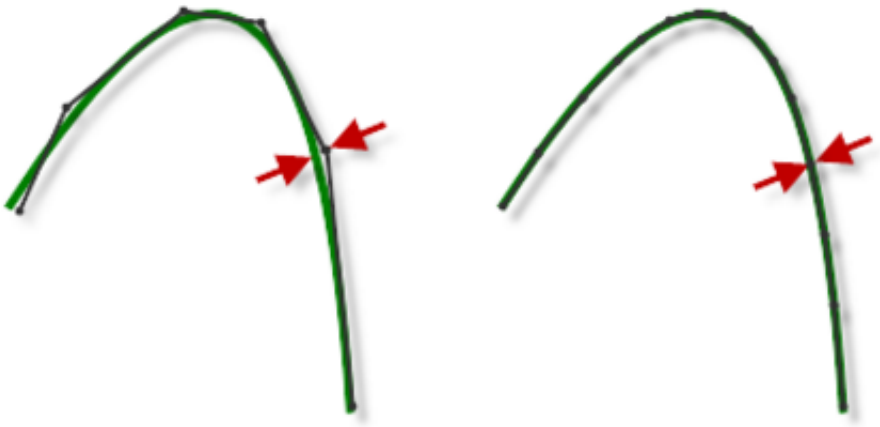
$$X = [-y\sqrt{2(k-k^2)} - x + 2xk]cos_C + [x\sqrt{2(k-k^2)} + 2yk - y]sin_C + (d-z)\sqrt{2(k-k^2)} \tag{2.34}$$

$$Y = [x\sqrt{2(k-k^2)} + yk]cos_C + [y\sqrt{2(k-k^2)} - xk]sin_C - z + d - dk + zk \tag{2.35}$$

$$Z = [x\sqrt{2(k-k^2)} + yk - y]cos_C + [y\sqrt{2(k-k^2)} - xk + x]sin_C + d - dk + zk \tag{2.36}$$

2.4 Toleranser

Verktøybaner for å kjøre i konturmodus med G-kode G1, G2 eller G3 genereres fra CAM ved å tilnærme hvert NC punkt med små linjer som tilsammen utgjør verktøybanen. Linjene genereres ved linearisering basert på toleranseverdien definert i CAM. Hvor nøyaktig verktøybanen blir avhenger av antall linjer som blir brukt. Som figur 2.12 illustrerer vil flere linjer generere



Figur 2.12: På bildet til venstre er toleransen satt til 0.1mm og i bildet til høyre er den 0.001mm, det kan observeres at verktøybanen blir betydelig bedre med en lavere toleranse(13).

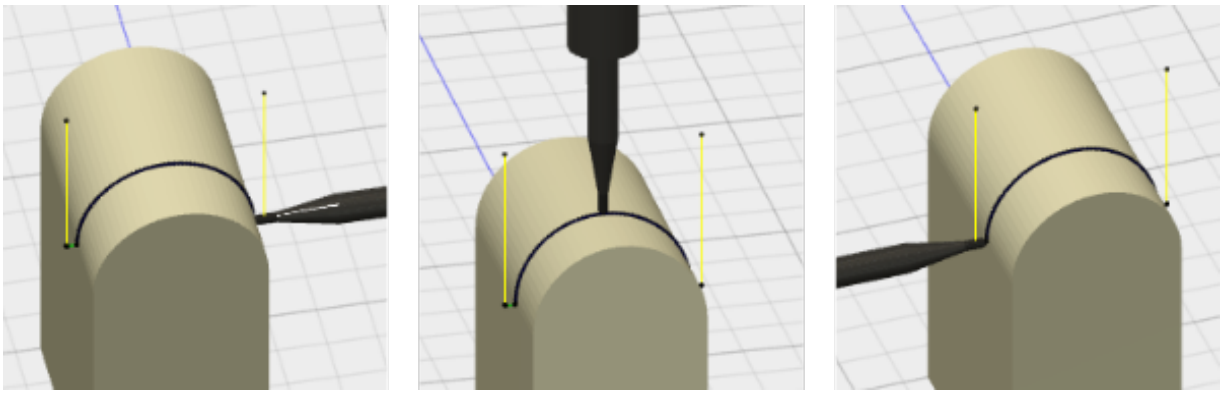
en verktøybane som ligger tettere opp mot den ønskede banen. For å unngå avvik kan toleranseverdien settes lav slik at det blir generert flere NC-punkt. Det vil derimot ta lenger tid å kalkulere verktøybaner, som kan være et problem dersom man har lange baner. Et større problem med lav toleranseverdi er at det blir veldig små bevegelser på hver verktøylinje, og med høy matingshastighet kan det oppstå *data starvation*.

Data starvation oppstår når styringssystemet til verktøymaskinen skal prosessere så mye data at det blir hengende etter. Et system kan bare prosessere et bestemt antall blokker/linjer med kode per sekund. På eldre maskiner kan det være så lite som 40 blokker per sekund, mens på nyere maskiner kan det prosesseres rundt 1000 blokker per sekund. Når styringssystemet ikke klarer å prosessere data i hastighet med linjer per sekund i programmet vil maskinen ta pauser etter hver bevegelse og vente på neste kommando. Fresingen blir derfor hakkete og overflaten blir ujevn. (13)

2.5 Singularitetskontroll

Når en av rotasjonsaksene er parallelle med verktøyaksen oppstår det en singularitet, som kan være et problem i 5-akse maskinering, ettersom enhver orientering av rotasjonsaksen vil gi samme verktøyorientasjon relativ til arbeidsstykket. For Deckel Maho DMU 50 eVolution oppstår en singularitet når vinkel B=0°, da er rotasjonsbordet horisontalt og C-aksen står parallell på verktøyaksen. Når B=0° bør da C-aksen ha en posisjon som minimerer bevegelsen fra den forrige posisjonen til C-aksen. Problemet med singularitet er at C-aksen må gjøre en rask rotasjon på 180° for å komme til neste NC-datapunkt, slik grafen i figur 2.15a illustrerer.

Problemet oppstår for eksempel når det skal maskineres en halvsylinder som i figur 2.13.

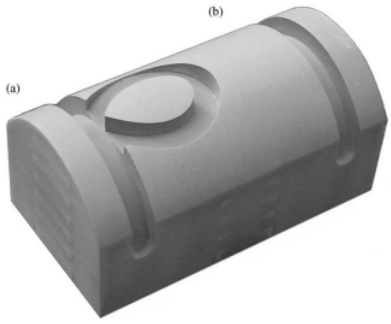


Figur 2.13: Det vil oppstå en singularitet når verktøyet er i posisjonen i bildet i midten. For å komme til neste NC-datapunkt må bordet rotere 180°, og uten singularitetskontroll vil verktøyet kollidere med arbeidsstykket.

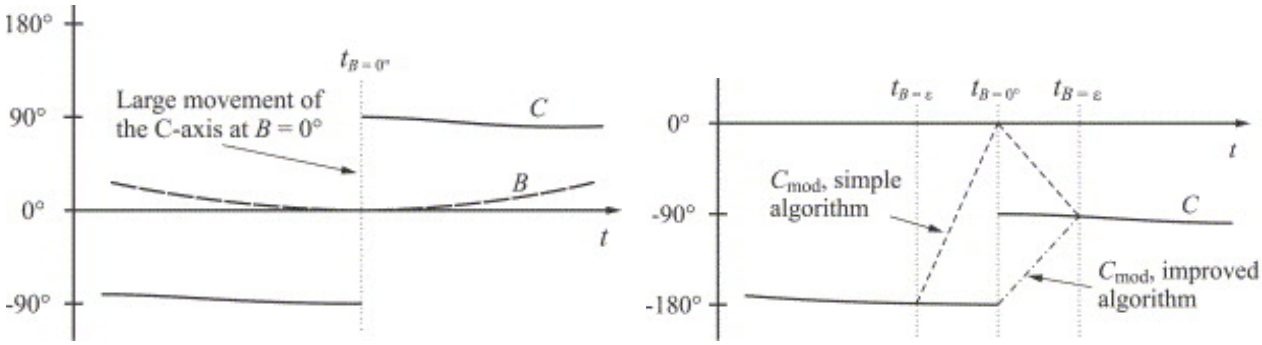
CNC-koden nedenfor viser verktøybanen til figur 2.13 postprosessert uten singularitetskontroll, det kan observeres at mellom NC-datapunktene N295 og N300 tar C-aksen en rask 180° rotasjon uten at det blir tatt høyde for i X, Y, Z og B. Verktøyet vil dermed kollidere med arbeidsstykket, som illustrert i figur 2.14 resulterer i en uønsket geometri. Vedlegg F.1 viser en tilsvarende CNC-kode postprosessert med singularitetskontroll.

```
1 N285 X-18.14 Y-11.379 Z+298.6 B+12.027 C+184.26
2 N290 X-12.393 Y-10.638 Z+299.354 B+8.162 C+182.888
3 N295 X-6.083 Y-10.153 Z+299.844 B+4.006 C+181.417
4 N300 X-0.275 Y+10. Z+300. B+0.148 C+0.052
5 N305 X-7.125 Y+9.814 Z+299.82 B+4.302 C+1.521
6 N310 X-13.4 Y+9.337 Z+299.354 B+8.162 C+2.888
```

For å håndtere en singularitet presenterte Sørby likning 2.37. Når parameteren ε har en lav



Figur 2.14: Bilde illustrerer til venstre ved (a) løkken verktøyet vil lage i arbeidsstykket når det oppstår en singularitet uten singularitetskontroll i postprosessoren. Til høyre ved (b) illustreres verktøybanen med singularitetskontroll i postprosessoren(11).



(a) Når B=0° oppstår det en singularitet og det kan bli en verktøykræsj og/eller grov feil i overflaten(11).

(b) C_{mod} med og uten variabelen C^* (11).

Figur 2.15

verdi, for eksempel 0.01° og vinkel B er mindre enn ε (og k=1) tvinger C_{mod} C-aksen til 0°.

$$if\ B < \varepsilon\ then\ C_{mod} = C\frac{B}{\varepsilon} \tag{2.37}$$

Når X, Y og Z kalkuleres med likningene 2.34-2.36 brukes verdien C_{mod} istedenfor C fra CL-data. Algoritmen kan i enkelte tilfeller føre til unødvendige bevegelser, derfor ble en forbedret utgave presentert i likning 2.38

$$if\ B < \varepsilon\ then\ C_{mod} = (C - C^*)\frac{B}{\varepsilon} + C^* \tag{2.38}$$

Variabelen C^* er posisjonen til C-aksen fra siste CL-data før singularitet-og lineariseringskontrollen begynte, og ble implementert for å unngå tilfeller illustrert i figur 2.15b, der C går innom 0° på banen fra -180° til -90°(11).

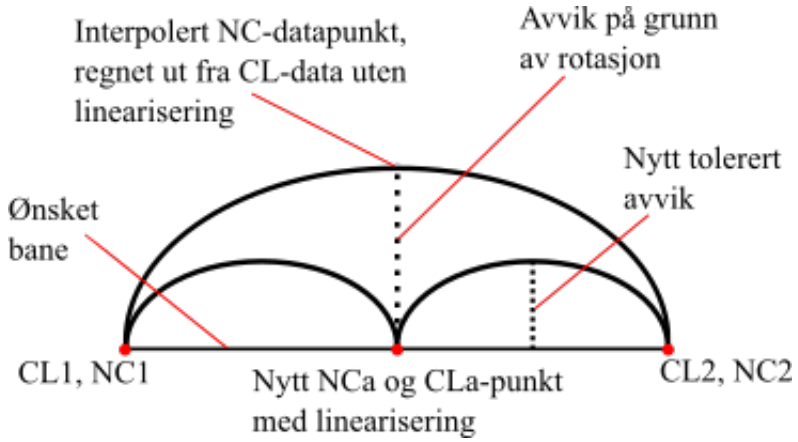
2.6 Linearisering

Når CL-data genereres vil banen mellom to CL-datapunkt være en rett linje, men på grunn av rotasjonsakser i maskinen vil det oppstå et avvik mellom den ønskede banen og den faktiske banen på arbeidsstykket gitt av NC-data.

Som figur 2.16 illustrerer vil CAM-systemet postprosessere bevegelsen mellom to CL-datapunkt som en rett linje. Når det er bevegelse mellom to CL-datapunkt i tre lineære akser vil bevegelsen være en rett linje både i CAM og i maskinen. Problemet oppstår når det er en simultan bevegelse i fem akser, der to roterende akser legges til de tre lineære. Startpunktet og sluttpunktet i verktøybanen mellom to CL-datapunkt vil være likt i CAM og i maskinen, men rotasjonsaksene fører til at bevegelsen mellom de to CL-datapunktene i realiteten være bueformet

og CAM systemet regner ut bevegelsen som en rett linje. Det vil oppstå et avvik mellom CAM modellen og den ferdig maskinerte komponenten(14). For å begrense avviket kan verktøybanen lineariseres ved å interpolere et nytt CL-data punkt langs den ideelle verktøybanen.

En lineariseringsalgortime vil undersøke om rotasjon vil føre til et avvik mellom verktøybanen gitt av NC og CL-data, dersom avviket er for stort legges det inn et nytt midtpunkt i CL-data, CL_a, som interpoleres mellom CL1 og CL2. Algoritmene med likninger fra inverskinematikken regner så ut et nytt NC_a punkt i verktøybanen, som har et tolerert avvik .



Figur 2.16: Nytt NC datapunkt ved linearisering

Kapittel 3

Utvikling av postprosessorer for Autodesk Fusion 360

Dette kapittelet beskriver arbeidet som ble gjort for å undersøke funksjonaliteten til ulike CAM-systemer og eksisterende postprosessorer, og hvordan Autodesk Fusion 360 står frem som det beste valget tanke på innføring i undervisning ved MTP, NTNU. Videre forklares arbeidet som ble gjort for å utvikle postprosessorer til Fusion 360 CAM, tilpasset 3-og 5-aksemaskiner på verkstedet ved MTP, NTNU.

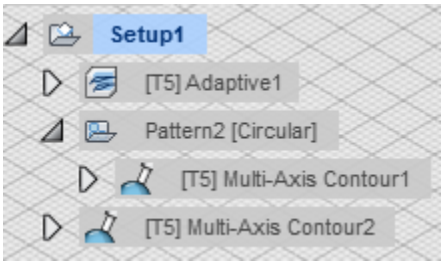
3.1 Oppbygging av en postprosessor til Autodesk Fusion 360

Autodesk har et postprossesorbibliotek(15) med ferdigprogrammerte postprosessorer til Fusion 360 CAM som passer til de mest brukte styringsspråkene og maskinene på markedet. Postprosessorene har filendelsen *.cps*, som er basert på JavaScript. Som forklart i kapittel 2.2.5 har CAM en midlertidig fil med instrukser. Postprosessorene til Fusion 360 er skrevet i seksjoner med rekkefølge som samsvarer med den midlertidige filen. Postprosessoren leser den midlertidige CAM-filen fra begynnelsen til slutt før filen slettes, og neste midlertidige fil med informasjon fra CAM prosesseres. Prosessen gjentas frem til CAM-programmet er slutt.

Seksjonene i en postprosessor er kodet med funksjoner i rekkefølgen onOpen, onSection og onMovement.

- onOpen starter NC-programmet med funksjoner som programnavn, kjøre verktøyet til trygg høyde før arbeidsbordet eventuelt roteres til 0°og kommentarer.
- onSection prosesserer alle operasjonene definert i *Setup*(figur 3.1) til modellen i CAM programmet. I Setup finnes informasjon om verktøyet, koordinatsystemet, kjøleveske, spindelhastighet og trygg verktøyhøyde for å unngå kollisjon i neste operasjon.

- onMovement inneholder funksjoner som onRapid, onLinear() og onCircular() som gir instrukser for å flytte verktøyet og rotere arbeidsbordet samt sette matingshastighet.
- onSectionEnd kalles på slutten av siste operasjon i en seksjon og gir ut instrukser som å skru av kjøleveske.
- onClose() brukes når hele CAM-programmet er over og genererer NC-kode for å avslutte programmet.



Figur 3.1: Setup-mappen med flere operasjoner i Fusion 360

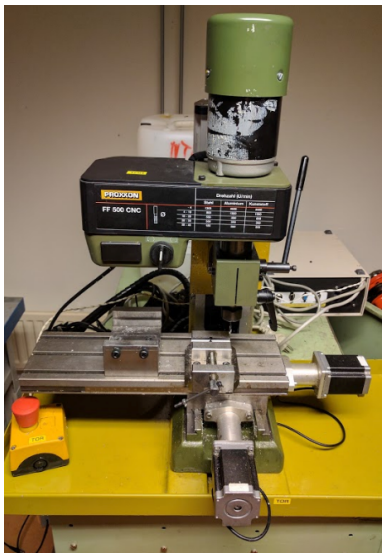
Ettersom en Autodesk Fusion 360 postprosessor er basert på JavaScript kan funksjonene i postprosessorene redigeres og/eller programmeres nye funksjoner, slik at det genereres NC-data som er tilpasset hver enkel CNC-maskin og det medfølgende styringsspråket(8).

3.2 Postprosessor tilpasset styringsspråket Mach3

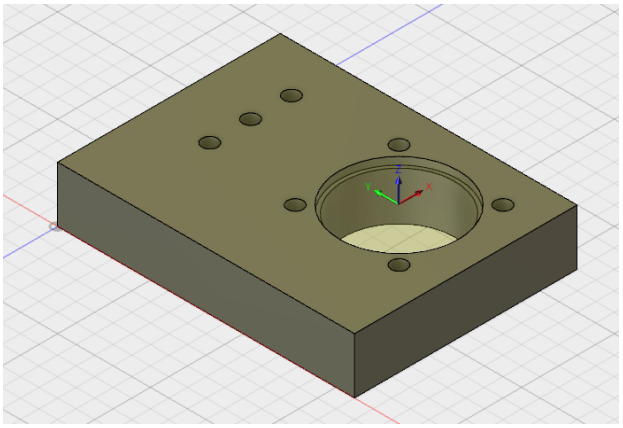
Professor Knut Sørby er emneansvarlig for TPK4190 - Produksjonsteknologi ved NTNU. Som en del av TPK4190 skal studentene produsere en Stirling motor. For å maskinere motorbasen til Stirlingmotoren følger studentene maskintegningen som finnes i vedlegg B og programmerer CNC-kode manuelt, ved å skrive G-og M-koder i tekstediter-programmet Notepad. Kodene skrevet manuelt av tidligere studenter finnes i vedlegg B.

Motorblokken til Stirlingmotoren maskineres med CNC-koden i en enkel 3-aksemaskin vist i figur 3.2a. Maskinen er en Proxxon-FF 500CNC, og styres kun av enkelte G og M-koder i de 3 lineære aksene X, Y og Z. Funksjoner som spindelhastighet kan kun settes til av og på i maks hastighet og det er ingen kjøleveske.

Proxxon-FF 500CNC styres av styringsprogrammet *Mach3*, som er et tilsynelatende populært program i industrien. Det er et rimelig program som gjør at en vanlig Windows datamaskin kan styre en CNC-maskin. På datamaskinen som var koblet til CNC-maskinen var det kun en demoversjon av Mach3 installert. Demoversjonen er gratis og har et begrenset antall linjer på 500 linjer med maskinkode. NTNU har en lisens for Mach3 og lisensfilen "Mach1Lic" ble funnet i Mach3 mappen på en annen datamaskin med lisensversjonen av Mach3 installert.



(a) 3-aksemaskinen som brukes i emnet TPK4190.



(b) CAD modell av motorblokken

Figur 3.2

Lisensfilen ble kopiert over til Mach3 mappen i demomaskinen koblet til Proxxon-FF 500 og det var da mulig å kjøre lisensversjonen av Mach3.

Det var ønskelig at studentene skulle gå fra å programmere CNC-koder manuelt, til å utarbeide et program i CAM på grunnlag av en CAD-modell. Som en del av denne masteroppgaven er det utviklet en manual for å modellere og maskinere ut motorblokken som brukes i Stirlingmotoren. Motorblokken, modelleres og verktøybanene defineres ved å bruke Autodesk Fusion 360 CAD og CAM. Manualen finnes i vedlegg C, og er tiltenkt å brukes som en del av undervisningsopplegget i TPK4190.

Manualen viser hvordan studentene kan bruke maskintegningen i vedlegg D til å modellere CAD-modellen illustrert i figur 3.2b, deretter programmere verktøybaner i CAM. Verktøybanene må så genereres til maskinkode for Mach3 styring på *Proxxon-FF 500CNC* ved å postprosessere i Fusion 360 CAM.

3.2.1 Utvikling av postprosessor

Ettersom Autodesk Fusion 360 er gratis for studenter og kan brukes på operativsystemene for PC, Mac og mobil, var det et naturlig valg å bruke som CAD/CAM-program i undervisningen.

For styringsspråket Mach3 har Autodesk utviklet en postprosessor som går under navnet *mach3mill.cps* og finnes i postprossesorbiblioteket til Autodesk(15).

En testmodell ble modellerert i CAD ut ifra maskintegningen i vedlegg B og det ble definert verktøybaner i CAM for å så postprosessere verktøybanene med *mach3mill.cps*. Den genererte

NC-koden ble så overført til datamaskinen med Mach3 installert og en testkjøring ble utført med mindre hell. Maskinen ga feilmelding på flere av kommandoene og kjørte sporadisk i XY-planet. Det var derfor nødvendig å gjøre endringer i postprosessoren, slik at den genererte maskinkode som fungerer med *Proxxon-FF 500CNC*.

Tabell 3.1 viser G-og M-kodene som ga feilmelding, eller kunne slettes fordi de var overflødige instrukser maskinen ikke klarte å tolke:

Tabell 3.1: G-og M-koder som ga feilmelding(16)

Kode	Kommando
G90	Absolutt koordinater
G94	Matingshastigheten i tommer eller millimeter per sekund. Dette velges i Mach3
G91.1	Inkrementelle koordinater IJK (B og C systemer)
G40	Radiuskompensasjon for verktøy oppheves
G49	Kanseler verktøylengdekompensering
G21	Programmering i tommer eller millimeter, velger dette i Mach3
G54	Nullpunktforflytting
G43+H2	Bruk kompensering for verktøylengde, H er verktøy-offset nr.2
M9	Kjølevæske av (Maskinen har ikke kjølevæske)

I motsetning til andre CAM programvarer på markedet kommer ikke Fusion 360 med en egen programvare for å redigere kildekoden i postprosessorer. For å redigere en postprosessor er det derfor nødvendig med en tredjeparts utviklingsverktøy. I dette prosjektet ble teksteditor-programmet Notepad++ benyttet til å redigere kode. Notepad++ er et gratis tekstprogram som blir anbefalt av Autodesk som utviklingsverktøy for postprosessorer(17).

Ettersom det var mange feilmeldinger og maskinen ble kjørt sporadisk med CNC-koden generert av *mach3mill.cps*, ble CNC-koden sammenliknet med CNC-koden studentene programmerer manuelt i faget TPK4190 (Som er lagt ved i vedlegg D), slik at *mach3mill.cps* kunne generere tilsvarende G-og M-koder. Dermed ble funksjonene i *mach3mill.cps* som genererte de overflødige kommandoene slettet.

G-og M-kodene brukt i NC-programmeringen i faget TPK4190, og som ble beholdt i *mach3mill.cps* forklares i tabell 3.2:

Tabell 3.2: G-og M koder fra TPK4190

Kode	Kommando
M3	Spindel på, rotasjon med klokken
M30	Program slutt
G0	Ilgang
G01	Lineær bevegelse
G03	Sirkulær interpolasjon, beveger seg i en sirkelbane i matingshastighet

NC-koden fra TPK-4190 blir kjørt i to omganger, en for å drille hullene i motorbasen, og en for

å frese arbeidsstykket. Maskinen stopper når alle blokkene i NC-koden for drilloperasjonen er maskinert. Deretter byttes det verktøy og freseprogrammet lastes inn i Mach3 for å så maskinere freseoperasjonen.

Programmet som genereres i CAM seksjonen av Fusion 360, genereres i en fil. Ettersom maskinen ikke har en funksjon for å bytte verktøy automatisk, må maskinen stoppe mellom drill og frese operasjonene for å bytte verktøy manuelt. Det ble derfor lagt til *M1* og *M6* som pauser programmet frem til det får en startkommando fra Mach3 og kjører videre, dermed får maskinoperatøren mulighet til å bytte verktøy manuelt(16).

Ettersom det ikke er programvare utviklet for å redigere postprosessorer i Fusion 360, er det heller ingen medfølgende funksjon for å feilsøke i koden, også kjent som *debugging*. For å feilsøke ble følgende funksjon implementert i begynnelsen av kildekoden til postprosessoren:

```
var deBugPost= 1;
}
if (deBugPost==1) {
  setWriteInvocations(true);
  setWriteStack(false);
}
```

Ved å sette variabelen *deBugPost*=1 vil den genererte utfilen vise hvilke funksjoner som brukes for å generere den påfølgende koden. Dette gjør prosessen for feilsøking vesentlig enklere da man kan gå direkte til funksjonen som genererer koden i Notepad++ og gjøre de nødvendige endringene.

For eksempel hvis følgende kode blir generert med postprosessoren:

```
(2D ADAPTIVE2)
T3 M6
M3
G0 X-3.672 Y-0.975 Z15.
Z5.
Z1.35
G1 Z0.55 F60.
```

Ønskes for eksempel G-koden *G0* (Hurtigmating) i blokken *G0 X-3.672 Y-0.975 Z15* å endres til *G1*, kan variabelen *deBugPost* settes lik 0, og den genererte koden i utfilen vil være:

```
(2D ADAPTIVE2)
T3 M6
M3
!DEBUG: onRapid(-3.67238, -0.975129, 15)
G0 X-3.672 Y-0.975 Z15.
```

```
!DEBUG: onRapid(-3.67238, -0.975129, 5)
Z5.
!DEBUG: onRapid(-3.67238, -0.975129, 1.35)
Z1.35
!DEBUG: onLinear(-3.67238, -0.975129, 0.55, 60)
G1 Z0.55 F60.
```

Fra koden over kan det observeres at *G0* blokken er hentet fra *onRapid*-funksjonen, og kan modifiseres ved å finne *onRapid*-funksjonen i kildekoden til *mach3mill.cps,* i *onRapid*-funksjonen står det at blokken skal skrives ut med *gMotionModal.format(0).* Ønskes det heller en G-kode *G01* endres denne til *gMotionModal.format(1)* slik at *onRapid*-funksjonen i kildekoden til *mach3mill.cps* vil være følgende:

```
function onRapid(_x, _y, _z) {
  var x = xOutput.format(_x);
  var y = yOutput.format(_y);
  var z = zOutput.format(_z);
  if (x || y || z) {
    if (pendingRadiusCompensation >= 0) {
      error(localize("Radius compensation mode cannot be changed at rapid traversal."));
      return;
    }
    writeBlock(gMotionModal.format(0), x, y, z);
    feedOutput.reset();
  }
}
```

Drilling

Tre G-koder som viste seg å være nyttig til drilling var *G98, G81* og *G80.* I tabell 3.3 er funksjonaliteten til de tre G-kodene forklart.

Tabell 3.3: Nyttige G-koder

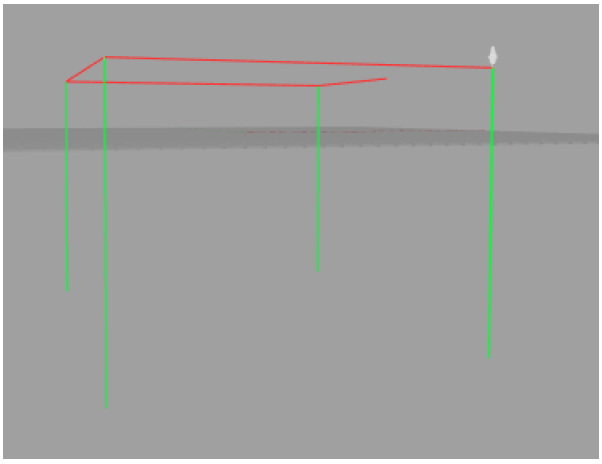
G98	Returner til startposisjonen på Z-aksen når syklusen er ferdig
G81	Enkel drill syklus
G80	Stopp syklus

Figur 3.3 illustrerer verktøybanen med CNC-koden til høyre i tabell 3.4. CNC-kodene gitt i tabell 3.4 viser forskjellen mellom den manuelle programmerte koden brukt i TPK4190 og NC-

Tabell 3.4: Forskjellen i CNC-koden med og uten G80 og G81

TPK4190	**G81 og G80**
M3	
G00 Z10	
X14 Y14	
G01 Z-17 F50	
G00 Z10	M3
X-14 Y14	X14. Z5.
G01 Z-17 F50	G81 X14. Y14. Z-16.262 F50.
G00 Z10	X-14.
X-14 -Y14	Y-14.
G01 Z-17 F50	X14.
G00 Z10	G80
X14 Y-14	
G01 Z-17	
F50	
G00 Z10	

kode med kommandoene G81 og G80. G81 gir instrukser at verktøyet skal gå til X14, Y14 Z-16.262 med mating på 50mm/min. Hver påfølgende NC-blokk vil bruke den forrige posisjonen om X-, Y-og Z-aksen, frem til en ny posisjon om en akse blir definert. Den nye akseposisjonen vil lagres frem til en ny posisjon defineres eller syklusen avsluttes med G-kode *G80*. Dette sparer mye plass i koden og gjør filen mindre og at den kan prosesseres raskere.



Figur 3.3: Simulering av verktøybanen til å drille fire hull

Fresing

For freseoperasjonen ble spindelturtall tatt ut av postprosessoren. Spindelturtallet til maskinen kan ikke endres, og kan dermed kun kjøre i en hastighet. Spindelen begynner å rotere når bryteren på maskinen skrus på, og fortsetter å rotere frem til bryteren skrus av igjen. CNC-koden for å spesifisere spindelturtall bruker adressen *S* etterfulgt av ønsket hastighet i rotasjoner per minutt.

I *User defined properties* er *use G28* definert til *true*. G28 returnerer maskinen til hjemposisjon, men det ble erstattet med at verktøyet skal gå til z+15 etter hver operasjon. I *user defined properties* ble standarden for G28 derfor endret til å være *false*. Forskjellen på CNC-koden til starten av freseoperasjonen før og etter endringene ble utført, er vist i 3.5

Tabell 3.5: NC-koder generert før og etter endringene i postprosessoren *mach3mill.cps*

Original CNC-kode for starten av freseoperasjonen 2D Adaptive, generert av den originale *mach3mill.cps*	Starten av samme verktøybane, med modifisert *mach3mill.cps* tilpasset *Proxxon-FF 500CNC*.
(T3,D=8. CR=0. - ZMIN=-1.5 - FLAT END MILL) G90 G94 G91.1 G40 G49 G17 G21 (2D ADAPTIVE2) M5 M9 T3 M6 S8000 M3 G54 M8 G0 X-3.672 Y-0.975 G43 Z15. H3 Z5. Z1.35 G1 Z0.55 F60. G3 X0.975 Y-3.673 Z0.342 R3.8 X3.673 Y0.975 Z0.133 R3.8 X-0.974 Y3.673 Z-0.075 R3.8	(2D ADAPTIVE2) T3 M6 M3 G0 X-3.672 Y-0.975 Z15. Z5. Z1.35 G1 Z0.55 F60. G17 G3 X3.673 Y0.975 Z0.133 R3.8 X-3.672 Y-0.975 Z-0.284 R3.8 X3.673 Y0.975 Z-0.701 R3.8 X-3.672 Y-0.975 Z-1.118 R3.8

3.3 Eksisterende postprosessorer for Deckel Maho DMU 50 eVolution

På verkstedet på NTNU Valgrinda står det en Deckel Maho DMU 50 eVolution 5-aksemaskin (Figur 3.4). I tillegg til de tre lineære aksene X, Y og Z har maskinen to rotasjonsakser *B* og *C*. B-aksen har et arbeidsområde på [0°-180°], og når B= 0° står B-aksen med en vinkel på 45°fra basen relativt til arbeidsbordet. C-aksen, som er arbeidsbordets rotasjonsakse, har et arbeidsområde på $[-\infty, \infty]$.

Figur 3.4: Deckel Maho DMU 50 eVolution

Maskinen ble kjøpt inn ny rundt årtusenskiftet og bruker styringssystemet Heidenhain TNC 426 med dialogspråket *Heidenhain.h*. Maskinen blir brukt i liten grad, som sannsynligvis skyldes utfordring med programvare og postprosessor, siden Autodesk Fusion 360 er det foretrukne CAD/CAM-programmet, og de eksisterende postprosessorene ikke er konfigurert for Fusion 360. Overgangen til Autodesk Fusion 360 har gjort kvaliteten på arbeidet i verkstedet betydelig bedre, og de ansatte styrer helst unna de eldre eller mindre brukervennlige CAD/CAM programmene.

Det ble undersøkt hvilke post prosessorer som tidligere har fungert med Deckel Maho 50 eVolution og eventuelt hvilke programvare som har vært brukt. Undersøkelser viste at det tidligere har det blitt forsøkt å utvikle ulike postprosessorer for 5-akse maskinering med Deckel Maho DMU 50eVolution, konfigurert for CAM-systemene: NX Manufacture, CAM Express 11.0 og

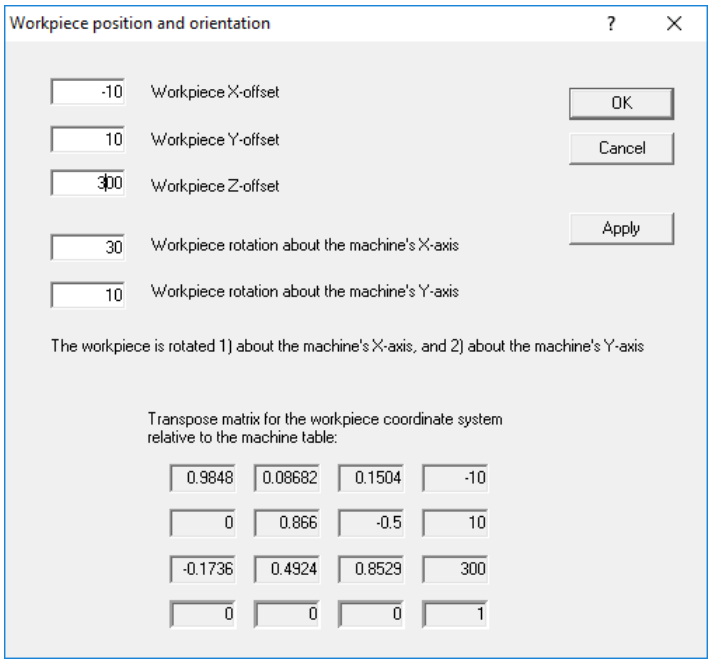
GibbsCam.

3.3.1 Professor Knut Sørby sin frittstående postprosessor

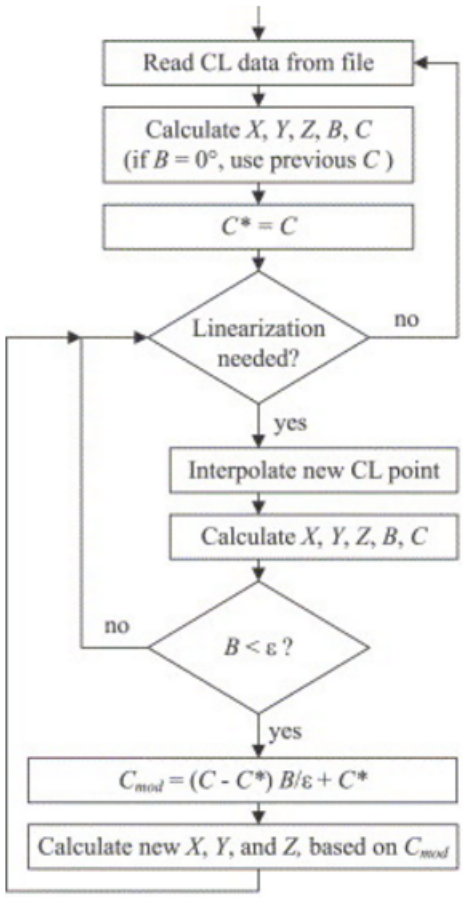
Professor Knut Sørby ved Institutt for maskinteknikk og produksjon, NTNU, har utviklet en frittstående postprosessor for 5-aksemaskinering med Deckel Maho DMU 50 eVolution. Postprosessoren bruker algoritmene for inverskinematikken definert i 2.3.6 til å generere NC-data fra CL-data i form av lineære GOTO-kommandoer

Oppmåling av offset-verdier

Postprosessoren har støtte for programnullpunktverdier, også kjent som offsetverdier, som er målinger med avstanden fra maskinnullpunktet til programnullpunktet i maskinen. Prosessen for oppmåling av offset-verdier blir forklart senere i kapittel 5.3. De oppmålte offset-verdiene implementeres inn i postprosessoren ved å gå til *Workpiece setup* i menyen øverst og velge *Set workpiece position*. Deretter fylles verdiene inn i vinduet vist i figur 3.5, for posisjonsavviket i retningene X, Y, Z og eventuelt rotasjonen til emnet om X og/eller Y-aksen. Postprosessoren bruker så likningene fra kinematikken for posisjon-og orientering forklart i kapittel 2.3.1 og kalkulerer et nytt koordinatsystem for nullpunktet i CL-dataen.



Figur 3.5: Offsetverdier for å definere arbeidsstykkets posisjon og orientering



Figur 3.6: Flytskjema for postprosessering ved en singularitet(11)

Linearisering og singularitetskontroll

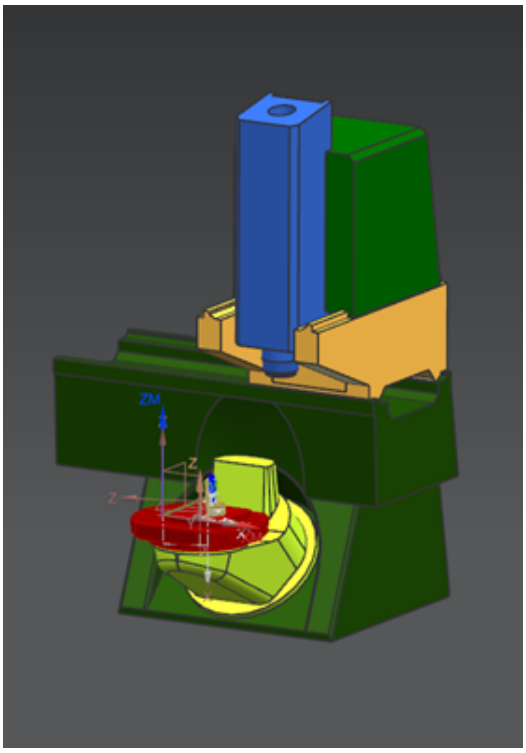
Postprosessoren har støtte for linearisering. Dersom rotasjonsaksene fører til et større avvik enn det som tolereres fra verdiene definert i innstillingene, blir det interpolert midtpunkt mellom NC-blokkene.

Singularitetskontrollen fungerer slik flytskjema i figur 3.6 viser. Dersom det nærmer seg en singularitet ved at B-aksen går mot 0°og C-aksen blir tvunget til å gjøre en rask 180°rotasjon for å bevege gå til neste blokk, vil det interpoleres nye blokker som korrigerer for rotasjonen i de lineære aksene X, Y og Z.

3.3.2 Eksisterende postprosessorer for ulike CAM-systemer

Siemens NX9 med CAM Express 11.0 med Sørby sin postprosessor

I *NX CAM Express* finnes det en modellert maskin som har mål som likner på Deckel Maho DMU 50 eVolution som kan brukes til å simulere maskineringen med fem akser relativt nøyaktig. Ved



Figur 3.7: Dummy-maskin, med mål som likner på deckel maho 50 eVolution

å importere en CAD-modell inn i programmet og plassere delen på arbeidsbordet til den modellerte maskinen, vil delen oppspent i maskinen simuleres som bildet 3.7 illustrerer. Deretter velges verktøy og verktøybanene defineres, og det genereres lineære GOTO-kommandoer som så kan implementeres i Sørby sin frittstående postprossesor som vil generere CNC-kode for Deckel Maho DMU 50 eVolution.

GibbsCAM

GibbsCAM er et populært frittstående CAM-system, med støtte for 3-, 4-og 5-aksefresing med ulike verktøy. Programmet har støtte for to og tredimensjonal geometriske CAD modeller, men sammenliknet med andre fullverdige CAD programvare er verktøyene enkle. GibbsCAM krever lisenser for å brukes og det må kjøpes egne lisenspakker for ulike filformat. Ifølge masteroppgaven til Halvorsen (6) må postprosessorer som støtter GibbsCAM bestilles fra leverandører som spesialiserer seg på utvikling av postprosessorer. Det finnes utviklerverktøy for å tilpasse og lage egne postprosessorer, men utviklerverktøyet er ikke tilgjengelige for vanlige brukere. Uten det utviklerverktøyet blir det ikke mulig å gjøre de nødvendige tilpasningene i en postprossesor. Halvorsen kom frem til i sine undersøkelser at den eksisterende postprosessoren til GibbsCam for Deckel Maho DMU 50eVolution 5-aksefres fungerer godt, men har en systematisk feil i genereringen av maskinkoden. I forbindelse med verktøyveksling i den genererte maskinkoden

må Z0 M92 endres til Z-1 M91.

NX Manufacture

NX Manufacture er en CAM-modul i Siemens NX som er et CAD/CAE/CAM-program for design, analyse og produksjon. Postprosessorer for NX manufacture baserer seg på åpne script og utviklerverktøy for å endre på postprosessorer følger med i installasjonen. I likhet med GibbsCAM er det også mulig å kjøpe skreddersydde postprosessorer fra utviklere som spesialiserer seg i å utvikle postprosessorer. CAM-programmet postprosesseres ved å kjøre en modul med den valgte postprosessoren som generer en fil med NC-kode. Halvorsen kom frem til at den utviklede postprosessoren til NX Manufacture for Deckel Maho DMU 50eVolu-tion 5-aksefres fungerte godt og genererte tilnærmet identisk maskinkode som Sørby sin frittstående postprosessor for de samme operasjonene og han fant ingen feil ved postprosessoren.

3.4 Utvikling av postprosessorer for Fusion 360 og dialogspråket Heidenhain

3.4.1 Konfigurasjon av en eksisterende postprosessor

På Autodesk HSM Post Processor Forum finnes en oppskrift((18) på å konfigurere en postprosessor til å fungere med en 4-eller 5-aksemaskin. Deckel Maho DMU 50 eVolution bruker Heidenhain TNC 426 som styringsspråk. Styringsspråket TNC 407 følger samme ISO programmering som TNC 426, og i Autodesk sitt postprosessor bibliotek(15) finnes det en postprosessor som heter *heidenhain407.cps*.

For å konfigurere postprosessoren for 5-akser ble den første *if*-setningen under funksjonen *onOpen()* i kildekoden til *heidenhain407.cps* endret fra *false* til *true*, for å aktivere maskinkonfigurasjonen med rotasjonsaksene B og C inkludert. Funksjonen ble etter redigeringene seende slik ut:

```
1 function onOpen() {
2   if (true) { // note: setup your machine here
3     var bAxis = createAxis({coordinate:1, table:true, axis:[0, 1, 1],
    range:[0, 180], preference:1});
4     var cAxis = createAxis({coordinate:2, table:true, axis:[0, 0, 1],
    range:[-180, 180], preference:0});
5     machineConfiguration = new MachineConfiguration(bAxis, cAxis);
6     setMachineConfiguration(machineConfiguration);
7     optimizeMachineAngles2(1); // map tip mode
8   }
```

Endringene som ble utført var å utvikle variabelen *bAxis*. Ved å sette *coordinate:1* vet maskinen at koordinaten til *bAxis* er B-aksen. Nummeret til de ulike rotasjonsaksene er: A=0, B=1 og C=2. Videre ble det spesifisert det at rotasjonsaksen er lokalisert i bordet og ikke i verktøyet ved å sette *table:true*.

DMU 50 eVolution har rotasjonsvektoren til basen en vinkel på 45 ° relativt til arbeidsbordet når B-aksen er posisjonert i 0°. Koordinatene til rotasjonsaksene defineres i *axis:* variabelen. Først ble vinkelen med enhetsvektoren [0, -1, 1] definert, men da kom feilmeldingen *Error: Rewind of machine is required for simultaneous multi-axis toolpath.* Så enhetsvektoren [0, 1, 1] ble testet og det ble da skrevet ut kode.

Arbeidsområdet til B-aksen defineres i variabelen *range:*, denne ble satt til [0, 180]. *Preferance* defineres ut ifra om det ønskes positive eller negative vinkler, der positive vinkler defineres med 1 og negative med -1. Dersom det er likegyldig kan *preferance* settes lik 0. Variabelen *cAxis* er rotasjonsaksen med adresse C, Koordinatene ble definert til *axis:* [0, 0, 1] med arbeidsområde på [-180, 180]. *bAxis* og *cAxis* ble definert som en ny maskinkonfigurasjon ved å sette *machineConfiguration = new MachineConfiguration(bAxis, cAxis);*

Et enkelt CAM-program med verktøybaner i simultan 5-akse ble utviklet og deretter postprosessert for å analysere maskinkoden. Uten å ha noe å sammenlikne den genererte maskinkoden med, ble det vanskelig å se om vinklene og posisjonene var korrekte, derfor ble koden testet ved å tørrkjøre programmet i maskinen uten arbeidsstykke og sette verktøyet i en trygg høyde. Nullpunktet ble definert i Z+300 mm over bordet. Maskinen stoppet og ga feilmelding på linje 9 i utfilen: 9 L Z+0 R0 FMAX M91. M91 er en M-kode som blir brukt i starten av et program for å trekke opp verktøyet før bordet eventuelt roterer tilbake til vinklene B og C=0°, slik at ikke verktøyet kolliderer med arbeidsbordet dersom vinklene i maskinen ikke er lik 0° ved programstart. Posisjonskoordinatene i en NC-blokk som slutter med M91 vil bruke maskinens referanse-koordinatsystem M91 i den gjeldenede blokken. Løsningen var å erstatte Z+0 med Z-1. Da programmet ble kjørt på nytt med Z-1 kunne det enkelt observeres at det var tydelig feil i programmet. Ettersom det ikke var noe å sammenlikne CNC-koden med, og det ikke fantes et program som kunne simulere koden måtte alle endringer i postprosessoren testkjøres i Deckel Maho DMU 50 eVolution, noe som krevde mye tid fra de ansatte på verkstedet og prøv og feil metoden ga ingen nyttige resultater.

3.4.2 Fusion 360 og Sørby sin postprosessor

Som nevnt i kapittel 3.3.1 har Sørby utviklet en frittstående postprosessor for 5-aksemaskinering til Deckel Maho 50 eVolution. Postprosessoren leser lineære GOTO-kommandoer, og skriver ut blokker med ISO-kode tilpasset Heidenhain TNC 426.

I Autodesk sitt postprosessor bibliotek finnes det en postprosessor med navnet *APT.CPS*, som skriver ut ISO-kode med lineære GOTO-kommandoer. *APT-CL* (Automatically Programmed

Tool Center Line)(19) er et programmeringsspråk som stammer fra tiden da man skrev maskinkode manuelt. Verktøybanene ble skrevet i *APT-CL* format og postprosessert til å generere riktig ASCII G-kode for å kjøre maskinen. I Alle CAM filer blir det lagret en senterlinje for verktøybanen i en midlertidig fil som er skjult fra brukeren. Det er denne midlertidige filen som sendes til en postprosessor som skal generere en G-kode. Postprosessoren *APT.CPS* bruker den midlertidige filen og konverterer den til APT-CL språk, slik at egenutviklede postprosessorer, som Sørby sin frittstående postprosessor, kan brukes fremfor de utviklet av Autodesk tilpasset Fusion 360.

For å bruke *APT.CPS* med Sørby sin postprosessor må det postprosesserers i to omganger. Først fra Fusion 360 med *APT.CPS* til CL-data med GOTO-kommandoer, som så implementeres inn i Sørby sin postprosessor som da genererer CNC-data som kan lastes over på Deckel Maho DMU 50 eVolution.

En enkel modell ble postprosessert i Fusion 360 med *APT.CPS*, den genererte filen ble så postprosessert i Sørby sin postprosessor. Det resulterte i en liste med følgende feilmeldinger:
Unknown command: MODE/MILL
Unknown command: PARTNO/'testSorbyPP'
Unknown command: PPRINT/'testSorbyPP'
Unknown command: PPRINT/"
Unknown command: UNITS/MM
Unknown command: MULTAX/ON
Unknown command: PPRINT/'Multi-Axis Contour1'
Unknown command: CUTTER/12., 0., 0., 0., 0., 0., 16.
Unknown command: LOADTL/5, 0, 40., 5
Unknown command: END
Unknown command: FINI

Som forklart i avsnittet 3.2.1 finnes det ingen innebygde muligheter for å feilsøke i postprosessorene til Fusion 360, så den samme feilsøkingsfunksjonen ble implementert inn i *APT.CPS*. Ved å feilsøke i *APT.CPS* ble det klart at den ukjente kommandoen MODE/MILL stammet fra onOpen() funksjonen. Kandidaten fikk tilgang til flere CL-filer med GOTO-kommandoer som tidligere har blitt postprosessert og maskinert med suksess på Deckel Maho DMU 50 eVolution. Dermed kunne de tidligere CL-kodene sammenliknes med CL-koden generert av *APT.CPS* og gjøre de nødvendige endringene i *APT.CPS* for å få bort feilmeldingene.

Tabell 3.6 forklarer de ulike endringene som ble utført i *APT.CPS*. For at spindelturtallet skulle inkluderes i den genererte filen, måtte det legges til mellomrom før og etter skråstreken i tillegg måtte *RPM* settes før kommandoen *mainFormat.format(tool.spindleRPM)*, som illustrert i figur 3.8.

Tabell 3.6: Endringer i APT

Unknown command: fra APT	**Eksempelfil**	**Endring i APT**
MODE/MILL	Ingen MODE/MILL kommando	Slettet kommandoen
PARTNO/'testSorbyPP'	PARTNO / TURBINBLAD	La til mellomrom før og etter skråstreken
PPRINT/'testSorbyPP'	Ingen PPRINT kommando	Slettet kommandoen
UNITS/MM	UNITS / MM	La til mellomrom før og etter skråstreken
MULTAX/ON	MULTAX / ON	La til mellomrom før og etter skråstreken
CUTTER/12., 0., 0., 0., 0., 0., 16	$$->CUTTER / 6.000000	Skrevet om nedenfor
LOADTL/5, 0, 40., 5	LOADTL / 1	Skrevet om nedenfor
END	$$->END /	writeln("END"); ble endret til writeln("$$->END / ");
FINI	FINI	Det manglet et mellomrom etter FINI kommandoen writeln("FINI"); ble endret til writeln("FINI ");

```
    writeln("SPINDL/" + mainFormat.format(tool.spindleRPM) + ", RPM, " + (tool.clockwise
 ? "CLW" : "CCLW"));
    writeln("SPINDL / " + "RPM, "+ mainFormat.format(tool.spindleRPM) + (tool.clockwise ?
 ", CLW" : ", CCLW"));
```

Figur 3.8: Endring i spindelhastighet

APT.CPS genererte en verktøyblokk med 7-parametere, der verktøyet var definert ut ifra:
var d = tool.diameter;
var r = tool.cornerRadius;
var e = tool.tipDiameter;
var f = r;
var a = 0;
var b = tool.taperAngle;
var h = tool.fluteLength;

I Sørby sin postprosessor var det av disse syv parameterne kun verktøyets diameter som var nødvendig å definere, så resten av kommandoene ble fjernet. Det ble samtidig lagt til " $$-> " i begynnelsen av blokken og mellomrom før og etter skråstreken. Blokken ble da generert av koden *writeln("$$-> CUTTER / " + mainFormat.format(d))*;

I *APT.CPS* var verktøyet definert av følgende parametere:

var t = tool.number;
var p = 0;
var l = tool.bodyLength;
var o = tool.lengthOffset;

Av disse parameterne var det bare plasseringen av verktøyet i maskinen som var av interesse, derfor ble resten av parameterne fjernet og det ble lagt til mellomrom før og etter skråstreken. Blokken med verktøybytte blir dermed generert av koden *writeln("LOADTL / " + t);*

Den siste feilmeldingen forsvant da *$$* Pro/CLfile Version Wildfire 4.0 - M070* ble lagt inn i begynnelsen av hver CNC-fil generert av *APT.CPS*, slik det gjøres i eksempelfilen. Det ble gjort ved å legge til *writeln("$$* Pro/CLfile Version Wildfire 4.0 - M070");* i starten av *onOpen()* funksjonen

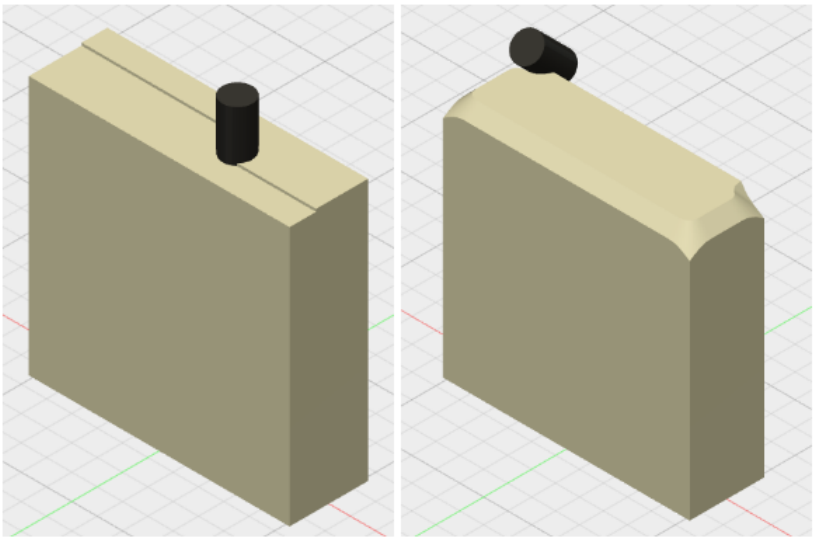
Den genererte koden ble kopiert over i teksteditoren *Notepad* og lagret med *.h* endelse. Gamle filer lagret lokalt på Deckel Maho DMU 50 eVolution, samt filene generert av *Heidenhain407.cps* fra avsnitt 3.4 hadde alle endelse på *.h*. Da den genererte filen fra Sørby sin postprosessor ble lagret med *.h* endelse, ble bare 12kb av koden, som opprinnelig var på flere megabyte, overført til Deckel Maho DMU 50 eVolution og filen var ikke mulig å åpne. Etter mye feilsøking ble det funnet ut at koden skrevet av Sørby sin postprosessor var av ISO-format og ikke generisk Heidenhain språk og skulle dermed lagres med endelsen *.i* og ikke *.h*.

Testkjøring 1

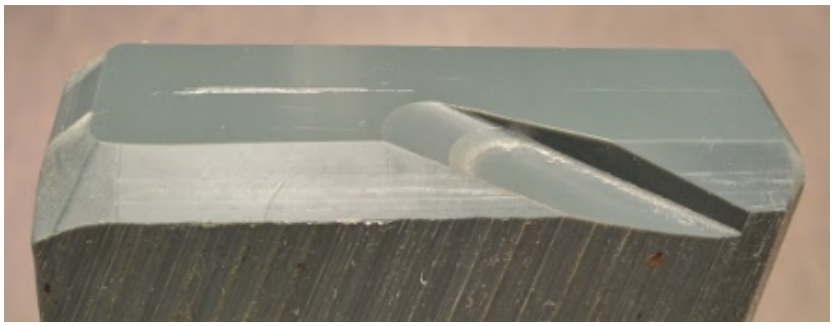
Da programmet omsider ble lastet inn lokalt på maskinen ble programmet først tørrkjørt med lav matingshastighet, uten verktøy eller arbeidsstykke. Programmet virket å kjøre de korrekte verktøybanene, det var dermed klart for å teste programmet med et arbeidsstykke og freseverktøy. Testmodellen skule først planefrese toppen av en plastkloss, og deretter bruke siden av freseverktøyet til å maskinere kanten, slik figur 3.9 illusterer. Arbeidsstykket ble plassert i maskinen, og programnullpunktet ble målt med måleproben i maskinen slik at offsetverdiene fra programnullpunktet til maskinnullpunktet kunne kalkuleres og implementeres i postprosessoren til Sørby.

Da maskinen skulle planfrese arbeidsstykket var det med verktøybaner generert fra en Fusion 360 CAM funksjon med navnet *Face*. *Face* er et 3-akseprogram som krever at vinklene B-og C=0°, men vinklene til rotasjonsbordet ble ikke nullstilt ved programstart, slik at verktøyet kjørte rett inn i arbeidsstykket som figur 3.10 viser. Som illustrert i figur 3.11 var løsningen å legge til vinklene [i, j, k]=[0, 0, 1] for 3-akseprogram, slik at CNC-koden inkluderer vinklene B og C som B+0.000 C+0.000 i hver blokk med 3-aksemaskinering.

Fra *APT.CPS* blir det generert en blokk med kommandoen *Rapid* som i Sørby sin postprosessor genereres til G-koden *G0*. *G0* angir at verktøyet skal beveges med hurtigmating langs



Figur 3.9: Simulering av *Face* og *Swarf* funksjonen



Figur 3.10: Feil vinkel på arbeidsbordet med kommandoen Face

den programmerte verktøybanen, og kan for eksempel brukes til å posisjonere skjæreverktøyet nærme arbeidsstykket etter et verktøybytte. Dersom matingshastigheten tidligere har vært definert, blir den oversett i den gjeldende blokken. *G0* må derfor kun brukes dersom det er sikkert at skjæreverktøyet ikke kolliderer med arbeidsstykket eller oppspenningsverktøy(2). På Deckel Maho 50 eVolution er det ikke mulig å overstyre *G0* som bruker matingshastighet *FMAX*, dermed er det ugunstig når man testkjører et nytt program der man vil manuelt styre matingshastigheten. Rapid ble dermed erstattet med writeln("FEDRAT / 2000., MMPM"); som vil kjøre maskinen med hastighet på 2000 mm/min når det opprinnelig skulle være FMAX, slik at man kan senke hastigheten manuelt dersom 2000 mm/min er for høyt.

Programmet ble kjørt på nytt med det samme arbeidsstykket og denne gangen var det en suksess.

```
function writeGOTO(x, y, z, i, j, k) {
  if (multax) {
    writeln("GOTO/" +
      mainFormat.format(x) + ", " +
      mainFormat.format(y) + ", " +
      mainFormat.format(z) + ", " +
      ijkFormat.format(i) + ", " +
      ijkFormat.format(j) + ", " +
      ijkFormat.format(k)
    );
  } else {
    writeln("GOTO/" +
      mainFormat.format(x) + ", " +
      mainFormat.format(y) + ", " +
      mainFormat.format(z) + ", 0" + ', 0' + ", 1"
    );
  }
}
```

Figur 3.11: Endring av kildekoden til *APT.CPS*

3.4.3 Maskinere en halvkule

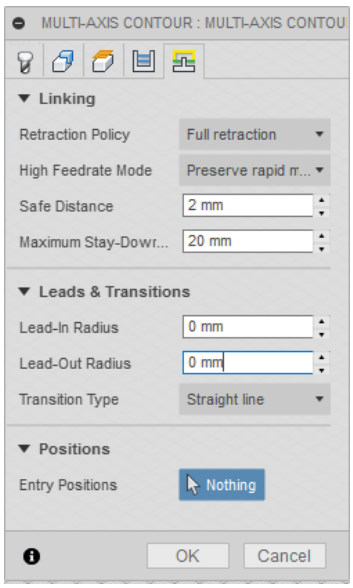
Det var nå klart for å maskinere en halvkule. I vedlegg E ligger en manual på hvordan Autodesk Fusion 360 ble brukt for å modellere halvkulen i CAD og hvordan verktøybanene ble programert i CAM. Manualene er utviklet med hensikt å innføres som en del av undervisningen ved instituttet, samt å kunne anvendes av doktorstipendiater.

Da det *APT.CPS* genererte programmet skulle postprosesserers i Sørby sin postprosessor kom feilmeldingen: *Unknown command: MOVARC/-39.912277, -29.022257, -8.8, 0.968724, -0.248139, 0., 1.2, ANGLE, 90.*

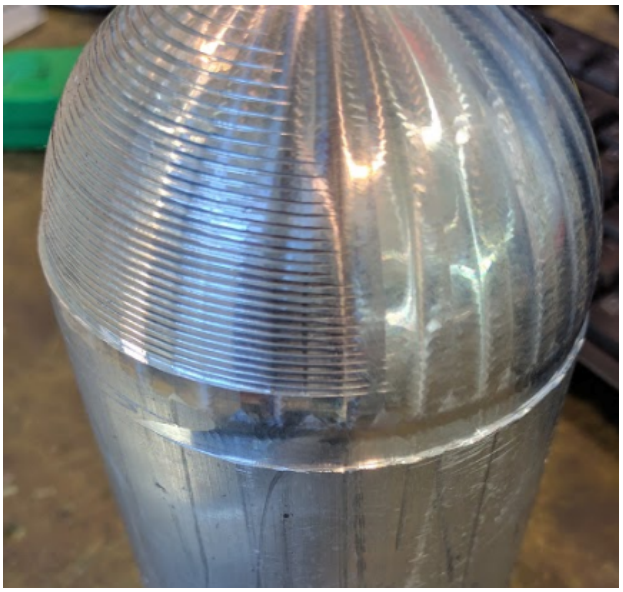
MOVARC er en sirkelfunksjon som beveger verktøyet i en sirkel med en gitt radius. For å unngå at Fusion 360 genererer en MOVARC kommando, kan inn og utgangsradier endres i CAM innstillingene i Fusion 360. Når verktøybanene defineres kan *leads and transitions* under fanen *linking* endres til *straight line* og ingen *lead-in and out radius* som vist i figur 3.12a, verktøyet vil da entre arbeidsstykket med G-koden *G1* og rette linjer.

Programmet ble generert, postprosessert og lastet inn på maskinen. Programmet ble så kjørt uten feil, men freseverktøyet hakket underveis. En forklaring på hakkingen kan være at toleranseverdien var definert for lav slik at det oppsto *data starvation* som forklart i teorikapittelet 2.4.

Da halvkulen var maskinert(figur 3.12b) kunne det observeres at det definerte nullpunktet ikke er sentrert og materialet ble derfor kun avvirket på den ene siden. Det ble derfor definert et nytt nullpunkt. Prosessen for å definere det nye nullpunktet er forklart kapittel 5.



(a) Ulike innstillinger for *Linking* i Fusion 360



(b) Det kan observeres at nullpunktet er forskjøvet

Figur 3.12

3.4.4 Inn og utgangsradier.

Det var ønskelig å endre *MOVARC*-kommandoen slik at man kunne bruke inn-og utgangsradier i Fusion 360 CAM med Sørby sin postprosessor. I eksempelfilen kandidaten fikk tilgang til som tidligere har fungert med postprosessoren til Sørby var sirkelbevegelser definert som *CIRCLE* og ikke *MOVARC*, slik sirkelbevegelser var definert i *APT.CPS*. En av *CIRCLE* kommandoene var:
CIRCLE/90.0000,30.0000,10.0000,0.0000,0.0000,1.0000,51.5000,0.0600,0.5000,10.0000,0.0000
Fra feilsøking av den genererte koden fra *APT.CPS* kunne det observeres at *MOVARC* blokken hentes fra *onCircluar* funksjonen i kildekoden til *APT.CPS*:
!DEBUG: onCircular(false, -39.9123, -29.0223, -8.8, -39.9123, -29.0223, -10, 1440)
Parameterene i *onCircular* funksjonen er:
function onCircular(clockwise, cx, cy, cz, x, y, z, feed)
Autodesk har en side for informasjon om de ulike funksjonsparameternes betydning(20) og i figur 3.13 er parametrene i *onCircular* forklart.

MOVARC returnerer følgende parametere:

```
1  writeln(
2    "MOVARC/" + mainFormat.format(cx) + ", " + mainFormat.format(cy) + ",
   " + mainFormat.format(cz) + ", " +
3    mainFormat.format(n.x) + ", " + mainFormat.format(n.y) + ", " +
   mainFormat.format(n.z) + ", " +
4    mainFormat.format(getCircularRadius()) + ", ANGLE, " + mainFormat.
   format(toDeg(getCircularSweep())));
```

clockwise	Specifies that the motion is clockwise.
cx	The X coordinate of the center position.
cy	The Y coordinate of the center position.
cz	The Z coordinate of the center position.
x	The X coordinate of the end position.
y	The Y coordinate of the end position.
z	The Z coordinate of the end position.
feed	The feedrate.

Figur 3.13: Parameterne i onCircular(20)

Kandidaten har fått tilgang til kildekoden til postprosessoren til Sørby og i *CIRCLE*-kommandoen tar postprosessoren inn parameterne [x, y, z, i, j, k, r, t, f, d, e]. Det var ikke helt klart hva parametrene t, f, d og e var, men det viktige er radiusen og [x, y, z, i, j, k] så t, f, d og e ble erstattet av "dummytall" lik null i kildekoden til *onCircular* i *APT.CPS*, slik at det genereres en *CIRCLE*-kommando fra følgende 11 parametere i kildekoden:

```
1  writeln(
2    "CIRCLE / " + mainFormat.format(cx) + ", " + mainFormat.format(cy) + "
   , " + mainFormat.format(cz) + ", " +
3    mainFormat.format(n.x) + ", " + mainFormat.format(n.y) + ", " +
   mainFormat.format(n.z) + ", " + mainFormat.format(getCircularRadius())
   + ", 0, " + '0, '+ '0, '+ '0');
```

Den genererte *APT.CPS* CNC-koden ble da seende slik ut:
CIRCLE / *-39.912277, -29.022257, -8.8, 0.968724, -0.248139, 0., 1.2, 0, 0, 0, 0*
Da denne blokken ble postprossesert i Sørby sin postprosessor ble den utelatt i den genererte koden, men neste *CIRCLE* kommando som var:
CIRCLE / -69.58992, -21.420322, -10., 0., 0., 1., 30.635792, 0, 0, 0, 0
ble generert til:
N140 G02 X-38.973 Y-22.489 R30.6358 *
Forklaringen på dette var at [i, j, k]=[0, 0, 1] i den nederste koden, slik at det kun er bevegelse i de tre lineære aksene X, Y og Z. I den første koden var [i, j, k]= [0.968724, -0.248139, 0], som vil si at maskinen kjører i et tilfeldig plan og ikke i det forhåndsdefinerte G17 planet. G17 betyr at maskinen opererer i XY-planet. G18 velger XZ-planet og G19 YZ-planet(16). Postprosessoren vil dermed utelate alle *CIRCLE*-kommandoer som ikke er i det forhåndsdefinerte planet. Det ble så undersøkt om det fantes en G-kode for *Heidenhain TNC 426* som kan kjøre i et tilfeldig plan.

I Autodesk Post Library(15) finnes det en annen postprosessor som genererer *ISO*-kode, i beskrivelsen står det *Also note that you can turn on 3D arcs by enabling the 'allow3DArcs' property so you will get arcs in any plane instead of only in the primary planes G17/G18/G19.* Denne bruker G-koden *G68.2* etterfulgt av koordinatene til det nye arbeidsplanet. Det kan så programmeres tilsvarende dersom det var XY-planet *G17*, og avslutter med *G69* for å komme tilbake til G17. Dette er en kommando som virker for språkene *HURCO* og *FANUC*, men ikke for *Heidenhain*. I *Heidenhain* kan G-koden *G80* brukes til å rotere arbeidsplanet, ved å skrive *G80* etterfungt av ønsket vinkel, for eksempel *B+15*. Det nye arbeidsplanet vil nå være rotert 15°om B-aksen, frem til en ny blokk med kun G-koden *G80*, og *G17* arbeidsplanet er gjeldende.

Dersom dette skal importeres i Sørby sin postprosessor må kildekoden skrives om.
En lettvint løsning var å endre *allow helical movements* til *false* under *user defined properties* i *APT.CPS*. Den genererte koden inneholder dermed kun *circle* kommandoer i XY-planet, og for et tilfeldig plan vil den genererte koden inneholde GOTO-blokker med rette linjer. Forskjellen mellom *allow helical movements* lik *true* og *false* illustreres i figurene 3.14 og 3.15.

```
GOTO/-27.271948, -18.171881, -10., 0, 0, 1
CIRCLE / -27.271353, -18.172495, -9.528761, 0.718174, 0.695863, 0., 0.471239, 0, 0, 0, 0
GOTO/-27.486509, -17.950441, -9.884382, 0, 0, 1
```

Figur 3.14: allow helical movements = true

```
GOTO/-27.271948, -18.171881, -10., 0, 0, 1
GOTO/-27.310866, -18.131715, -9.996567, 0, 0, 1
GOTO/-27.349226, -18.092125, -9.98652, 0, 0, 1
GOTO/-27.386485, -18.053672, -9.970001, 0, 0, 1
GOTO/-27.422116, -18.016899, -9.947243, 0, 0, 1
GOTO/-27.455615, -17.982326, -9.918568, 0, 0, 1
GOTO/-27.486509, -17.950441, -9.884382, 0, 0, 1
```

Figur 3.15: allow helical movements =false

Målet med Sørby sin postprosessor var i utgangspunktet at den skulle kunne generere CNC-kode fra verktøybaner programert i Fusion 360 CAM, for deretter å kunne sammenlikne den genererte CNC-koden med en CNC-kode generert direkte fra en postprosessor i Fusion 360, slik at programmet ikke måtte overføres til Deckel Maho DMU 50 eVolution for å analysere koden hver gang noe ble endret i postprosessoren. Nå Autodesk Fusion 360 CAM kunne brukes til å generere CNC-kode som fungerte med alle fem aksene i Deckel Maho DMU 50 eVolution var det mulig å sammenlikne CNC-koden generert av de ferdigprogrammerte Heidenhain postprosessorene i Autodesk sitt postprosessor bibliotek, med CNC-koden generert av Sørby sin postprosessor og gjøre de nødvendige endingene slik at de genererte CNC-kodene ble like. Dermed kan det postprosesseres direkte fra Fusion 360 CAM uten å først gå gjennom *APT.CPS* og Sørby sin frittstående postprosessor.

Heidenhain407.cps, skrevet om i avsnitt 3.4, genererer BNC-kode (Basic NC) og Sørby sin postprosessor skriver ISNC(ISO NC). I Autodesk postprosessor bibliotek finnes det en Heidenhain postprosessor, *Heidenhain iso.cps*, som genererer ISO-kode.

3.5 Postprosessoren Generic Heidenhain ISO

Autodesk har utviklet en postprosessor under navnet *Heidenhain iso.cps* som genererer NC-kode med formatet generisk heidenhain ISO-kode, samme format som NC-koden generert av Sørby sin frittstående postprosessor. *Heidenhain iso.cps* ble derfor brukt som utgangspunkt når en postprosessor tilpasset Deckel Maho DMU 50 eVolution ble utviklet, slik at den genererte maskinkoden enkelt kunne analyseres opp mot maskinkode generert av Sørby sin postprosessor, via *APT.CPS*.

For at en postprosessor skal generere NC-kode direkte fra Fusion 360, tilpasset Deckel Maho DMU 50 eVolution, kan likningene fra inverskinematikken 2.32-2.36 implementeres som en funksjon i kildekoden til postprosessoren. Funksjonen erstatter da funksjonen *createAxis*, som ble forklart i avsnittet 3.4, der postprosessoren ble konfigurert for fem-aksemaskinering ved å sette if(false) til if(true) i onOpen() funksjonen.

Funksjonen *createAxis* ble slettet og erstattet av en funksjon med navnet *invKin*. Funksjonen *invKin* ble programmert inn i kildekoden til *Heidenhain iso.cps* og bruker likningene 2.32-2.36 til å kalkulere verktøyets posisjon og orientering, gitt av maskinaksenes posisjon og vinkler relativt til maskinens nullpunkt. Den programmerte funksjonen ser slik ut:

```
1  function invKin(__x,__y,__z,_i,_j,_k) {
2    var machineX ;
3    var machineY ;
4    var machineZ ;
5    var machineB ;
6    var machineC ;
7    var _x=__x+ properties.xOffset;
8    var _y=__y+ properties.yOffset;
9    var _z=__z+ properties.zOffset;
10
11   if ( _k <0) {
12   error (localize("k can not be negative, change the maximum tilt value
       under the passes tab"
13   + " to be 90 degrees or less from the table koordinates [0,0,1]" ));
14   return
15  }
16
17   if ( _k >= 0) {
18     machineB = Math.acos(2*_k-1);
```

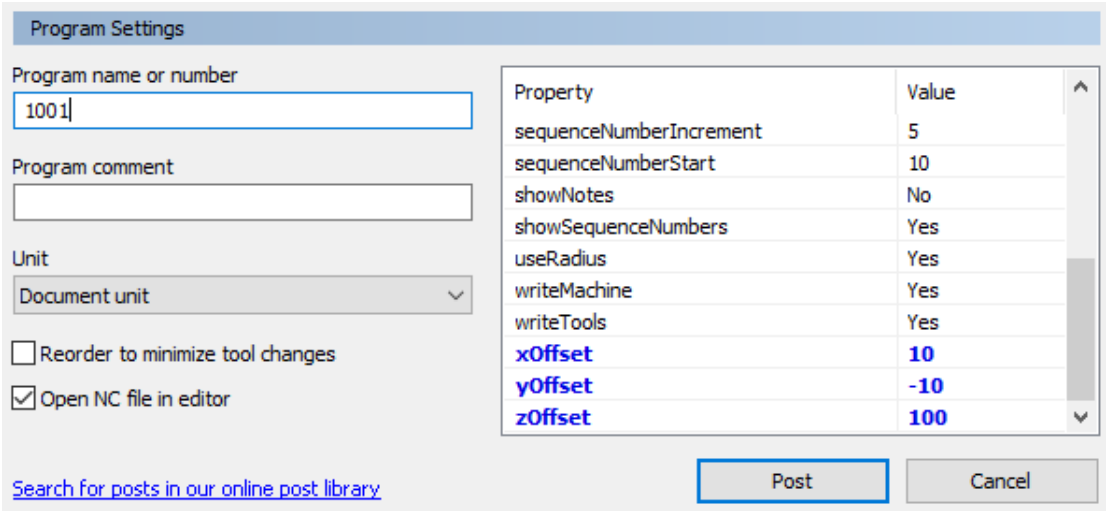
```
19    } else  {
20      machineB = Math.PI;
21      }
22
23    if (_k==1 ) {
24      machineC = 0;
25    } else {
26      machineC=Math.atan2((_i-_i*_k+Math.sqrt(2.0)*Math.sqrt(_k*(1-_k))*_j),
27    (Math.sqrt(2.0)*Math.sqrt(_k*(1-_k))*_i-_j+_j*_k));
28      }
29      if (machineC < 0) {
30      machineC = machineC+ 2*Math.PI;
31      }
32
33    var d = 154.996; // Rotatory table offset
34
35    machineX = _x*Math.cos(machineB)*Math.cos(machineC)+0.5*_x*Math.sqrt
        (2.0)*Math.sin(machineB)*Math.sin(machineC)
36    +_y*Math.cos(machineB)*Math.sin(machineC)-0.5*_y*Math.sqrt(2.0)*Math.sin
        (machineB)*Math.cos(machineC)
37    -0.5*Math.sqrt(2.0)*Math.sin(machineB)*_z+0.5*Math.sqrt(2.0)*Math.sin(
        machineB)*d;
38    machineY = 0.5*_x*Math.sqrt(2.0)*Math.sin(machineB)*Math.cos(machineC)
        -0.5*_x*Math.sin(machineC)
39    -0.5*_x*Math.cos(machineB)*Math.sin(machineC)+0.5*_y*Math.sqrt(2.0)*Math
        .sin(machineB)*Math.sin(machineC)
40    +0.5*_y*Math.cos(machineC)+0.5*_y*Math.cos(machineB)*Math.cos(machineC)
41    -0.5*_z+0.5*_z*Math.cos(machineB)+0.5*d-0.5*d*Math.cos(machineB);
42    machineZ = 0.5*_x*Math.sqrt(2.0)*Math.sin(machineB)*Math.cos(machineC)
        +0.5*_x*Math.sin(machineC)
43    -0.5*_x*Math.cos(machineB)*Math.sin(machineC)+0.5*_y*Math.sqrt(2.0)*Math
        .sin(machineB)*Math.sin(machineC)
44    -0.5*_y*Math.cos(machineC)+0.5*_y*Math.cos(machineB)*Math.cos(machineC)
45      +0.5*_z+0.5*_z*Math.cos(machineB)+0.5*d-0.5*d*Math.cos(machineB);
46
47  return [machineX, machineY, machineZ, machineB, machineC];
48  }
```

invKin-funksjonen tar først inn posisjonsvektoren [x, y, z, i, j, k]. Posisjonen x, y, z hentes fra Fusion 360 CAM i forhold til nullpunktet definert i *setup*. For at likningene for inverskinematikken skal inkludere avstanden fra programnullpunktet til det oppspente arbeidsstykket og ned til maskinnullpunktet, må verdiene fra de målte avstandene inkluderes. Det ble derfor implementert offset-variabler i *user defined properties* slik at brukeren kan skrive inn de målte off-

setverdiene i CAM før det genereres maskinkode.

```
1 // user-defined properties
2 properties = {
3   xOffset: 0,
4   yOffset: 0,
5   zOffset: 0,
```

I *Program Settings*-instillingene til postprosessor-vinduet i CAM defineres de målte verdiene inn i rutene xOffset, yOffset og zOffset slik figuren 3.16 viser.



Figur 3.16: Offsetverdier i *user defined properties* i Fusion 360

invKin-funksjonen tar inn posisjonen *p*=[x, y, z] og bruker offsetverdiene til å transformere posisjonen til en ny posisjon ved å lage nye variabler der den bruker det nye punktet $\_p = [\_x, \_y, \_z]$. Den nye posisjonen er da gitt av $\_p = p + Offset$, der *Offset*=[xOffset, yOffset, zOffset].

I CAM-innstillingene for simultan 5-aksefunksjonene *Swarf* og *Multi-Axis Contour* er verktøyet forhåndsdefinert til å la verktøyet rotere 180°, men dersom verktøyet roterer over 90° om X-aksen, når X_0, Y_0, Z_0=[0,0,1], der Z_0 er parallell med C-aksen som står vinkelrett på arbeidsbordet, når vinklene B og C er lik 0°. Dette fører til at k i [i, j, k] blir negativ og likningen 3.1 fra inverskinematikken for vinkelen B blir ugyldig.

$$B = arccos(2k-1), \ for \ 0° \leqslant B \leqslant 180° \tag{3.1}$$

For at kinematikken skal bli gyldig må $-1 \leqslant k \leqslant 1$. *Maximum Tilt* fra Z-aksen må defineres til 90° eller mindre som vist i figur 3.17, dersom verktøykoordinatsystemet er definert likt som programkoordinatsystemet i *setup*.

Figur 3.17: Maksimum tilt må settes til maks 90°

Det ble implementert en feilmelding i postprosessoren som genereres dersom *k* er negativ. Feilmeldingen illusteres i figur 3.18.

```
21  Generated by: Fusion 360 CAM 2.0.3034
22  ...
23  Error: k can not be negative, change the maximum tilt value under the passes tab to be 90 degrees or less from the table
    koordinates [0,0,1]
24  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
    ^^^^^^^^
25  Error: Failed to execute configuration.
```

Figur 3.18: Feilmelding med negativ k

For å kalkulere vinkelen til C-aksen brukes likning 2.33. På DMU 50 eVolution er arbeidsområdet til C-aksen $[-\infty, \infty]$, men vinklene på maskinen blir presentert på skjermen til kontrollsystemet i arbeidsområdet $[0°, 360°]$. En *if*-setning ble implementert i *invKin*-funksjonen som sjekker om C er negativ, dersom det stemmer legges det til $2 * \pi$ slik at vinkelen blir positiv.

Posisjonen X, Y og Z blir så regnet ut med likningene 2.34-2.36 og funksjonen returnerer variablene [machineX, machineY, machineZ, machineB, machineC].

Funksjonene onLinear og onRapid

Funksjonene *onLinear* og *onRapid* henter inn parametrene [x, y, z] og *onLinear* tar i tillegg inn parameteren *feed*. Funksjonene brukes for å kalkulere NC-punktene langs verktøybanen når det maskineres med tre lineære akser [X, Y, Z]. For at *invKin*-funksjonen skal kalkulere posisjonen [X, Y, Z], må [i, j, k] være definert. I 3-akse maskinering er vinklene B og C lik 0°, slik at [i, j, k] = [0, 0, 1]. I funksjonen *onRapid* og *onLinear* ble variablene [i, j, k] programmert til å bli definert av *currentSection.workPlane.forward* som kodesnutten nedenfor viser:

```
function onRapid(_x, _y, _z) {
 var _i =  currentSection.workPlane.forward.x;
 var _j =  currentSection.workPlane.forward.y ;
 var _k =  currentSection.workPlane.forward.z;
 var[xx, yy, zz, bb, cc]=invKin(_x, _y, _z, _i, _j, _k);
```

currentSection.workPlane.forward gir arbeidsplanet til den ønskede koordinaten og brukes til å regne ut posisjon og vinkler [x,y,z,b,c] fra *invKin*-funksjonen.

Variablene lagres i utskriftsvennlig format. Matingshastigheten ble endret fra *G0* med matingshastighet *FMAX* til *2000mm/min*, med G-kode *G1*, der *G1* angir at bevegelsene mellom to NC-punkt skal foregå med rette linjer, med matingshastighet lik den sist definerte. *G1* forblir aktiv frem til den erstattes av en ny G-kode(2)

I *onLinear*-funksjonen ble tilsvarende endringer utført, bortsett fra at matingshastigheten for en operasjon blir definert av brukeren i Fusion 360 CAM. Kodesnutten under viser hvordan funksjonen *onRapid* ble seende ut:

```
1  var x = xOutput.format(xx);/Runder av tallet og setter X+/- forran
2   var y = yOutput.format(yy);
3   var z = zOutput.format(zz);
4   var b = bOutput.format(bb);
5   var c = cOutput.format(cc);
6   var f = feedOutput.format(2000);
7 //var f = feedOutput.format(feed); i onLinear
8   if (x || y || z) {
9     if (pendingRadiusCompensation >= 0) {
10      error(localize("Radius compensation mode cannot be changed at rapid
      traversal."));
11    }
12    writeBlock(gMotionModal.format(1), x, y, z, b, c, f);
13  }
14
```

I funksjonene *onRapid5D* og *onLinear5D* hentes parameterne [x, y, z, i, j, k] og i *onLinear5d* blir itillegg matingshastigheten *f* hentet inn:

```
1   function onRapid5D(_x, _y, _z, _a, _b, _c) {//input parameterne i
      onRapid5D
2
3   function onLinear5D(_x, _y, _z, _i, _j, _k, feed) {//input parameterne i
      onLinear5d
4
```

I disse funksjonene kan inverskinematikkfunksjonen *invKin* brukes direkte til å finne posisjonen og vinklene [X, Y, Z, B, C], uten å definere nye variabler slik det ble gjort i 3-aksefunksjonene *onRapid* og *onLinear*.

```
1 function onRapid5D(_x, _y, _z, _a, _b, _c) {
2
3   if (pendingRadiusCompensation >= 0) {
```

```
    error(localize("Radius compensation mode cannot be changed at rapid
    traversal."));
    return;
  }

  var[xx, yy, zz, bb, cc]=invKin(_x, _y, _z, _a, _b, _c);
  var x = xOutput.format(xx);
  var y = yOutput.format(yy);
  var z = zOutput.format(zz);
  var b = bOutput.format(bb);
  var c = cOutput.format(cc);
  var f = feedOutput.format(2000)
  writeBlock(gMotionModal.format(1), x, y, z, b, c, f);
  return [x,y,z,b,c];
}
```

CAM-programmet for å maskinere den samme halvkulen som ble postprosessert med Sørby sin postprosessor i avsnitt 3.4.3, ble nå postprosessert med *Heidenhain iso.cps*. For å finne eventuelle feil i koden før den lastes over til maskinen ble de to genererte kodene sammenliknet. Posisjonene og vinklene virket å stemme overens, men blokken for sirkelkommando ble skrevet ut i formatet:

N150 I-69.59 J-21.42 G03 X-38.973 Y-22.489

For å endre sirkelkommandoer til å skrive blokker med posisjon og radius på sirkelen slik det gjøres i postprosessoren til Sørby ble *useRadius* i *user defined properties* endret til å være *true*. Blokken med sirkelkommando skrev nå ut posisjon og radius på sirkelen, men planet ble endret fra G17 til G19 som vist i CNC-koden i figur 3.19. Som forklart i kapittel 3.4.4 betyr G-koden *G17* at maskinen opererer i XY-planet, *G18* i XZ-planet og *G19* i YZ-planet. Sørby sin postprosessor tillater kun *G17* planet for *Circle*-kommandoer, mens *Heidenhain iso.cps* bruker *G17, G18* og *G19* for sirkelkommandoer. For at det skal kun skrives ut sirkelkommandoer i *G17* planet også i *Heidenhain iso.cps*, ble det skrevet en *if*-setning i begynnelsen av *onCircular*-funksjonen. *If*-setningen sjekker om planet er XY med *G17* ved å sjekke om [i, j, k]=[0, 0, 1]. Dersom det ikke er XY-planet vil sirkelbanen lineariseres i en bane med avstanden mellom NC-punktene lik den definerte toleranseverdien i CAM-instillingene. For forklaring av toleranse og linearisering henvises det til kapittel 2.12.

```
N67055 Z+55.313 F1440.
N67060 G19 G02 X+10.928 Y+39.496 Z-27.17 R+1.2
N67065 G01 X+143.106 Y+53.12 Z+54.113
N67070 X+116.944 Y+41.213
```

Figur 3.19: Koden generert av *Heidenhain iso.cps* med G19 G-kode.

```
function onCircular(clockwise, cx, cy, cz, x, y, z, feed) {
  var wpx = currentSection.workPlane.forward.x;
  var wpy = currentSection.workPlane.forward.y;
  var wpz = currentSection.workPlane.forward.z;

if (!currentSection.isMultiAxis() && wpx == 0 && wpy == 0 && wpz == 1) {
//
// her er koden dersom det er XY-planet
//
 else {
    linearize(tolerance);
}
```

Posisjonene og vinklene så ut til å samsvare med koden generert fra postprosessoren til Sørby, og programmet for en halvkule ble lastet inn på Deckel Maho DMU 50 eVolution. Det kom en feilmelding i begynnelsen av programmet på G-koden *G94:Units per minute feed mode*. Figur 3.20 illustrerer begynnelsen av CNC-koden generert fra Sørby sin postprosessor.

```
N10 G40 G90 M126
N15 G01 Z-1.000 F5000 M91
N20 Y-420.000 M91
N25 X-500.000 M91
```

Figur 3.20: Begynnelsen av koden generert av Sørby sin postprosessor.

G og M-kodene betyr følgende(2):

- M126: Traversere korteste bane mellom roterende akser
- G40: Radiuskompensasjon for verktøy oppheves
- G90: Absolutt koordinater

Dersom vinklene B og C i arbeidsbordet ikke er lik 0° når programmet starter, vil bordet roteres til 0°. Derfor vil blokkene som inneholder M-koden *M91* bruke maskinen sitt referansesystem for å trekke verktøyet til en trygg høyde, før bordet eventuelt roterer. I blokkene N20 og N25 i figur 3.20 vil verktøyet trekkes opp til referansepunktet [X, Y, Z]=[-500, -420, -1], som er en trygg høyde helt i hjørnet på maskinen, og verktøyet vil ikke kollidere med arbeidsstykket dersom bordet roteres ved programstart. Kodesnutten nedenfor ble implementert i begynnelsen av *onOpen*-funksjonen i *Heidenhain iso.cps*, slik at hvert program som postprosesseres starter med M-koden *M91*.

```
writeBlock(
      gPlaneModal.format(40), gPlaneModal.format(90), mFormat.format(126)
    );
writeBlock(
      gPlaneModal.format(1), "Z-1.000", "F5000", mFormat.format(91)
    );
writeBlock(
      gPlaneModal.format(1), "Y-420.000", mFormat.format(91)
    );
writeBlock(
      gPlaneModal.format(1), "X-500.000", mFormat.format(91)
    );
```

CAM-programmet med halvkulen ble postprosessert på nytt og lastet over på Deckel Maho DMU 50 eVolution. Kulen ble da maskinert uten feil som illustrert i figur 3.21.

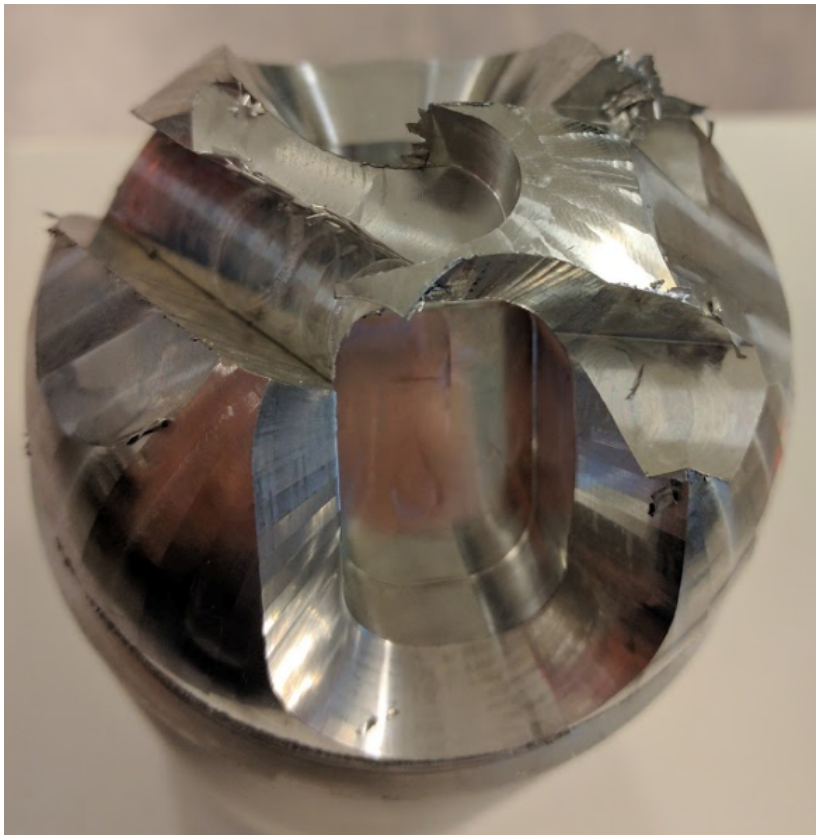
Figur 3.21: Halvkulen maskinert, med råemet saget av.

Testkjøring

For å teste flere funksjoner fra Fusion 360 CAM, ble en testmodell som bygger videre på halvkulen modellert. Manualen i vedlegg E forklarer hvordan Fusion 360 CAD og CAM brukes til å modellere og programmere verktøybaner for å maskinere testmodellen. Testmodellen bruker Fusion 360 funksjonen *Swarf*, som maskinerer med alle fem aksene simultant. Modellen bruker også

ulike 3+2-aksefunksjoner, som først roterer bordet til ønsket posisjon for deretter å bruke de tre lineære aksene for å maskinere emnet. De definerte verktøybanene blir så postprosessert med *Heidenhain iso.cps*.

Mellom de forskjellige operasjonene ble ikke verktøyet trukket opp til trygg høyde i Z før verktøyet beveget seg til neste punkt i XY-planet. Det førte til at verktøybanen til neste XY-posisjon var i samme høyde som slutten av forrige operasjon, og maskinerte derfor tvers gjennom arbeidsstykket slik figur 3.22 illustrerer.



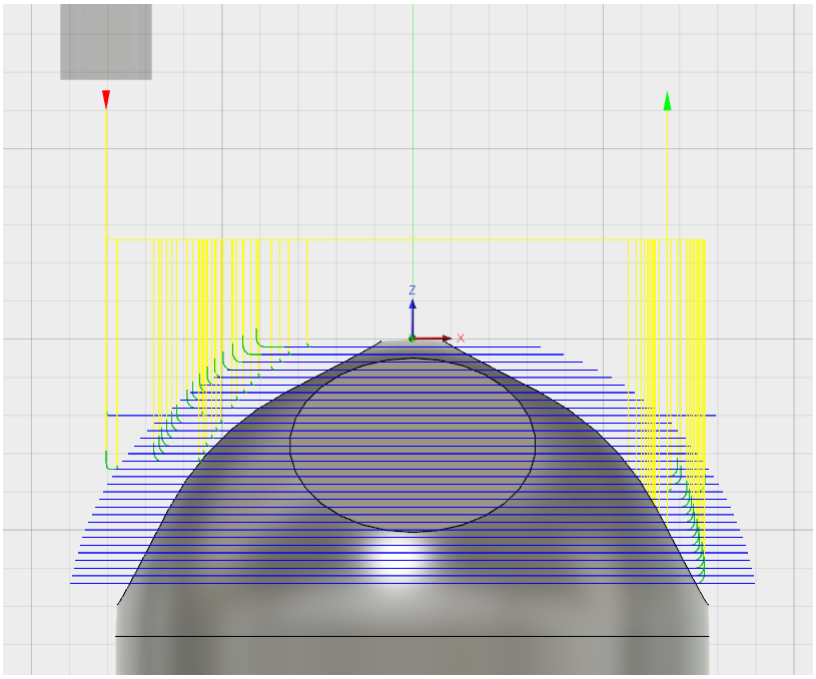
Figur 3.22: Freseverktøyet trakk ikke opp i riktig Z-høyde mellom operasjonene og maskinerte gjennom arbeidsstykket på vei til ny posisjon

I kildekoden til *Heidenhain iso.cps* under *onSetup*-funksjonen var det mellom to forskjellige maskineringsoperasjoner som for eksempel grovfres og finfres, programmert en *if*-setning som sjekker om verktøyet sin nåværende høyde er lavere enn startposisjonen til Z i trygg høyde. Dersom det stemmer trekkes verktøyet opp til startposisjonen før det flyttes i XY-planet. Offsetverdiene brukeren definerer i CAM var ikke inkludert i høyden til Z i denne if-setningen. Dermed trodde verktøyet at det var i en trygg høyde før det flyttet seg til neste XY posisjon, og det ble ikke skrevet ut en ny blokk for å trekke verktøyet opp i Z.

Det ble definert en ny variabel, initialPosZ, som legger offsetverdien til startposisjonen til Z. Startposisjonen til Z er som figur 3.23 illustrerer definert i CAM som klaringshøyden over

nullpunktet i toppen av arbeidsstykket. Den programmerte koden er følgende:

```
1  var initialPosition = getFramePosition(currentSection.getInitialPosition
     ());
2  var initialPosX=(initialPosition.x+properties.xOffset);
3  var initialPosY=(initialPosition.y+properties.yOffset);
4  var initialPosZ=(initialPosition.z+properties.zOffset);
5  if (!retracted) {
6    if ((getCurrentPosition().z+properties.zOffset)< initialPosZ) {
7      writeBlock(gMotionModal.format(1), zOutput.format(initialPosZ));
8  }
9  }
```



Figur 3.23: Start-og sluttposisjonen til verktøyet i forhold til nullpunktet til arbeidsstykket

Koden fungerte fint helt til en testkjøring der det var en ny operasjon da Z var under nullpunktet definert i toppen av arbeidsstykket i CAM slik at Z var negativ. *getCurrentPosition().z* gir ut absoluttverdien til Z, så dersom Z var -110 og offsetverdien var Z+300, regnet if setningen ut at *(getCurrentPosition().z+properties.zOffset)* var 110+300, som da er høyere enn initialPosZ som er høyden av *clearance* + *retract height* i CAM + offsetverdien til Z for den gjeldende operasjonen. Funksjonen ble derfor endret til å trekke freseverktøyet 10mm over offsetverdien for nye operasjoner, med følgende kode:

```
1  if (!retracted) {
2        writeBlock(gMotionModal.format(1), zOutput.format(properties.zOffset
     +10));
3   }
```

Det ble kjørt en ny testkjøring med en ny modell med verktøybytte underveis. For nye operasjoner med verktøybytte gikk freseverktøyet til *initialPosZ* før neste operasjon. For en operasjon nede på arbeidsstykket, der høyden av *clearance* + *retract height* i CAM var lavere enn Z_0 gikk freseverktøyet veldig nærme arbeidsstykket før det begynte å bevege seg i XY-planet og rotere arbeidsbordet. Det ble ingen kollisjon og programmet så ut til å ha kontroll, men for å være på den sikre siden ble koden for nye operasjoner med verktøybytte endret til å sjekke om den første Z-verdien er større enn offsetverdien til Z. Dersom den er det går verktøyet til *initialPosZ*, ellers trekkes verktøyet 10mm over offsetverdien til Z. Variabler for startverdien til X og Y ble definert på samme måte som *initialPosZ*, men med variabelnavn *initialPosX* og *initialPosY*. *G0* ble også endret til *G1* i hele postprosessoren. Den implementerte koden ble seende slik ut:

```
1  if(initialPosZ < properties.zOffset){
2    writeBlock(gMotionModal.format(1), zOutput.format(properties.zOffset+10)
       );
3  }
4  else {
5  writeBlock(gMotionModal.format(1), zOutput.format(initialPosZ));
6      writeBlock(
7      gAbsIncModal.format(90),
8      gMotionModal.format(1), xOutput.format(initialPosX), yOutput.format(
     initialPosY));  }
```

Implementering av program-kommentarer

For hver ny operasjon begynner NC-filen med en kommentar som inneholder navnet på operasjonen i CAM og en kommentar med informasjon om verktøyet. Kommentarer ble opprinnelig skrevet ut som en linje, og ikke en blokk med N etterfulgt av blokknummeret. Maskinen klarte ikke å tolke disse kommentarene og ga ut feilmelding. En kommentarblokk ble derfor endret til å skrive ut en blokk med nummer. I *Heidenhain ISO programming user's manual*(21) står det som figur 3.24 illustrerer at en kommentar må ha et semikolon forran kommentaren.

Det ga også ut feilmelding da programmet skulle kjøres på maskinen. Løsningen ble å gi en kommentar en blokk etterfulgt av semikolon med kommentaren i en parentes.

Example

```
.
.
.
N50     G00  X+0 Y–10 *
;       PRE-POSITIONING ..... A comment is indicated by a semicolon at the beginning of the block.
N60     G01 G41 F100 *
.
.
.
```

Figur 3.24: Kommentar fra ISO manualen(21)

Den første blokken i NC-filen inneholder navnet på programmet, denne blokken måtte endres til å være en linje uten blokknummer.

For å huske offsetverdiene som brukes i et program ble en blokk med offsetverdiene i en kommentar lagt til begynnelsen av programmet. Figur 3.25 illustrerer hvordan den genererte NC-filen ser ut med endringene sammenliknet med en *Heidenhain iso.cps* postprosessor uten endringer.

```
N10 %before G71
N15 G30 G17 X-40. Y-40. Z-150.
N20 G31 G90 X+40. Y+40. Z+0.
(T5  D=12. CR=0. - ZMIN=-32. - FLAT END MILL)
N25 G94
(ADAPTIVE1)
N35 M09
N40 T5 G17 S12000
N45 S12000 M03
N50 M08
N60 G00 X-40.21 Y-30.185
N65 Z+30.
N70 G00 Z+13.
N75 Z-7.6
N80 G01 Z-8.8 F1440.
N85 X-40.208 Y-30.177 Z-8.934
N90 X-40.203 Y-30.156 Z-9.067
N95 X-40.193 Y-30.119 Z-9.196
N100 X-40.181 Y-30.07 Z-9.321
N105 X-40.164 Y-30.007 Z-9.438
```

```
%After G71
N10 ;(Offset values: X: 0 Y:  0 Z:  300 )
N15 G40 G90 M126
N20 G01 Z-1.000 F5000 M91
N25 Y-420.000 M91
N30 X-500.000 M91
N35 G30 G17 X-40. Y-40. Z+150.
N40 G31 G90 X+40. Y+40. Z+300.
N45 ;(T5  D=12. CR=0. - ZMIN=-32. - FLAT END MILL)
N50 ;(ADAPTIVE1)
N55 M09
N60 T5 G17
N65 S12000 M03
N70 M08
N80 G01 Z+330.
N85 X-40.21 Y-30.185
N90 G01 Z+313.
N95 Z+292.4
N100 Z+291.2 F1440.
N105 X-40.208 Y-30.177 Z+291.066
```

Figur 3.25: Starten av programmet generert før og etter endringene i koden

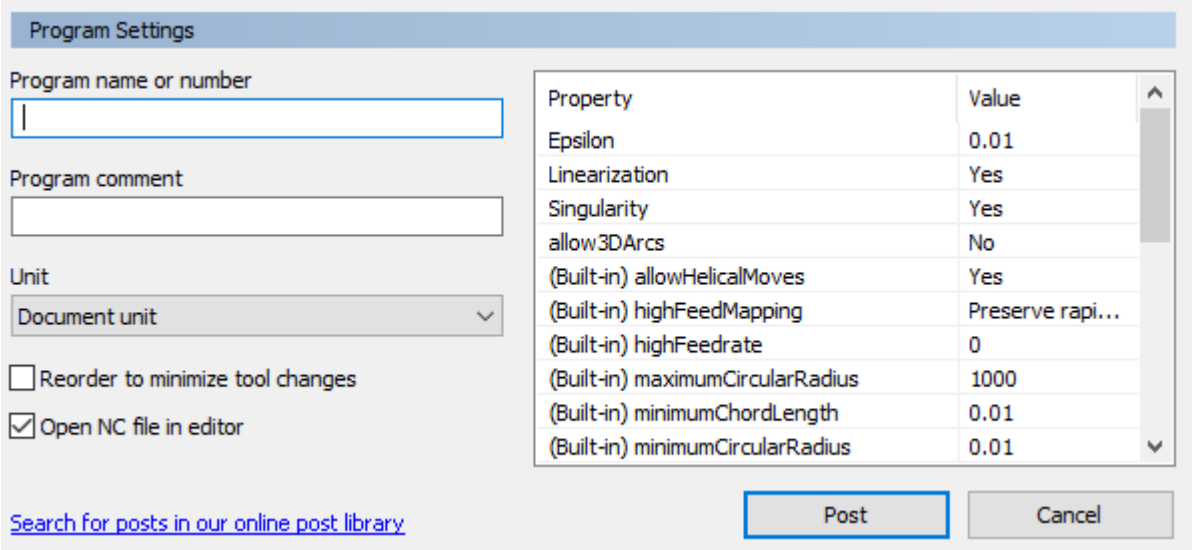
Kapittel 4

Linearisering-og Singularitetskontroll

Behovet for linearisering oppstår som forklart i kapittel 2.6 for å kontrollere avviket som oppstår grunnet rotasjonsaksene i en CNC-maskin.

Behovet for en singularitetskontroll oppstår som forklart i kapittel 2.5 når en av rotasjonsaksene er parallelle med verktøyaksen. Det ble utviklet en singularitet-og lineariseringskontroll som en del av funksjonen *onLinear5D* i postprosessoren *heidenhain iso.cps*.

For å bruke kontrollen ble det lagt til brukerdefinerte variabler der brukere kan skru kontrollen på/av, ved å sette *Linearization* og *Singularity* til *Yes*/*No*, slik figur 4.1 illustrerer.



Figur 4.1: Vindu for å skru på linearisering og singularitetskontroll i Fusion 360

4.1 Lineariseringskontroll

OnLinear5D tar som forklart i kapittel 3.5 inn CL-data med posisjonen [x, y, z, i, j, k]. De oppmålte offsetverdiene i X, Y og Z legges så til [x, y, z], inverskinematikkfunksjonen regner så ut posisjonen og vinklene i NC-data [X, Y, Z, B, C].

Posisjons-og vinkelavviket regnes ut ved å finne avviket mellom to CL-datapunkt CL_{mid} og CL_{mid-ny} som i utgangspunktet burde være like, men rotasjonsaksene fører til en posisjonsfeil.

Midtverdien av CL

Verdiene til CL_{mid} finnes ved å ta midtpunktet mellom det nåværende NC-datapunktet og neste NC-datapunkt. Nåværende NC-data hentes ved å bruke funksjonen $(x,y,z,b,c)Output.getCurrent()$;, som gir ut nåværende posisjon og vinkler i form av NC data. Neste NC-datapunkt regnes ut ved å bruke likningene for inverskinematikk som forklart i kapittel 3.5. Funksjonen som går under navnet *invKin* i kapitell 3.5 ble endret navn til *CLtoNC* for å at koden skulle bli ryddigere.

En ny variabel NC_{mid}, regner ut midtpunktet mellom de to NC-datapunktene. Midtpunktet regnes ut med funksjonen *ncMid*:

```
1  function ncMid(nc1, nc2){
2  var nc1b = nc1[3];
3  var nc1c = nc1[4];
4  var nc1x= nc1[0];
5  var nc1y= nc1[1];
6  var nc1z = nc1[2];
7
8  var nc2b = nc2[3];
9  var nc2c = nc2[4];
10 var nc2x= nc2[0];
11 var nc2y= nc2[1];
12 var nc2z = nc2[2];
13
14 var nc_mid=[(nc1x+nc2x)/2, (nc1y+nc2y)/2, (nc1z+nc2z)/2, (nc1b+nc2b)/2, (
       nc1c+nc2c)/2];
15
16 return [nc_mid];
17 }
```

Variabelen CL_{mid} regnes så ut fra NC_{mid}. For å regne ut CL-data fra NC-data må likningene for direkte kinematikk i kapittelet 2.3.3 brukes. Disse ble implemetert i postprosessoren som en egen funksjon *NCtoCL*:

```
function NCtoCL(aa, bb, cc, machineB, machineC)
{
  var d = 154.996; // Rotatory table offset
var ii = -0.5*Math.cos(machineB)*Math.sin(machineC)+0.5*Math.sqrt(2.0)*
    Math.sin(machineB)*Math.cos(machineC)+0.5*Math.sin(machineC);
var jj =  0.5*Math.sqrt(2.0)*Math.sin(machineB)*Math.sin(machineC)-0.5*
    Math.cos(machineC)+0.5*Math.cos(machineB)*Math.cos(machineC);
var kk = 1/2+(1/2)*Math.cos(machineB);


var xx = -0.5*Math.cos(machineB)*Math.sin(machineC)*bb-0.5*Math.sqrt(2.0)*
    Math.sin(machineB)*Math.cos(machineC)*d
        +0.5*Math.sqrt(2.0)*Math.sin(machineB)*Math.cos(machineC)*cc+0.5*
    Math.sqrt(2.0)*Math.sin(machineB)*Math.cos(machineC)*bb
        -0.5*Math.sin(machineC)*d+0.5*Math.sin(machineC)*cc+0.5*Math.sqrt
    (2.0)*Math.sin(machineB)*Math.sin(machineC)*aa
        -0.5*Math.sin(machineC)*bb+0.5*Math.cos(machineB)*Math.sin(
    machineC)*d-0.5*Math.cos(machineB)*Math.sin(machineC)*cc+Math.cos(
    machineC)*aa*Math.cos(machineB);
var yy = -0.5*Math.sqrt(2.0)*Math.sin(machineB)*Math.sin(machineC)*d+0.5*
    Math.sqrt(2.0)*Math.sin(machineB)*Math.sin(machineC)*cc
        +0.5*Math.sqrt(2.0)*Math.sin(machineB)*Math.sin(machineC)*bb+0.5*
    Math.cos(machineC)*d-0.5*Math.cos(machineC)*cc
        -0.5*Math.sqrt(2.0)*Math.sin(machineB)*Math.cos(machineC)*aa+0.5*
    bb*Math.cos(machineC)-0.5*Math.cos(machineB)*Math.cos(machineC)*d
        +0.5*Math.cos(machineB)*Math.cos(machineC)*cc+0.5*Math.cos(
    machineB)*Math.cos(machineC)*bb+Math.sin(machineC)*aa*Math.cos(machineB
    );
var zz =  0.5*bb*Math.cos(machineB)+0.5*cc-0.5*Math.sqrt(2.0)*Math.sin(
    machineB)*aa+0.5*Math.cos(machineB)*cc
        +0.5*d-0.5*bb-0.5*d*Math.cos(machineB);

return [xx, yy, zz, ii, jj, kk];
}
```

Ny midtverdi av CL

Det andre CL-datapunktet er CL_{mid-ny} som er midtpunktet mellom det nåværende CL-datapunktet CL_1 og det neste CL-datapunktet CL_2. CL_2 hentes direkte ut fra $function\ onLinear5D(\_x, \_y, \_z, \_i, \_j, \_k, feed)$, og offsetverdiene definert av brukeren legges til _x, _y og _z.

CL_1 regnes ut fra algoritmene for inverskinematikk fra nåværende NC-data, NC_1. For å regne midtpunktet mellom CL_1 og CL_2 ble en funksjon tilsvarende *ncMid*, bare med CL-data istedenfor NC-data, implementert.

Posisjon-og vinkelavvik

Det ble implementert en funksjon *errorxyzijk* for å regne ut posisjons-og vinkelavviket grunnet rotasjonsaksene :

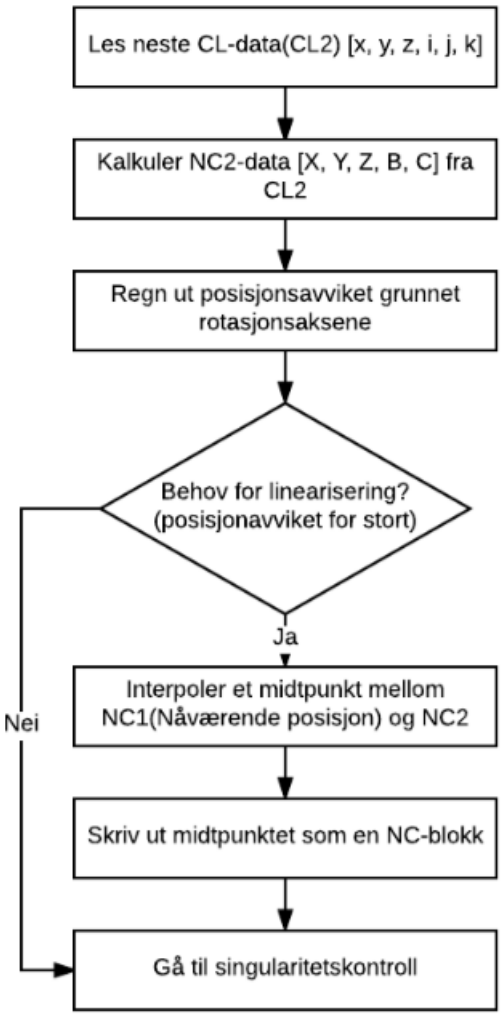
```
function errorxyzijk(clmidnew, clmid){

var errorx = Math.abs(clmidnew[0]-clmid[0]);
var errory = Math.abs(clmidnew[1]-clmid[1]);
var errorz = Math.abs(clmidnew[2]-clmid[2]);
var errori = Math.abs(clmidnew[3]*clmid[3]);
var errorj = Math.abs(clmidnew[4]*clmid[4]);
var errork = Math.abs(clmidnew[5]*clmid[5]);
var pos_error = (Math.sqrt(errorx*errorx + errory*errory + errorz*errorz))
   *Math.PI/180;


var dot = (errori+errorj+errork)
if (dot>1){
  dot=1;
    }
var ang_error = Math.acos(dot);
return[pos_error, ang_error];
}
```

Lineariseringsfunksjonen

Algoritmen utviklet for lineariseringsskontrollen følger flytskjemaet i figur 4.2, og ble programert inn i *Heidenhain iso.cps* med kodesnutten nedenfor. Algoritmen sjekker først om posisjonsavviket er større enn 0.0001mm og mindre enn 0.01mm. Dersom det stemmer blir en ny blokk skrevet ut med verdier midt mellom NC_1 og NC_2. Grunnen til at det sjekkes om posisjonsavviket er mindre enn 0.01mm er at dersom for eksempel vinkel C fra NC_1 er 359°og vinkel C fra NC_2 er 0.1°, blir posisjonsavviket større enn 0.01, men det skal ikke interpoleres en NC-blokk med C=179.55° mellom NC_1 og NC_2.

```
var [NC_mid]=ncMid(nc1, nc2);
var CL_mid=NCtoCL(NC_mid[0],NC_mid[1],NC_mid[2],NC_mid[3],NC_mid[4]);
var [CL_mid_new]=clMid(cl1,cl2);
```

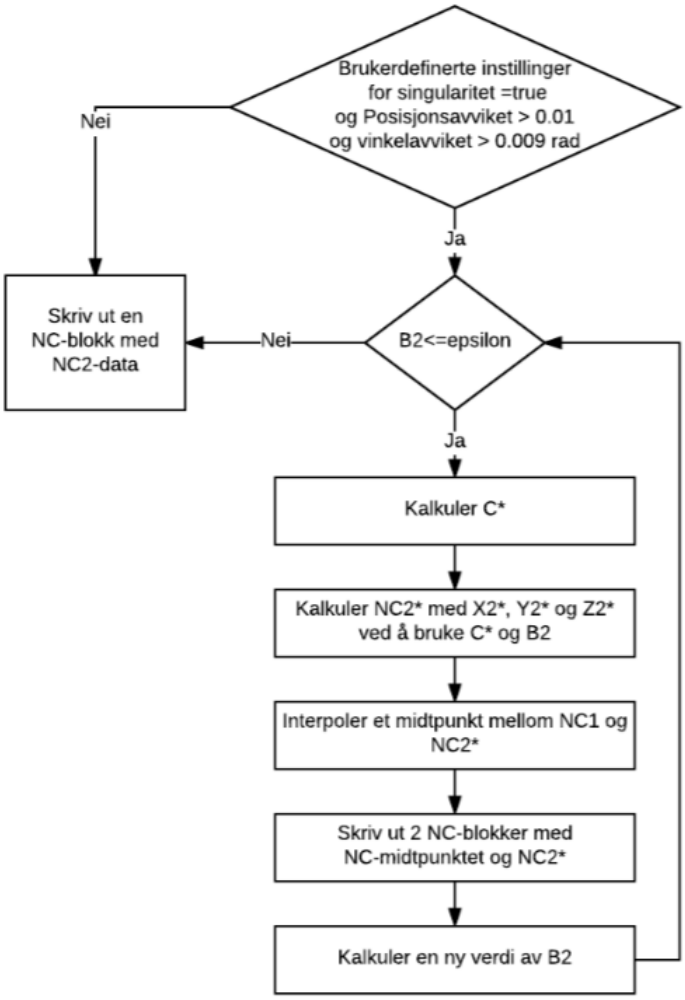
```
4 var [poserror, angerror]=errorxyzijk(CL_mid_new, CL_mid);
5
6 if (poserror>0.0001 && poserror<0.01){
7 writeBlock(gMotionModal.format(1), xOutput.format(NC_mid[0]), yOutput.
    format(NC_mid[1]),
8 zOutput.format(NC_mid[2]), bOutput.format(NC_mid[3]), cOutput.format(
    NC_mid[4]));
9 }}
```



Figur 4.2: Algoritme for linearisering i postprosessoren

4.2 Singularitetskontroll

Algoritmen utviklet for singularitetskontroll følger flytskjemaet i figur 4.3, som flytskjemaet illus-



Figur 4.3: Algoritme for singularitetskontroll i postprosessoren

trerer starter singularitetskontrollen med å sjekke om posisjonsavviket og vinkelavviket er større enn henholdsvis 0.01mm og 0.009 radianer. Dersom det stemmer sjekker den om vinkel B_2 er større enn 0.01 radianer, om det stemmer deles B_2 på 100 slik at den blir mindre enn epsilon(ε) som er satt til 0.01, men kan endres i de brukerdefinerte-innstillingene. Grunnen til at vinkel B_2 deles på 100 er fordi det oppstod en singularitet i en testmodell med stor toleranseverdi mellom to NC-blokker med B_1=0.015 rad og B_2=0.0105 rad. Vinkelen ε var definert til 0.01 rad, slik at vinkel B_2 aldri ble lavere 0.01 rad og singularitetskontrollen startet ikke. Problemet ble løst ved å dele B_2 på 100. Gapet mellom B_2 og den kommende B_3 blir da større, men det er såpass lite utslag at det tolereres.

Vinkelen ε blir definert til en lav verdi av brukeren i innstillingene til postprosessoren, og er definert som 0.01 radianer som standard. Dersom B_2 er mindre enn ε, går koden inn i en *while-loop* frem til B_2 blir større enn ε igjen.

Det lages så en ny vinkel C_{mod} som bruker en modifisert versjon av likningen 2.38.

$$If\ C_1 > C_2\ \ then\ C_{mod} = C_1 + (C_1 * \frac{B_2}{\varepsilon}) \tag{4.1}$$

$$If\ C_1 < C_2\ \ then\ C_{mod} = C_1 + ((C_2 - C_1) * \frac{B_2}{\varepsilon}) \tag{4.2}$$

Deretter sjekker algoritmen om C_{mod} er innenfor arbeidsområdet [0-2π], og det legges til eller trekkes fra 2π dersom det ikke stemmer.

Posisjonen [X, Y, Z] i det nye NC_2 datapunktet regnes ut ved å bruke likningene for inverskinematikk med CL_2-posisjonene [x, y, z](+ offsetverdiene) og vinklene B_2 og C_{mod}. Det ble implementert en egen funksjon som er lik *CLtoNC*-funksjonen, bare at den tar inn vinklene B og C istedenfor å regne de ut fra vektoren [i, j, k]. Funksjonen *singular* tar inn vektoren [x, y, z, B, C] og returnerer posisjonen X, Y, Z. For å være på den sikre siden interpoleres det inn et ekstra datapunkt mellom det gamle og nye NC-datapunktet, på samme måte som det interpoleres et nytt punkt dersom lineariseringskontrollen slår ut. Det ville sannsynligvis gitt et like godt resultat ved å heller dele på *400* istedenfor *200* i likning 4.3, men for å unngå et avvik fra rotasjonsaksene ble det valgt å interpolere. Deretter skrives det ut to NC-blokker med det interpolerte og det nye NC-datapunktet.

B_2 regnes ut på nytt med likning 4.3:

$$B_2 = B_2 + \frac{\varepsilon - B_2^*}{200} \qquad B_2^*\ er\ den\ originale\ verdien\ av\ B_2\ før\ while-løkken \tag{4.3}$$

Verdien *200* i likning 4.3 ble definert for at det skulle genereres nok NC-blokker for å håndtere en singularitet. Ved å sammenlikne antall NC-blokker for håndtering av en lik singularitet i Sørby sin postprosessor virket det som at det ble generert mer enn nok NC-blokker for å håndtere singulariteten med verdien *200* i likning 4.3.

While-løkken skriver ut nye blokker med NC-data frem til $B_2 > \varepsilon$ og den skriver ut det endelige NC_2-blokken. Koden for algoritmen til singularitetskontrollen i *heidenhain iso.cps* er følgende:

```
1  if ( poserror>0.01 && angerror>0.009 &&  properties.Singularity) {
2 var epsilon= (properties.Epsilon);
3  if(nc2b>0.01){
4  nc2b=nc2b/100;
5  }
6  var nc2bb=nc2b;//Variable to store the original nc2b value
7  var nc22b=nc2b;// variable to store the previous value of nc2b in the
```

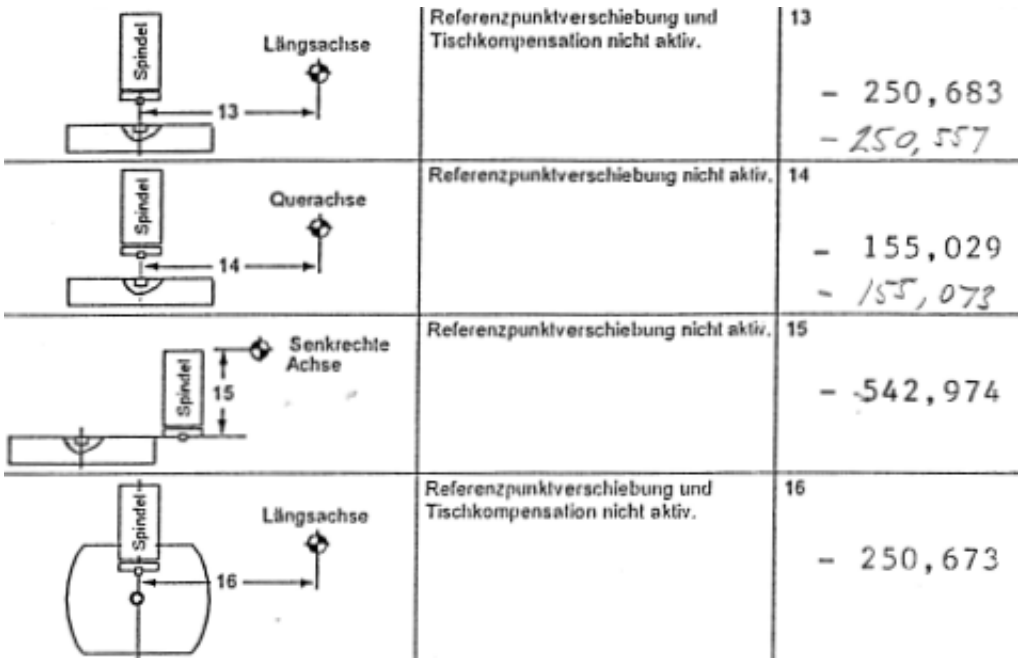
```
    while loop
  //writeBlock("NC2b", nc2c);
while (nc2b <= epsilon){ //Is B smaller than epsilon?

var C = nc1c+(nc1c* (nc2b / epsilon));

if (nc1c<nc2c){
  C=nc1c +(nc2c-nc1c)*(nc2b/epsilon);}

if (C>2*Math.PI){
    C=(C-2*Math.PI);
  } else if( C<0){
    C=(C+2*Math.PI);
  }
var [nc2x, nc2y, nc2z] = singular(cl1[0],cl1[1],cl1[2], nc2b, C);

var cl2mid=NCtoCL(nc2x,nc2y,nc2z,nc22b,C);

var [cl1mid]=clMid(cl1, cl2mid);

var [ncmidx, ncmidy, ncmidz, ncmidb, ncmidc]=CLtoNC(cl1mid[0],cl1mid[1],
    cl1mid[2],cl1mid[3],cl1mid[4],cl1mid[5]);

writeBlock(gMotionModal.format(1), xOutput.format(ncmidx), yOutput.format(
    ncmidy),
 zOutput.format(ncmidz), bOutput.format(ncmidb), cOutput.format(ncmidc));

writeBlock(gMotionModal.format(1), xOutput.format(nc2x), yOutput.format(
    nc2y),
 zOutput.format(nc2z), bOutput.format(nc2b), cOutput.format(C));

cl1=NCtoCL(nc2x,nc2y,nc2z,nc2b,C);
nc22b=nc2b;
nc2b=nc2b+((epsilon-nc2bb)/200);

 }}
```

Kapittel 5

Oppmåling av nytt nullpunkt

Når et arbeidsstykke skal bearbeides må CNC-styringen, som forklart i kapittel 2.2.6, vite hvor emnet befinner seg slik at posisjonen og orienteringen til det oppspente arbeidstykket i maskinen stemmer overens med arbeidsstykket definert i CAM.

På Deckel Maho DMU 50 eVolution er maskinnullpunktet(figur 5.1) definert med XY-planet på overflaten til rotasjonsbordet, og Z-aksen i sentrum slik at den er kolineær med C-aksen. En verktøymaskin vil etter mye bruk oppleve både naturlig slitasje, samt slitasje fra ting som verktøykræsj. Slitasje på målesystemene fører til feilposisjonering som igjen fører til at nullpunktet blir forskøvet og det må tidvis kalibreres et nytt nullpunkt i overflaten til rotasjonsbordet som er kolineært med C-aksen.



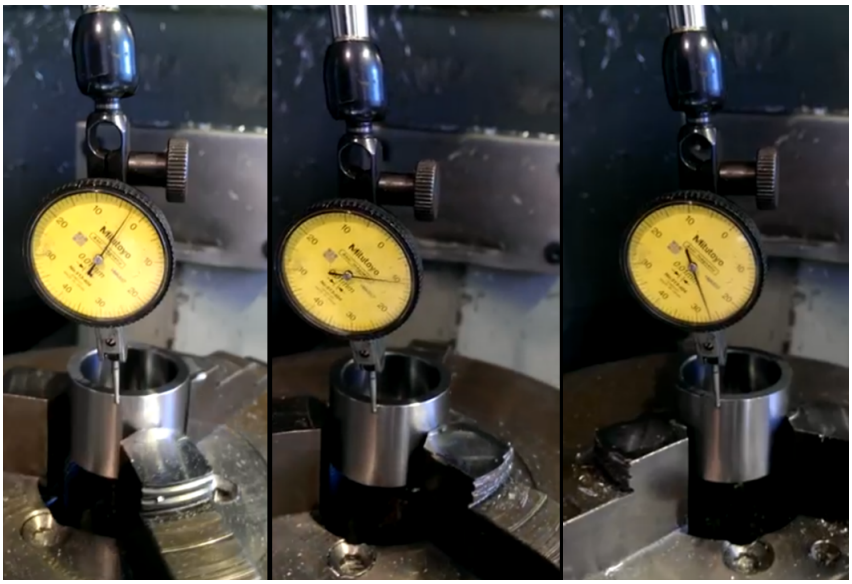
Figur 5.1: Maskinnullpunktet til Deckel Maho DMU 50 eVolution fra maskinmanualen, med avstanden målt fra referansepunktet til maskinen.

Som forklart i avsnitt 3.4.3 kunne det observeres fra maskineringen at nullpunktet definert i rotasjonsbordet var feil og det måtte kalibreres et nytt nullpunkt.
En kjoks med 3 bakker ble plassert i maskinen og brukes for å spenne opp akslinger i senter av kjoksen. Kjoksen ble plassert i senteret til rotasjonsbordet og sentreres ved at det går en senterkloss (se figur 5.3b) fra bordet og inn i kjoksen.

5.1 Måleur

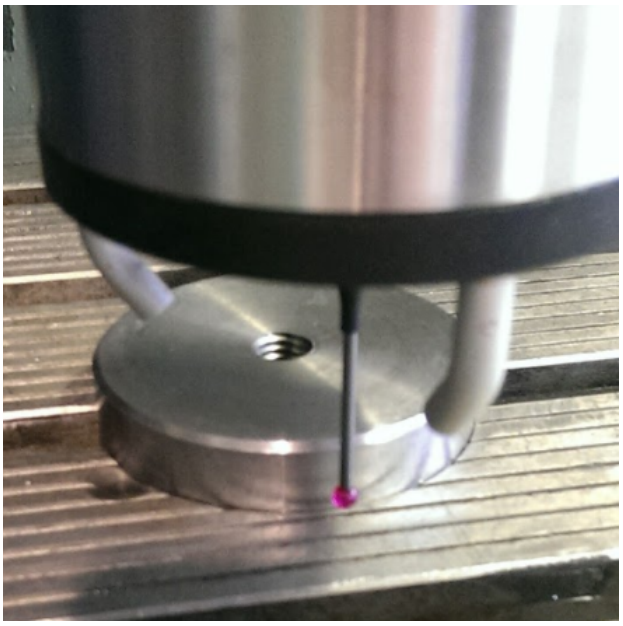
For å være sikker på at det var maskinnullpunktet som var definert feil, og ikke noe galt med innstillingene i postprosessoren, ble et måleur montert på spindelen og plassert på en dreid sylinder spent fast i kjoksen, som igjen var plassert i senter av rotasjonsbordet. Måleuret ble så plassert inntil sylinderen og nullstilt. Deretter ble bordet rotert en hel omdreining om C-aksen, samtidig som utslaget på måleuret ble dokumentert. Som figur 5.2 illustrerer var utslaget på måleuret 30 hundredels millimeter, fra en minimumsverdi på -3 hundredels millimeter til en maksverdi på 27 hundredels millimeter.

Den motsatte målingen ble så utført, og som figur 5.3a illustrerer ble måleuret montert på kjoksen og plassert inntil verktøyet før det ble nullstilt. Bordet ble rotert en omdreining om C-aksen, og utslagene viste -20 til 10 hundredels millimeter. Det var derfor nødvendig å måle opp et nytt maskinnullpunkt.



Figur 5.2: Måleurverdier

(a) Oppspenning av måleuret



(b) Måleproben inntil senterklossen

Figur 5.3

5.2 Måleprobe

Det definerte maskinnullpunktet stammer fra en kalibrering utført av Vegard Brøtan ved Sintef. I brukermanualen (figur 5.1) til DMU 50 eVolution står nullpunktverdiene definert av maskinfabrikanten da maskinen var ny. Ved siden av disse verdiene er det notert nye håndskrevet nullpunktverdier, som stammer fra en tidligere kalibrering. De tre tidligere definerte nullpunktene er gitt i tabell 5.1 i millimeter fra referansepunktet i X-og Y-verdier.

Tabell 5.1: Måleverdier

	Maskinmanual	Håndskrevet	Brøtan
X	-250.683	-250.557	-250.664
Y	-155.029	-155.073	-155.024

En måleprobe er en elektronisk kantsøker som måler verdiene for plasseringen av emnet i forhold til maskinnullpunktet. Målingene utføres ved å styre proben, slik at den berører de aktuelle sidene på arbeidsstykket. Verdien fra det målte punktet, gitt i verdier fra referansepunktet, kan leses av styringssystemet til maskinen. Det ble først gjort et forsøk på å plassere måleproben ned i hullet der den selvsentrerende klossen til kjoksen plasseres, slik at senteret av senterhullet kunne måles, men måleproben kan ikke kjøres lavere enn Z+8mm over bordet. Senterklossen fra kjoksen ble dermed plassert i hullet for å ta mål med måleproben. Måleproben ble som figur 5.3b illustrerer styrt slik at den berørte senterklossen og målte positiv og negativ Y-verdi med X=0, deretter ble positiv og negativ X verdi målt med Y=0. Dersom den positive verdien

fra måleproben i en akse blir addert med den negative og resultatet er lik null, vil nullpunktet være korrekt. Selv om senterklossen nærmest må hamres inn i hullet for å plasseres, kan man føle litt slakk i klossen når den er plassert i hullet, det ble derfor gjort 3 forskjellige målinger med de 3 nullpunktene fra tabell 5.1 som utgangspunkt. Differansen av de målte verdiene i X og Y retning ble delt på to og lagt til den gamle nullpunktverdien. Det nye nullpunktet ble regnet ut fra gjennomsnittet av de 3 verdiene og er gitt i tabell 5.2. Det ble gjort en ny måling av

Tabell 5.2: Nye måleverdier

	Maskinmannualen	Brøtan	Håndskrevet	Gjennomsnitt
X	-250.661	-250.665	-250.662	-250.663
Y	-155.056	-155.071	-155.053	-155.060

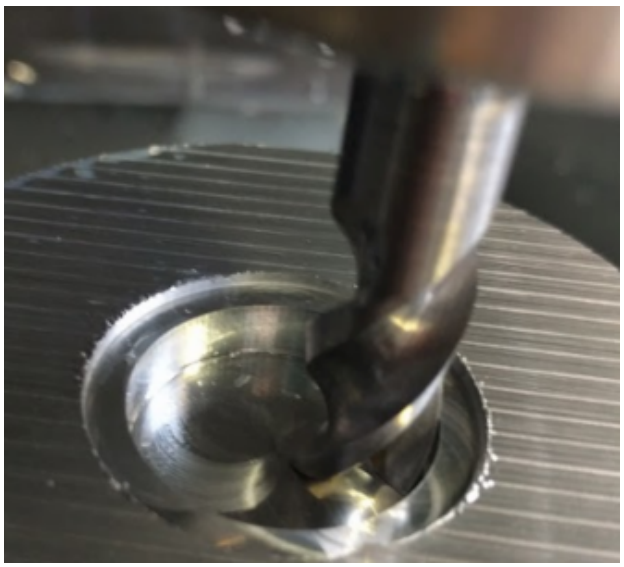
senterklossen med det nye nullpunktet som referanse og det ga et nullpunkt på X=-250.660 og Y= -155.059 som er 0.00245mm avvik i X retning og 0.00095 i Y retning som er godkjent avvik. Det nye nullpunktet ble dermed satt til X=-250.660 mm og Y= -155.059 mm iforhold til DMU 50 eVolution sitt referansepunkt.

Halvkuleprogrammet ble maskinert på nytt, denne gang med programnullpunktverdier regnet ut fra det nye maskinnullpunktet. Det var en tydelig forbedring, men det var fortsatt ikke helt korrekt. Sannsynligvis var det fordi det var litt slakk i senterklossen, men det kan også skyldes at senter av leddene mellom arbeidsbordet, senterklossen, kjoksen og arbeidsstykket ikke er helt kolineære, slik at det blir et lite avvik i maskinnullpunktet og toppen av arbeidsstykket.

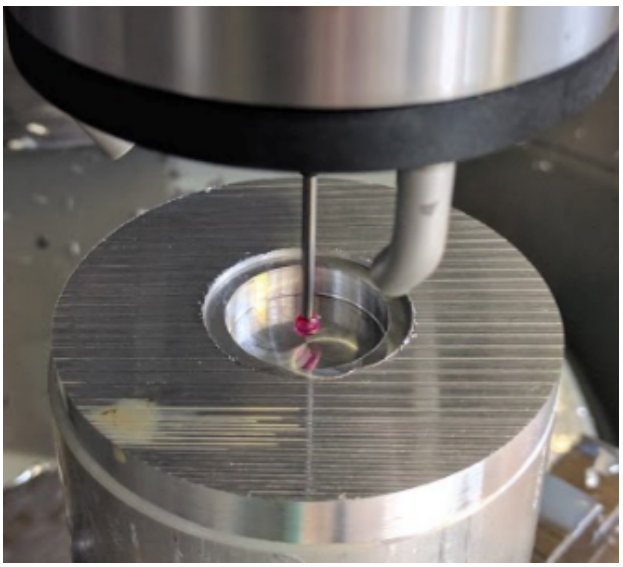
Nullpunkt i toppen av et arbeidsstykke montert i kjoksen

En ny løsning for å definere maskinnullpunktet var å feste en ny aksling inn i kjoksen, for så å frese et Ø12mm skjærverktøy ned i akslingen med verktøyet posisjonert i X=5mm og Y=0mm med en spindelhastighet på 1500rpm, som vist i figur 5.4a. Uten å bevege sleidene ble arbeidsbordet rotert, slik at det ble frest ut et hull med radius på 11mm med senter i X-og Y=0. Det neste var da å kjøre et probeprogram for å finne avstanden fra senteret av hullet til det definerte maskinnullpunktet, som vist i figur 5.4b. Hullet hadde senter i X=-250.675mm og Y=-155.023mm fra referansepunktet, som gir et avvik fra maskinnullpunktet på 0.015mm i X-retning og 0.036mm i Y-retning.

Maskinnullpunktet ble definert til X=-250.675mm og Y=-155.023mm fra referansepunktet. En ny maskineringstest ble utført med verktøybanen slik figurene i figur 5.5 illustrerer. Deretter ble måleuret montert i verktøyholderen med en ny test rundt det maskinerte arbeidsstykket, det var denne gangen ingen synlige utslag. En skyvelære ble også brukt for å måle kanten rundt arbeidsstykket i figur 5.5b, det var samme avstand rundt hele akslingen, så det kunne konkluderes med at det definerte nullpunktet var korrekt.

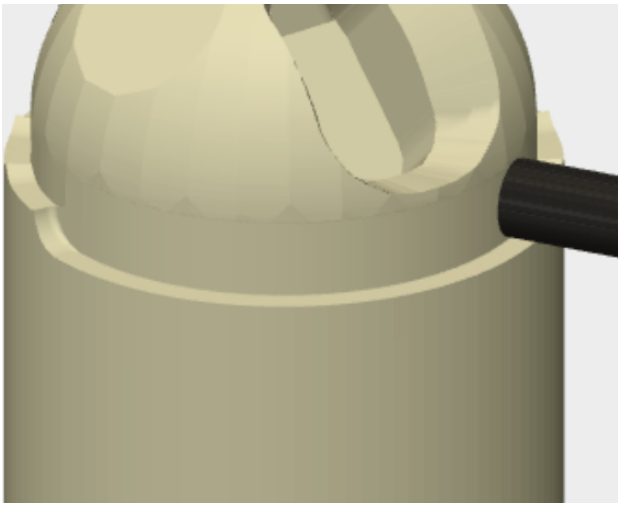


(a) Freseverktøyet i spindelen roteres samtidig som freseverktøyet manuelt styres til X=5, Y=0 og Z styres ned til et hull et dypt nok til å plassere måleproben. Bordet blir så rotert manuelt om C-aksen samtidig som spindelen spinner og det freses ut et hull i toppen av arbeidsstykket som har origo i XY-planet

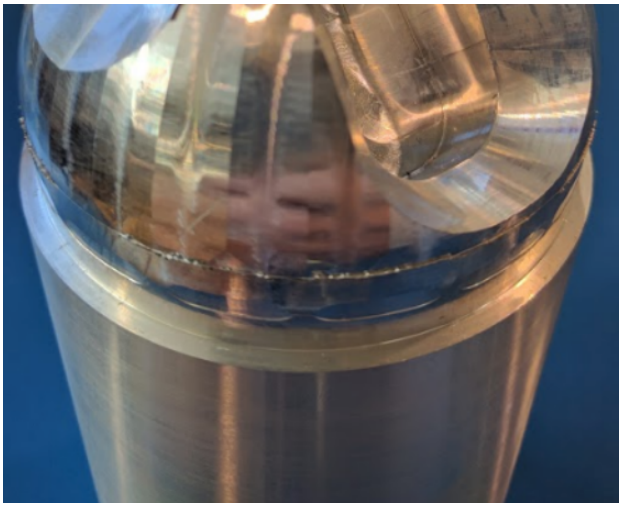


(b) Måleproben måler positiv og negativ X og Y verdi i det freste hullet, origo av det oppmålte koordinatsystemet skal være kolinært med C-aksen i rotasjonsbordet

Figur 5.4



(a) Simulering av verktøybanen



(b) Måleproben måler positiv og negativ X og Y verdi i det freste hullet, origo av det oppmålte koordinatsystemet skal være kolinært med C-aksen i rotasjonsbordet

Figur 5.5

5.3 Måle offsetverdier

Når emnet er spent fast i kjoksen måles programnullpunktet til emnet med måleproben montert i DMU 50 eVolution. Et probeprogram installert på maskinen måler de positive og negative X-og Y-verdiene målt fra maskinnullpunktet til programnullpunktet. For å kalkulere *offsetverdiene* som skal implementeres i postprosessoren i CAM, slik at programnullpunktet i maskinen og i CAM samsvarer, brukes likningene 5.1 og 5.2. Likningene ble implementert inn i et excelprogram som vist i figur 5.6, slik at verdiene som skulle implementeres inn i postprosessoren enkelt kunne kunne kalkuleres.

$$XOffsetverdi = NegativX + \frac{absoluttverdi(NegativX - positivX)}{2} \tag{5.1}$$

$$YOffsetverdi = NegativY + \frac{absoluttverdi(NegativY - positivY)}{2} \tag{5.2}$$

Offsetverdien til høyden **Z** leses fra måleresultatet direkte av maskinen. *Offsetverdiene* til X, Y og Z implementeres inn i postprosessoren via brukerdefinerte innstillinger i *Program Setings* i CAM.

	Negative Value	Positive Value	Abs(negative-positive)/2	Negative Value + Previous Ans
X	-59.736	60.325	60.0305	0.2945
Y	-60.875	59.583	60.229	-0.646
Z				281

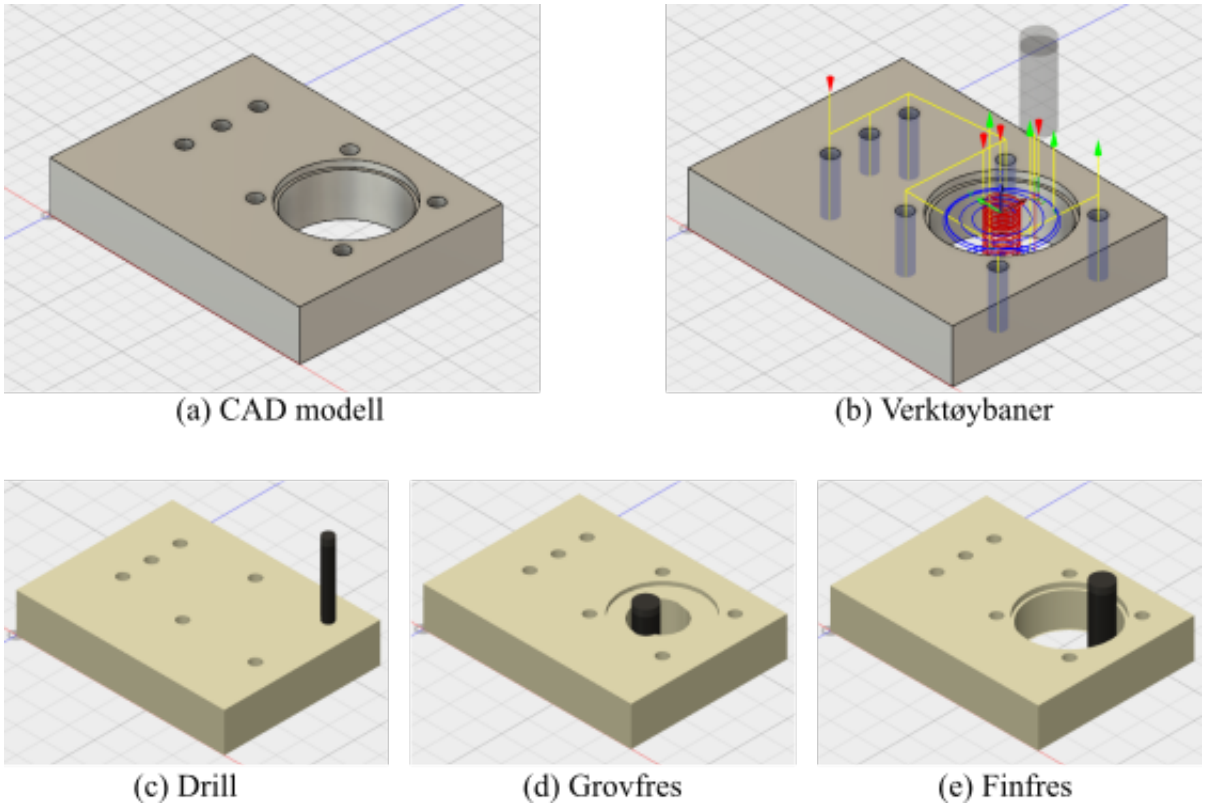
Figur 5.6: Excelprogram for å kalkulere offsetverdiene

Kapittel 6

Maskinerte modeller

6.1 Stirling motorbase

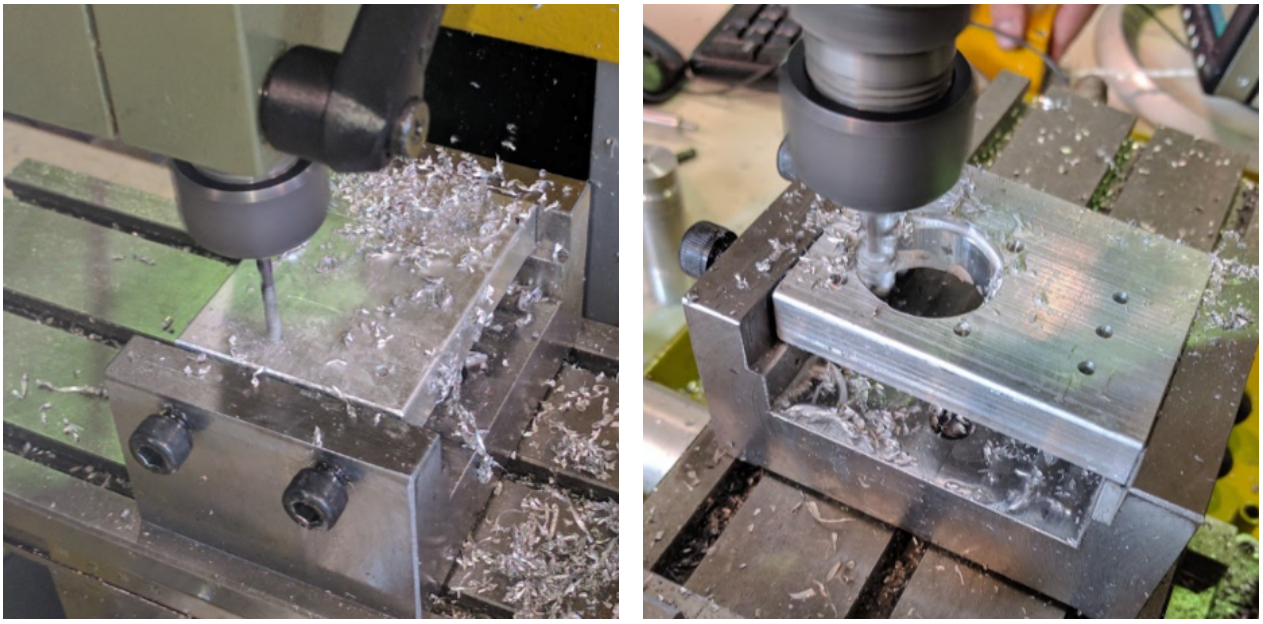
Som en del av faget TPK 4190-Produksjonsteknologi ved NTNU skal studentene modellere og maskinere en Stirling motorbase, i vedlegg C finnes en manual for hvordan motorbasen ble modellert i Fusion 360 CAD, og hvordan verktøybanene ble definert i CAM. Figurene i figur 6.1 illustrerer hvordan CAD modellen ser ut, samt verktøybanene og en simulering av maskineringen. Figur 6.2a viser drilloperasjonen i *Proxxon-FF 500CNC* og figur 6.2b viser freseoperasjonen.



(a) CAD modell (b) Verktøybaner

(c) Drill (d) Grovfres (e) Finfres

Figur 6.1: Simulering av maskineringen av hullet i motorbasen



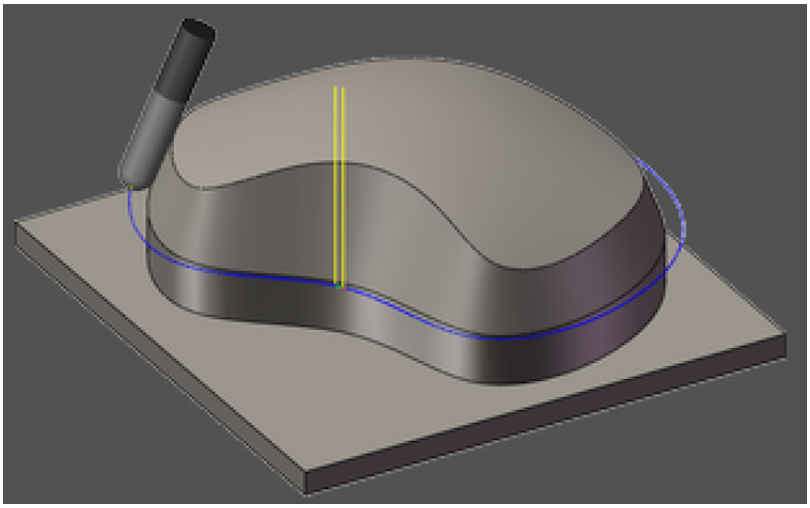
(a) Drilling av hullene i motorbasen (b) Fresing av hullet i motorbasen

Figur 6.2

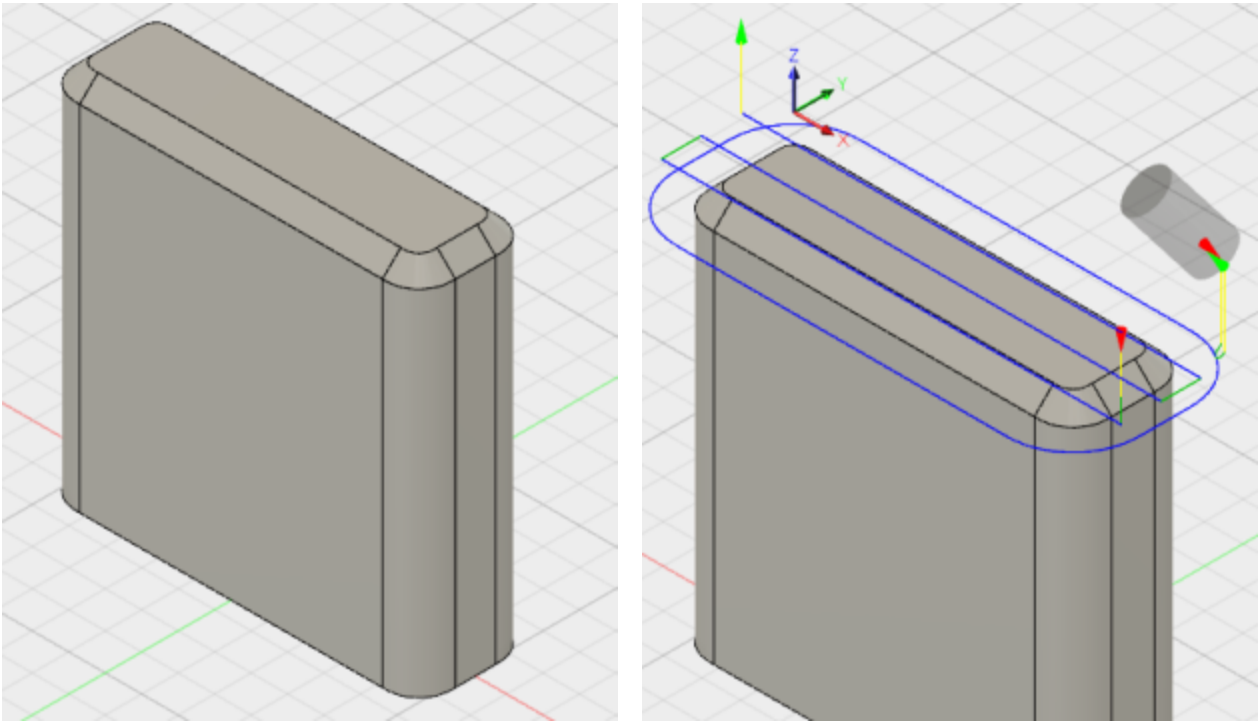
6.2 Plastmodell

Den første modellen som ble maskinert med simultan fem-akse var en modell postprosessert fra Fusion 360 med postprosessoren *APT.CPS* og så via Sørby sin frittstående postprosessor. Verktøyet skulle først plane arbeidsstykket med 3-aksefunksjonen *2D-Face*, så bruke simultan fem-aksefunksjonen *Swarf*. Som figur 6.3 illustrerer, bruker *Swarf* siden av verktøyet til å maskinere kanter på et arbeidsstykke.

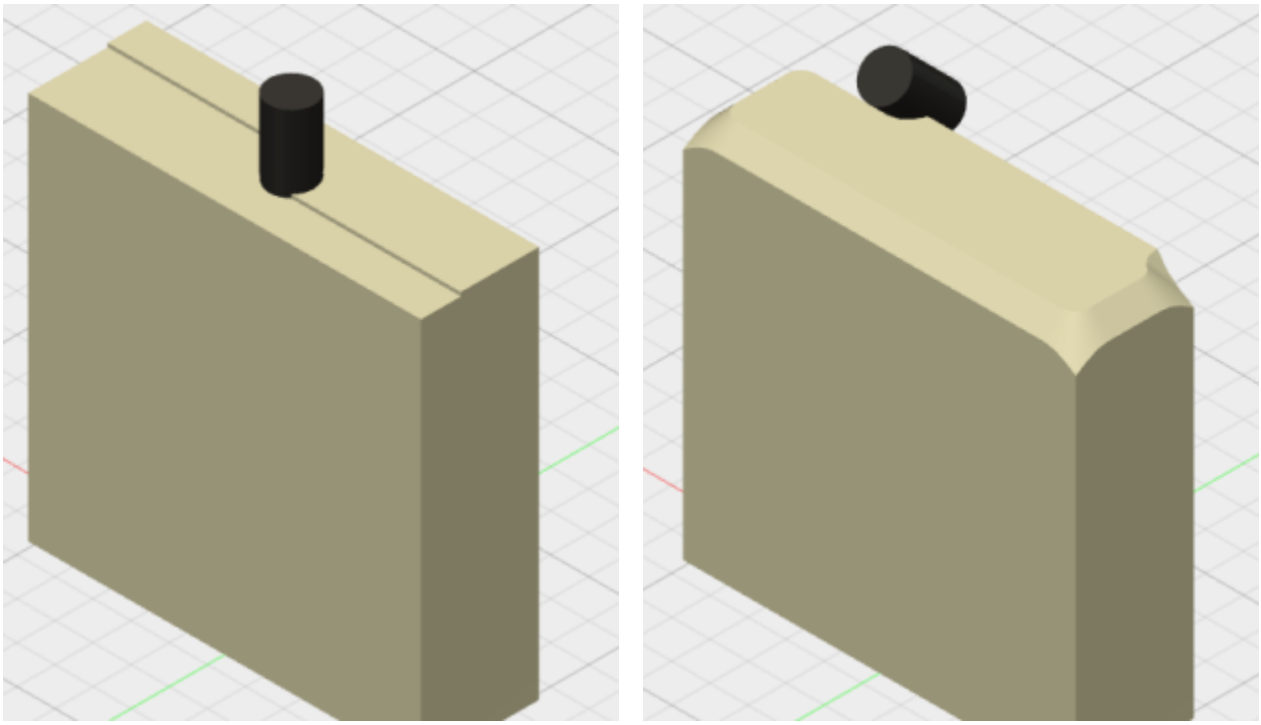
En kloss ble modellert der verktøyet skulle maskinere kantene på klossen slik figur 6.4 illustrerer. En plastkloss med lik geometri ble spent fast i Deckel Maho DMU 50 eVolution, og som figur 6.5 viser, ble klossen tilfrestillende maskinert.



Figur 6.3: Funksjonen Swarf i Fusion 360

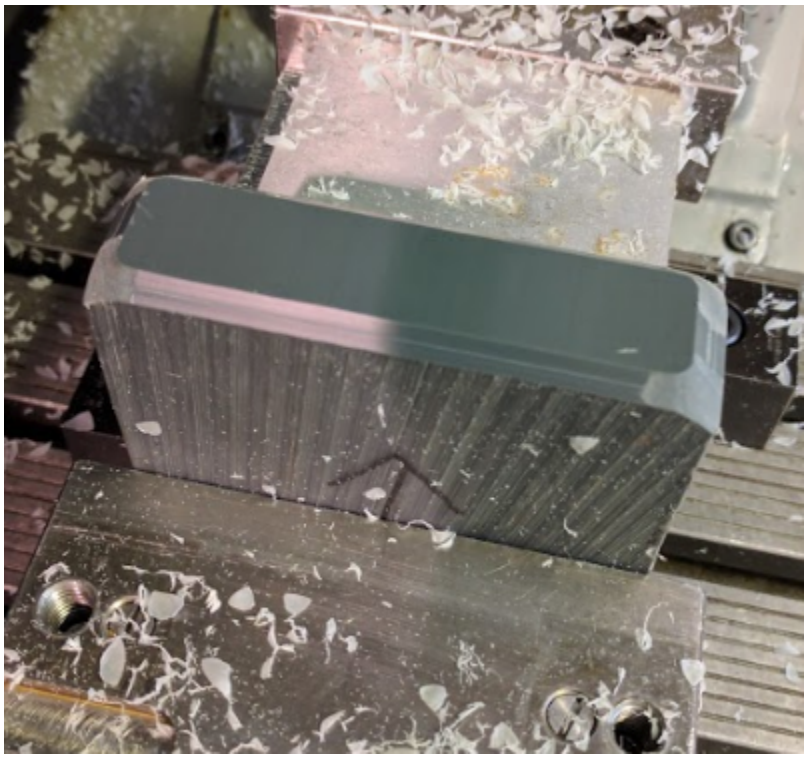


(a) (b)



(c) (d)

Figur 6.4: CAD-modell og simulering av verktøybanene med funksjonene *Face* og *Swarf*



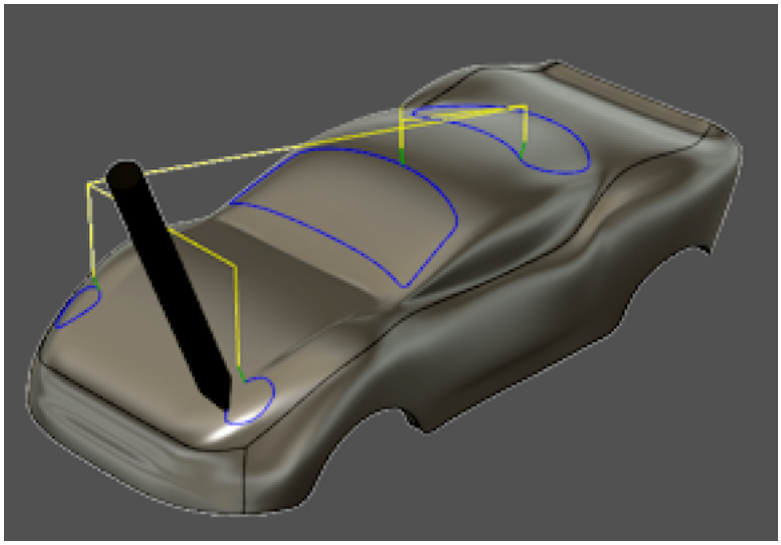
Figur 6.5: Første simultan 5-aksemaskinering

6.3 Halvkule

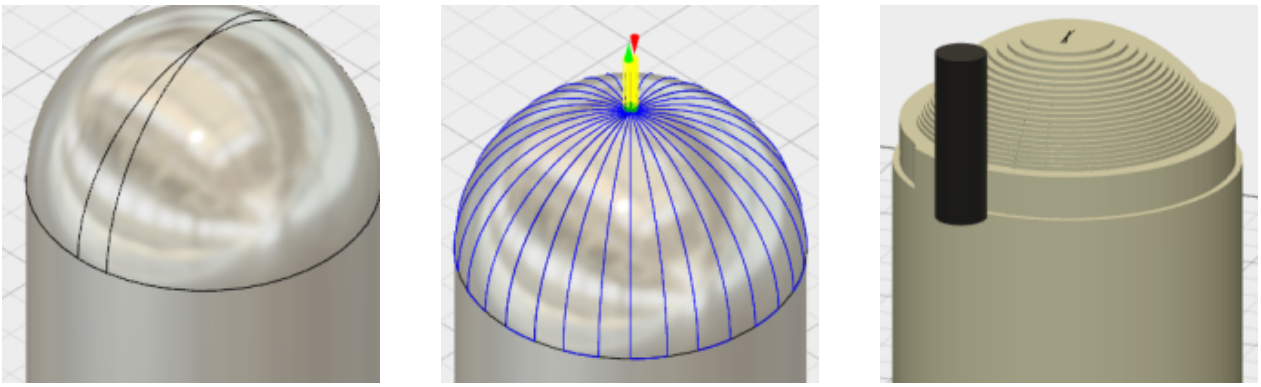
En manual for å modellere halvkulen i CAD og maskinere verktøybanene i CAM i Autodesk Fusion 360 finnes i vedlegg. En halvkule ble valgt slik at simultan femakse-funksjonen *multi axis contour* kunne prøves. Fusion 360 har per dags dato kun to funksjoner for maskinering med simultan fem-akse. Den ene er *Swarf*-funksjonen nevt i 6.2, den andre er *Multi-Axis Contour* som brukes til å maskinere langs en gitt kurve med enden av skjærverktøyet, slik figur 6.6 illustrerer. For å bruke denne funksjonen til å maskinere en hel overflate må det tenkes litt kreativt, og som forklart i manualen ble det brukt en kombinasjon av funksjonalitetene *Split Face* og *Circular Pattern* for å maskinere overflaten til halvkulen. Først ble et plan tvers gjennom halvkulen brukt til å splitte overflaten ved å bruke *Split Face*-funksjonen, deretter ble et nytt plan definert 12° om Z-aksen slik at overflaten kunne splittes med det nye planet slik figur 6.7a illustrerer. For å unngå singularitet over toppen av kulen ble verktøybanen definert til å gå langs trekanten på den ene siden, for deretter å bruke *Circular Pattern* for å kopiere verktøybanen 360° minst 15 ganger om Z-aksen, slik at hele halvkulen blir maskinert som figur 6.7b illustrerer.

For å avvirke materiale ble emnet først grovfrest med 3-akse funksjonen *3D Adaptive Clearing* med 0.8mm klaring i Z-aksen. Deretter ble *Multi-Axis Contour* brukt for å maskinere en glatt og jevn overflate, og tilslutt ble igjen *Multi-Axis Contour* brukt for å sirkulere verktøyet normalt på Z-aksen slik bildesekvensen 6.7c- 6.7f illustrerer.

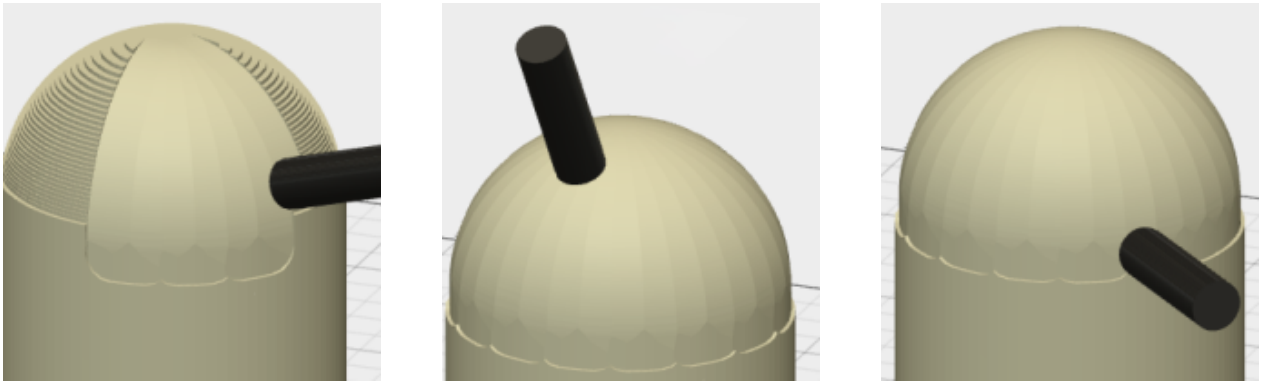
Programmet ble kjørt flere ganger i Deckel Maho DMU 50 eVolution for å teste ulike endringer i postprosessoren, og for å verifisere at nullpunktet var definert korrekt, samt ulike endringer i CAM-innstillingene som toleranseverdier. Figur 6.8a og 6.8b illustrerer utklipp fra en video av en testkjøring. Figur 6.8c illustrerer den endelige modellen med nullpunktet korrekt definert.



Figur 6.6: Multi-Axis Contour funksjonen(13)



(a) (b) (c)



(d) (e) (f)

Figur 6.7: Definering av verktøybaner og simulering av maskineringen av en halvkule

(a) (b) (c)

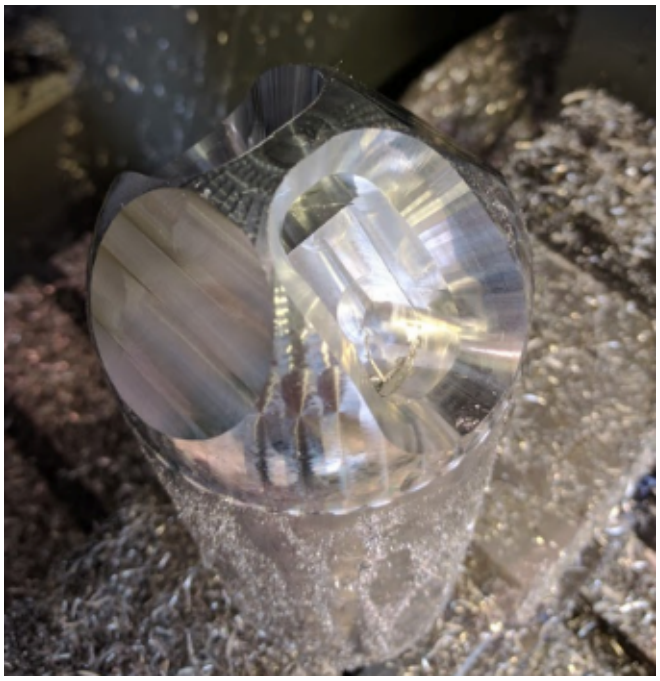
Figur 6.8: Fysisk maskinering av halvkulen (a) og (b), og et endelig resultat i (c)

6.4 Halvkule med forskjellige 5-aksefunksjoner

For å bygge videre på halvkulen ble det lagt til ulike 5-aksefunksjoner. Manualen i vedlegg E forklarer hvordan Fusion 360 CAD/CAM kan brukes til å modellere modellen og programmere verktøybanene. Først ble halvkulen i avsnittet over maskinert, så ble (3+2)-aksemaskinering brukt til å avvirke materiale i hullene ved å definere et verktøyplan som arbeidsbordet roterer til, for så å bruke 3-aksemaskinering på den gjeldende orienteringen til arbeidsstykket.

Deretter ble simultan fem-akse funksjonen Swarf brukt til å maskinere en kant rundt hullene.

Figur 6.9: Fysisk fem-aksemaskinering med funksjonen *Swarf*

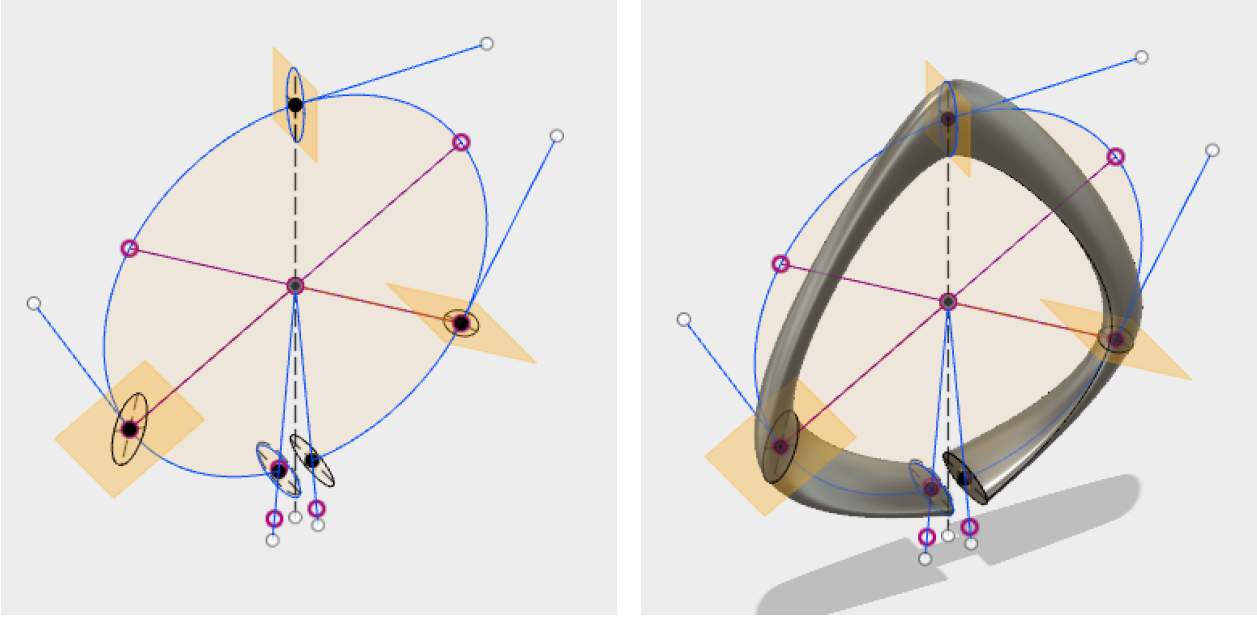


Figur 6.10: Resultat av maskinering

6.5 Mobius ring

En Mobius ring blir tatt for å være en krevende modell å frese. Formen av ringen er slik at utsiden av ringen blir innsiden for å så bli utsiden igjen i en loop, altså at ringen snurrer 180°. Selv med en høyteknologisk 5-aksemaskin er det en krevende del å produsere(22).

Ringen ble modellert i Fusion 360 CAD ved å bruke funksjonen *Loft* til å sette sammen ellipseskissene slik at de snurrer 180° om senteret slik skissene i figur 6.11 illustrerer.



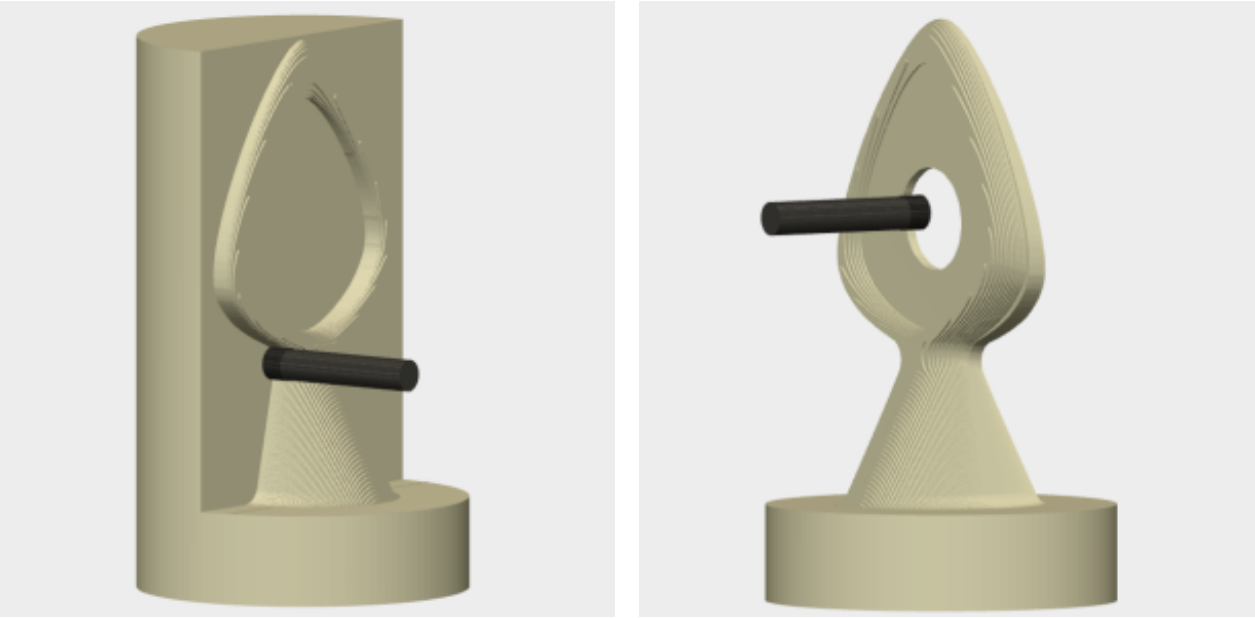
(a) (b)

Figur 6.11: Ringen ble først designet i Fusion 360 CAD ved å bruke *Loft*-funksjonen til å sette sammen skissene

Først ble ringen grovfrest som figurene 6.12 og 6.13 illusterer. Deretter ble finfres med funksjonen *3D-Contour* brukt til å glatte ut ringen. Selve oppsettet minner om grovfres operasjonen, der bordet først roterer til det definerte verktøyplanet og verktøyet beveger seg om 3-akser mens bordet står stille. Når den ene siden er ferdig, roterer bordet 180° og maskinerer den andre siden på samme måte. Det ble brukt et skjærverktøy av typen Ø10mm radiefres for å maskinere finfres-operasjonen med en klaring på 0.6mm, slik at det var igjen litt ekstra materiale til testing av simultan 5-aksefunksjoner.

På enden av modellen var det ikke nok støttestruktur i arbeidsstykket slik at det oppsto vibrasjoner mellom verktøyet og arbeidsstykket, kjent som sperring, som førte til en hakkete overflate.

Som forklart i avsnittet med halvkulen inneholder ikke Fusion 360 avansert simultan 5-aksefunksjonalitet, og verktøybanen må derfor programmeres manuelt. På Mobius-ringen ble overflaten splittet med *Split-Face*-funksjonen, slik at overflaten til ringen er delt opp med



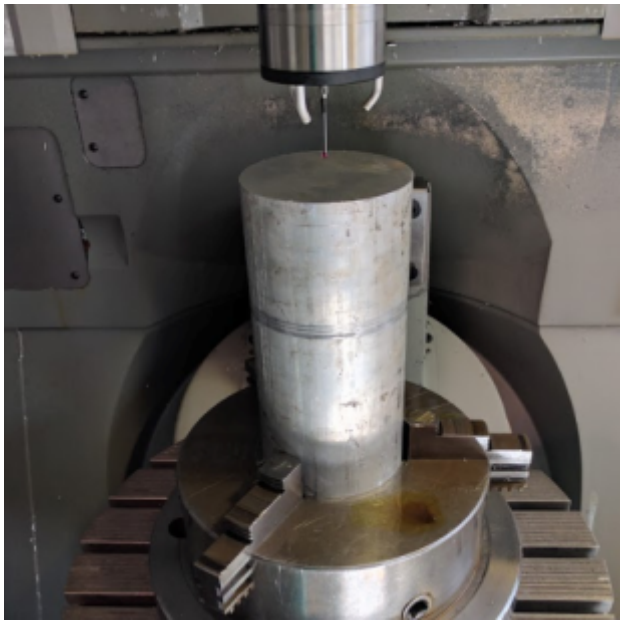
(a) (b)

Figur 6.12: For å grovfrese ble funksjonen *3D-Adaptive Clearing* brukt, med definert verktøyplan. Figuren illustrerer en simulering av grovfres-operasjonen

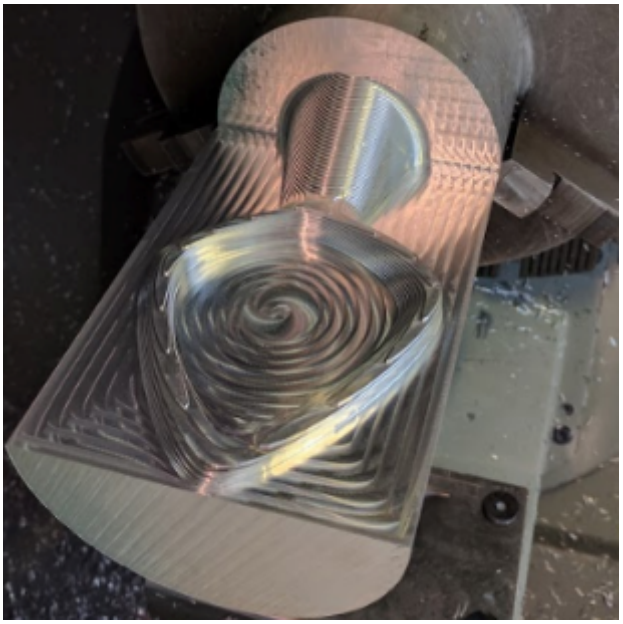
små mellomrom slik at hver del kan brukes som verktøybane rundt ringen. Figur 6.14 illustrerer prosessen med hvordan *Split-Face* og *Multi-Axis Contour* har blitt brukt til å definere 20 verktøybaner rundt modellen med 0.4mm mellomrom nedover. Ettersom dette er en svært tidkrevende prosess, samt at det tar lang tid for maskinoperatøren ansatt på verkstedet å maskinere, ble det ikke maskinert mer enn 20 baner nedover.

For å redusere sperring ble den samme finfres *3D-Contour*operasjonen maskinert på nytt, denne gangen med en Ø2mm radiefres, med en klaring på 0.5mm. Det ble mindre sperring, men verktøybanene ble tydeligere, og en annen Fusion 360 funksjon som for eksempel *3D-Pencil* hadde gitt et finere resultat, med tettere verktøybaner, men tar mye lenger tid å maskinere.

For å maskinere rundt ringen i en operasjon kreves det nøye definert verktøy og verktøybaner for å unngå kollisjon. Som en siste kontroll av postprosessoren ble verktøyet programmert til å maskinere en runde rundt ringen uten klaring. Figur 6.15 illustrerer simulering og maskinering av operasjonen. Verktøyet beveget seg som det skulle, uten kollisjon og resultatet er illustrert i figurene 6.16 og 6.17



(a) Oppmåling av råemnet apent fast i kjoksen



(b) Grovfres

(c) Grovfres motsatt side

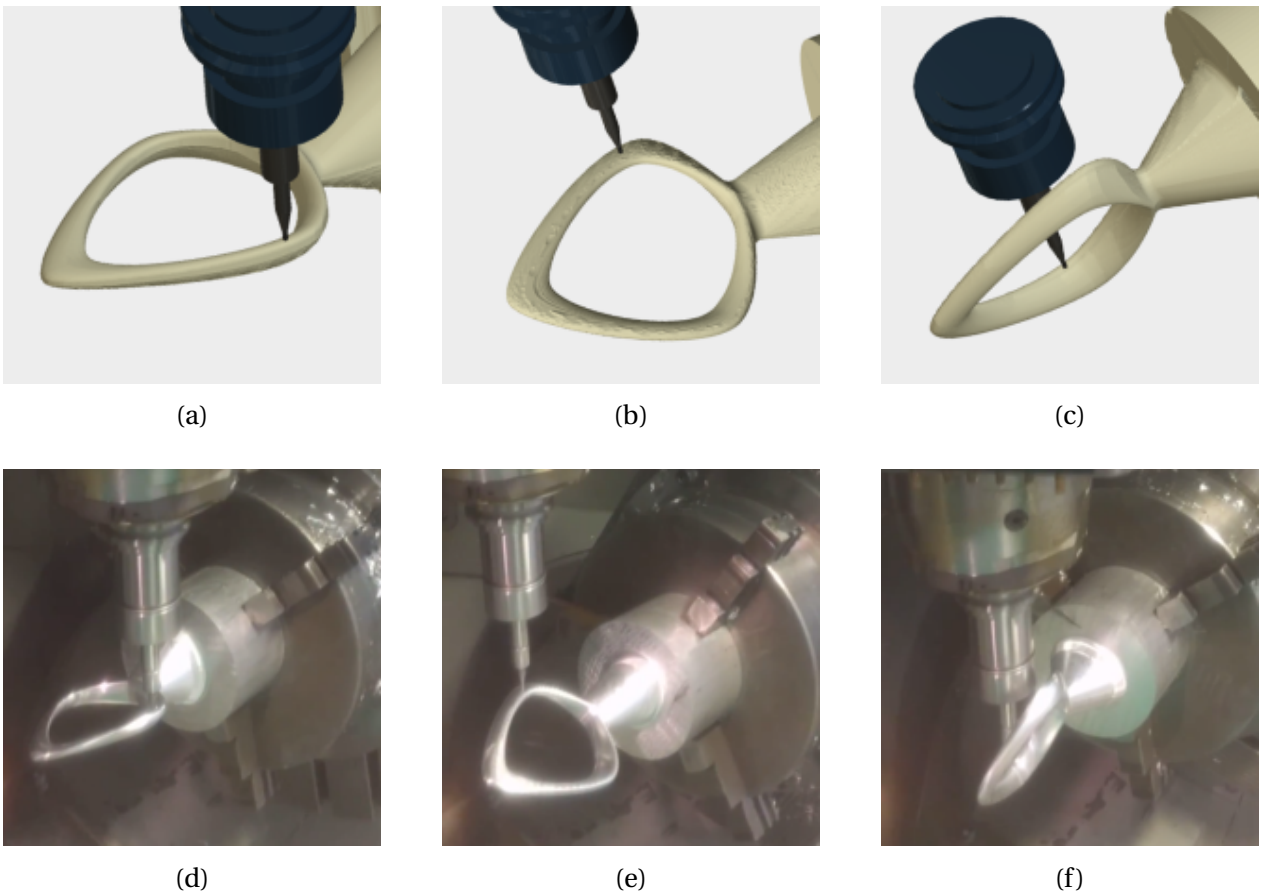
(d) Resultat av grovfres

Figur 6.13: Resultat av grovfresing

(a) (b) (c)

(d) (e) (f)

Figur 6.14: Maskinering med simultan 5-akse rundt ringen. Figur (f) illustrerer resultatet, det ble maskinert med 0.4mm klaring, slik at det var igjen litt materiale til en eventuell senere maskinering. Det kan observeres noen hakk på tuppen, disse skyldes sperring fra finfres maskineringen.



(a) (b) (c)

(d) (e) (f)

Figur 6.15: Fysisk maskinering rundt ringen

(a) (b)

Figur 6.16: Resultat av maskinering rundt ringen

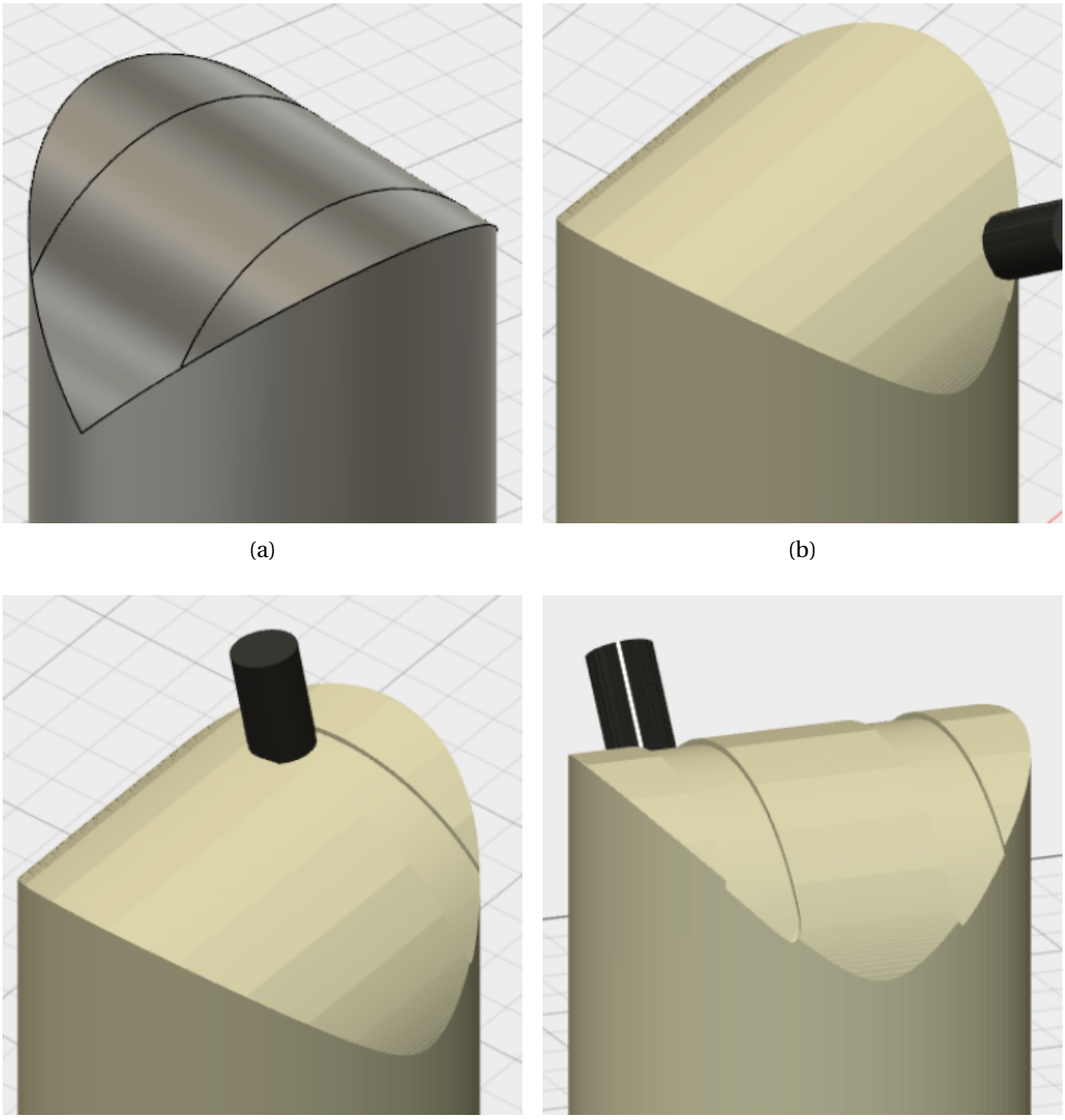
Figur 6.17: Endelig resultat av mobius-ringen

6.6 Singularitet-og Lineariseringskontroll

For å verifisere at singularitetskontrollen fungerte ble det modellert en CAD-modell med en halvsylinder slik figur 6.18a illustrerer. To verktøybaner ble definert med funksjonen *Multi-Axis Contour* der verktøyet stod normalt på verktøybanen som fulgte halvsylinderen 20mm ut fra hver sin side av sentrum. Målet med modellen er at den ene siden skal maskineres med singularitetskontrollen i postprosessoren av, og den andre med kontrollen på. En singularitet oppstår som forklart i kapittel 2.5 når vinkel B=0° og bordet må foreta en rask rotasjon på 180°for å komme til neste NC-punkt. Dersom kontrollen virker vil den ene siden bli maskinert med verktøyet som korrigerer for 180° rotasjonen til arbeidsbordet ved å følge bevegelsen til bordet, og den maskinerte verktøybanen se lik ut som simuleringen i figur 6.18. Den andre siden som maskineres med kontollen skrudd av vil ha verktøyet i ro når bordet roterer, slik at verktøyet vil kollidere med arbeidsstykket.

Singularitetskontrollen ble postprosessert på flere halvsylindere med ulik geometri og ulike programnullpunkt definert om Z-aksen samt verktøybaner med start og sluttpunkt på motsatt side. Ved å analysere NC-koden så det ut til at postprosessoren håndterte alle singulariteter der bordet roterte fra 0°-180°, 90°-270°og tilbake med motsatt rotasjon.

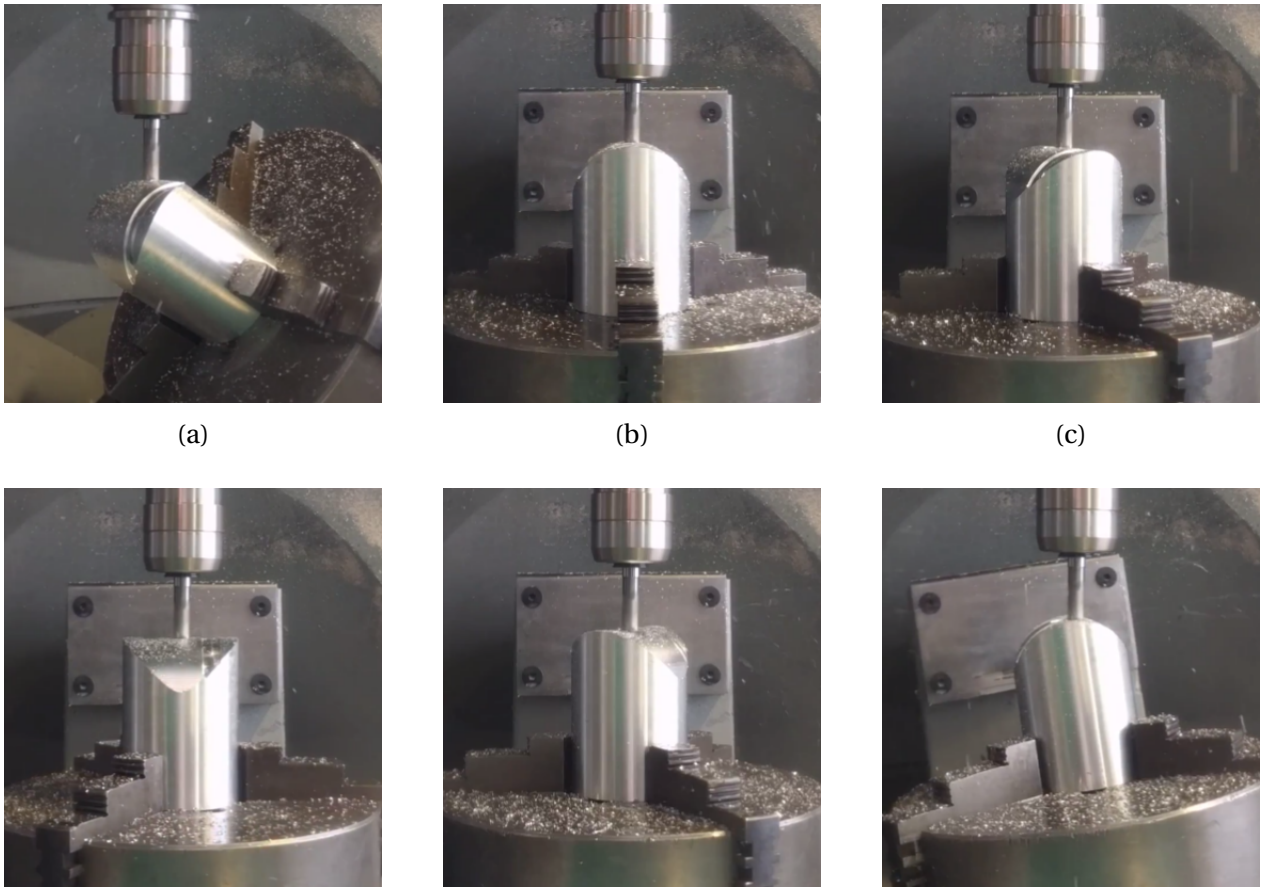
Verktøybanene i figur 6.18 ble postprosessert med singularitetskontrollen skrudd av og maskinert slik figur 6.19, og med kontrollen på i figur 6.20. Figur 6.21 viser den maskinerte strukturen.



(a) (b)

(c) (d)

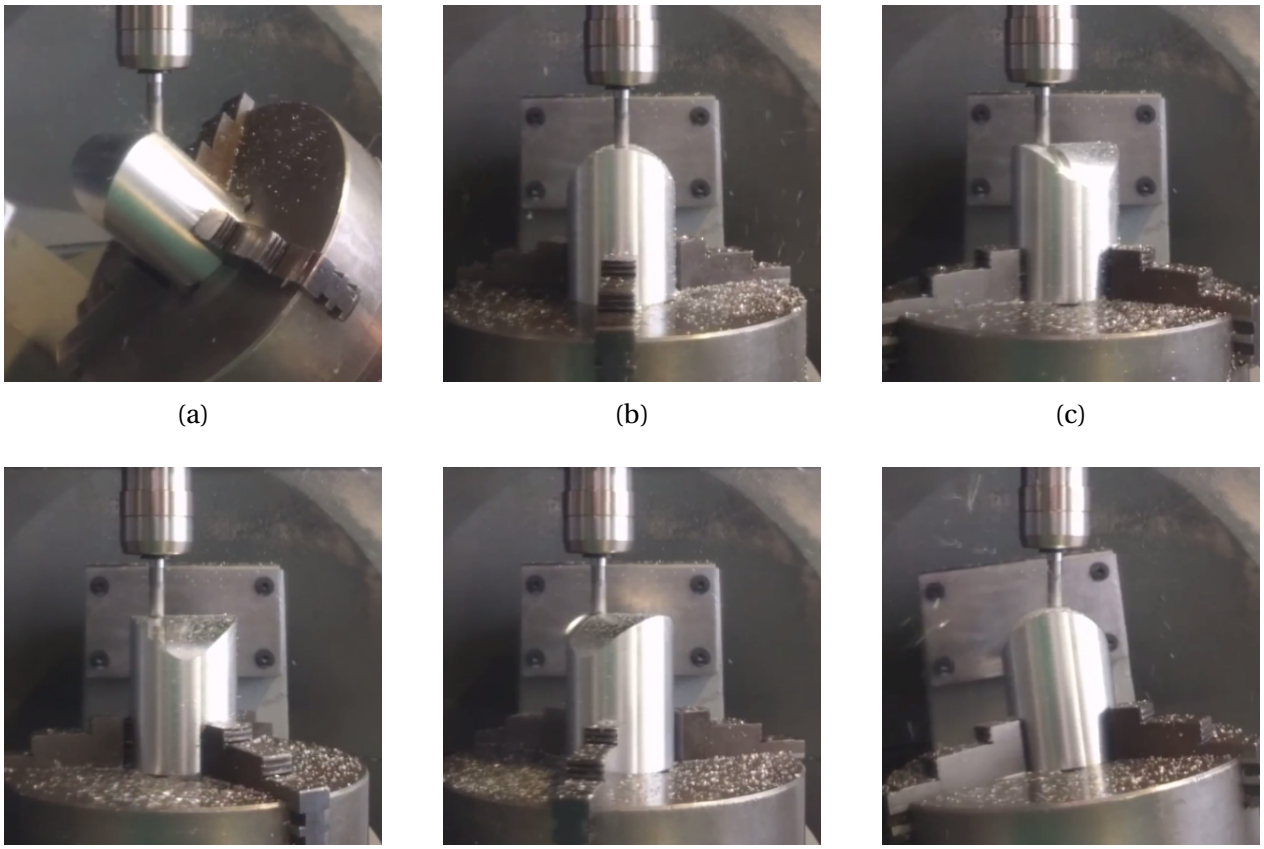
Figur 6.18: Simulering av verktøybaner for singularitetskontrollen



(a) (b) (c)

(d) (e) (f)

Figur 6.19: Figurene illustrerer maskinering der singularitetskontrollen er skrudd av. I bilde (b) er B=0° og det oppstår en singularitet slik at bordet må rotere 180° for å maskinere videre (bilde (f)). Bildesekvensen (c)-(e) illustrerer hvordan freseverktøyet står i ro om X-,Y-og Z-aksene, samtidig som arbeidsbordet roterer 180°. Dette fører til at verktøyet kolliderer med arbeidsstykket og det blir en uønsket geometri. CNC-koden for maskineringen finnes i vedlegg F.2



(a) (b) (c)

(d) (e) (f)

Figur 6.20: Simuleringen illustrerer hvordan verktøyet beveger seg når singularitetskontrollen er skrudd på. Singulariteten oppstår i bilde (b) der B=0°, postprosessoren har da kalkulert nye NC-punkt som flytter freseverktøyet med rotasjonsbordet i X-,Y-og Z-aksen, slik at verktøyet ikke kolliderer med arbeidsstykket. CNC-koden for maskineringen finnes i vedlegg F.1

Figur 6.21: Resultat av maskineringen illustrert i figurene 6.19 og 6.20. Til høyere er singularitetskontrollen på, slik at verktøyet har håndtert singulariteten uten at det har gått utover geometrien til verktøybanen. Til venstre er kontrollen av og verktøyet har kollidert med arbeidsstykket slik at det har blitt maskinert en uønsket løkke.

Kapittel 7

Konklusjon og videre arbeid

7.1 Konklusjon

Det har i dette prosjektet blitt utviklet ulike postprosessorer til å generere maskinkode fra Autodesk Fusion 360. For å avdekke eventuelle feil i de utviklede postprosessorene, har de utførte endringene i de ulike postprosessorene blitt testet på modeller av ulik geometri og ulike maskineringsmetoder fra CAM-seksjonen i Autodesk Fusion 360. Dersom det oppsto feil ble de nødvendige endringene i postprosessoren utført, slik at samtlige maskineringer til slutt ga gode resultater.

Det har blitt utviklet manualer for å modellere med Autodesk Fusion 360 CAD, samt definere verktøybaner for å maskinere de samme modellene i Fusion 360 CAM. Manualene er utviklet med hensikt å innføres som en del av undervisningen ved instituttet, samt å kunne anvendes av doktorstipendiater.

For å maskinere en modell med 3-aksemaskinering ble det gjort endringer i kildekoden til en eksisterende postprosessor utviklet av Autodesk for Fusion 360 CAM, slik at postprosessoren ble tilpasset 3-aksemaskinen Proxxon-FF 500CNC. Endringene var hovedsakelig å fjerne funksjoner i kildekoden, som førte til feilmeldinger på maskinen.

Professor Knut Sørby ved MTP, NTNU har utviklet en frittstående postprosessor som leser CL-data i form av GOTO-kommandoer, og genererer maskinkode tilpasset 5-aksemaskinen Deckel Maho 50 eVolution. Autodesk har utviklet en egen postprosessor som går under navnet *APT.CPS*. Denne postprosessoren genererer en fil med GOTO-kommandoer som kan brukes med Sørby sin frittstående postprosessor. For at den genererte filen kunne brukes med Sørby sin postprosessor ble det utført nødvendige endringer i *APT.CPS* slik at Sørby sin postprosessor kunne lese filen. Et eksempel på en endring var å erstatte kommandoen *MOVARC* til å heller skrive ut *CIRCLE* for buet verktøybane. Da alle de nødvendige endingene var utført ble den genererte maskinkoden fra Sørby sin postprosessor via *APT.CPS* testet på Deckel Maho DMU 50 eVolution. Den postprosesserte modellen var formet som en halvkule, og ble vellykket maskin-

ert ved å bruke 5 akser simultant.

Med en maskinkode som fungerte på Deckel Maho DMU 50 eVolution og med verktøybaner definert i Fusion 360, kunne det ved å sammenlikne maskinkoder utvikles en postprosessor for å generere maskinkode direkte fra Fusion 360. Som utgangspunkt ble en postprosessor utviklet av Autodesk under navnet *Heidenhain iso.cps* brukt, og kinematikken ble endret til å bruke liknigene for inverskinematikken til Deckel Maho DMU 50 eVolution, med mulighet til å definere arbeidstykkets posisjon i maskinen via *Offset-verdier*. Ved å sammenlikne maskinkoden generert av postprosessoren til Sørby kunne den genererte koden fra *Heidenhain iso.cps* analyseres uten å laste over maskinkoden på verktøymaskinen for å teste hver endring. Ved å teste flere ulike modeller ble det avduket ulike feil i postprosessoren som til slutt, basert på resultatet fra testene, virket å være feilfri.

For at postprosessoren skulle håndtere situasjoner der det oppstår en singularitet uten at verktøyet kolliderer med arbeidsstykket ble en singularitetskontroll utviklet ved å bruke forover- og inverskinematikken til Deckel Maho DMU 50 eVolution. Det ble også utviklet en linearisseringskontroll som sjekker om avviket som skyldes at rotasjonsaksene i maskinen blir for stort og eventuelt interpolerer et midtpunkt mellom de to NC-blokkene. Singularitetskontrollen ble postprosessert med verktøybaner rundt flere halvsylindere med ulik geometri og ulike programnullpunkt definert om Z-aksen samt verktøybaner med start og sluttpunkt på motsatt side. Ved å analysere CNC-koden så det ut til at postprosessoren klarte å håndtere alle singulariteter der bordet roterte fra 0°-180°, 90°-270°og tilbake med motsatt rotasjon. En testmodell ble maskinert med en vellykket håndtering av en singularitet.

Nøyaktigheten til Deckel Maho DMU 50 eVolution ble utbedret ved å kalibrere et nytt maskinnullpunkt. Nullpunktet ble verifisert ved å måle opp og sjekke at avstanden fra råemnet til en maskinert halvkule var lik rundt arbeidstykket.

7.2 Videre arbeid

Videre arbeid i dette prosjektet kan være å implementere offsetverdier i postprosessoren fra målinger med måleproben for orientering om X-og/eller Y-aksen i maskinen, slik Sørby sin frittstående postprosessor har støtte for. Det ble ikke fokusert på i dette prosjektet, da de maskinerte råemnene var sylinderformet og plassert i en selvsentrerende kjoks, slik at emnet ikke ble rotert om verken X-eller Y-aksen. Dersom det implementeres offsetverdier for orientering, kan rotasjonsmatrisene forklart i teorikapittelet brukes for å kalkulere orienteringen.

Det ble ikke implementert en funksjon som sjekker om de kalkulerte verktøybanene er innenfor arbeidsområdet til Deckel Maho DMU 50 eVolution i postprosessoren *Heidenhain iso.cps*. Dersom dette implementeres ved en senere anledning må freseverktøyet definert i Fusion 360 ha identiske mål med freseverktøyet og verktøyholderen plassert i Deckel Maho DMU 50 eVolu-

tion. Ulike freseverktøy har ulike mål og et kortere freseverktøy kan muliggjøre en verktøybane, som ville vært utenfor arbeidsområdet med et lenger freseverktøy. Postprosessoren må derfor vite målene til freseverktøyet for å kalkulere om verktøybanen er mulig.

For å unngå kollisjon med freseverktøyet og kjoksen eller bakkene som holder arbeidsstykket, kan det modelleres en realistisk CAD-modell av kjoksen i Fusion 360. Emnet kan da plasseres i kjoksen i Fusion 360 CAM, og verktøybaner kan genereres slik at freseverktøyet og verktøyholderen ikke kolliderer med kjoksen. Dersom dette utføres vil det spare mye råmateriale i det avkappede arbeidsstykket, ved at det ikke vil kappes til et ekstra langt arbeidsstykke for å være på den sikre siden, men heller kappe til et arbeidsstykke som er langt nok til å maskinere modellen uten kollisjon, men uten unødvendig ekstra råmateriale.

Det kan utvikles et program som simulerer maskineringsprosessen i en modellert verktøymaskin med de samme målene og rotasjonsaksene som Deckel Maho DMU 50 eVolution, slik at operasjoner definert i Autodesk Fusion 360 CAM kan simuleres mer nøyaktig.

Appendices

Appendix A

Digitale vedlegg

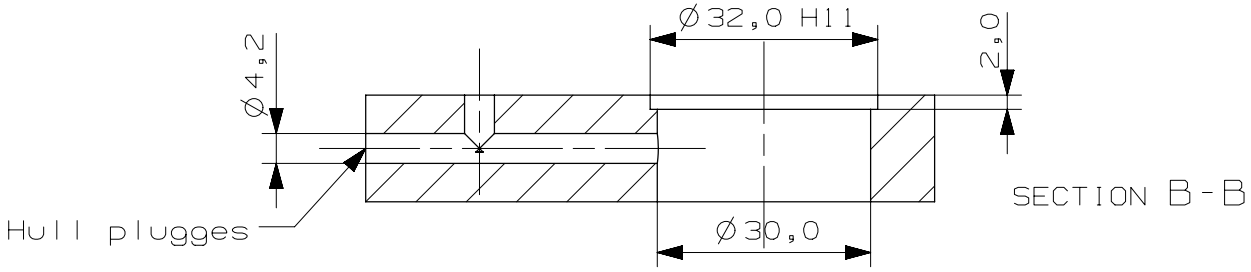
De utviklede postprosessorene i denne masteroppgaven. Postprosessorene anbefales å åpnes i teksteditoren *Notepad++* og fra menyen velge *Language->JavaScript*

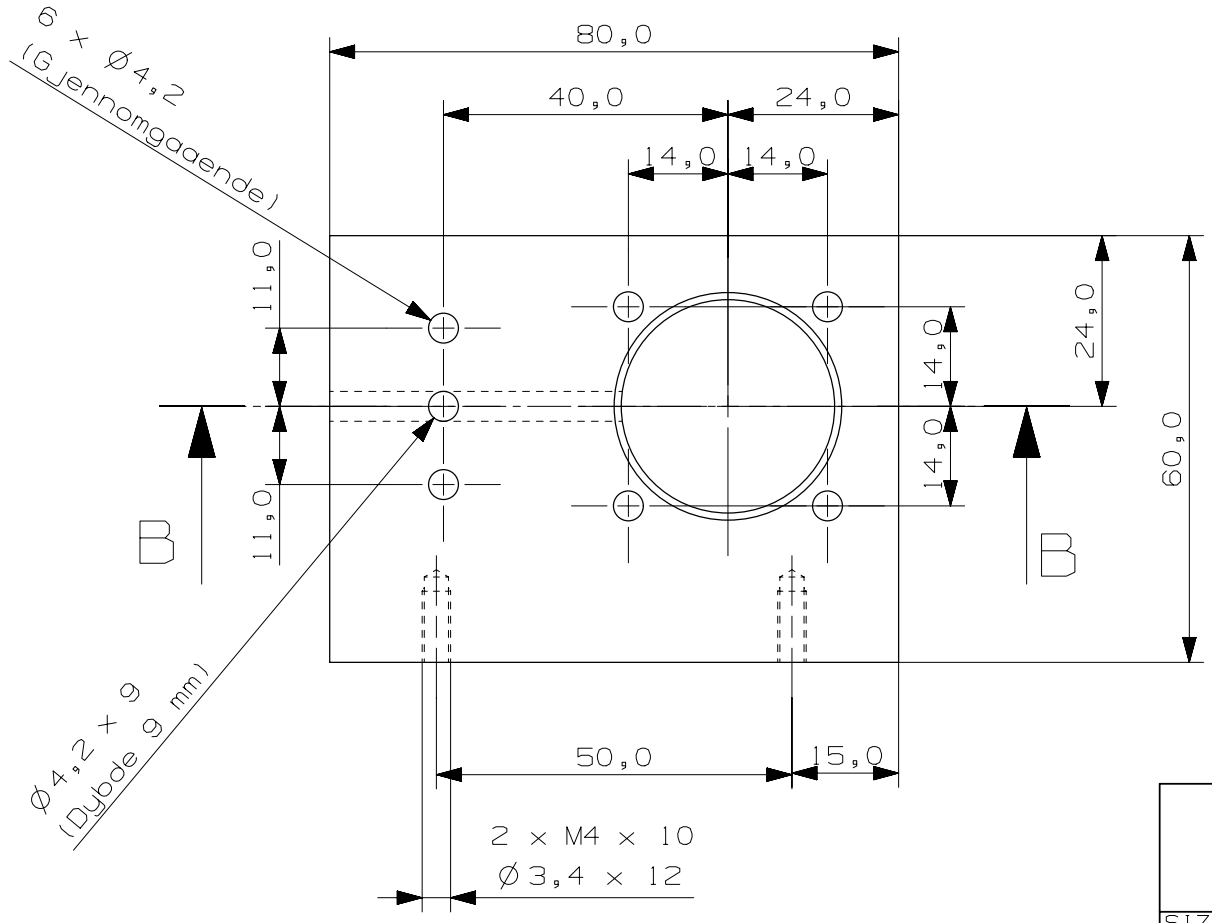
- Mach3millendret.cps: Modifisert versjon av mach3mill.cps, som kan brukes til å generere maskinkode for motorbasen til studentprosjektet i TPK4190. Koden som ble tatt ut er skjult ved å være markert som kommentarer.
- apta.cps: Modifisert versjon av *APT.CPS*, som genererer GOTO-kommandoer som kan brukes med Sørby sin frittstående postprosessor.
- AADM50.cps: Modifisert versjon av *Heidenhain iso.cps*, som bruker den programmerte funksjonen *invKin* for å kalkulere NC-data.
- AADM50sinlin.cps: Singularitet-og lineariseringskontroll ble utviklet og implementert i AADM50.cps, likningene fra inverskinematikken i funksjonen invKin ble erstattet med en ny funksjon *CLtoNC*. Det ble også utviklet en funksjon *NCtoCL*, som bruker likningene for direktekinematikken for å kalkulere CL-data fra NC-data. Andre funksjoner som ble utviklet og brukes i kontrollene er beskrevet i kapittel 4.

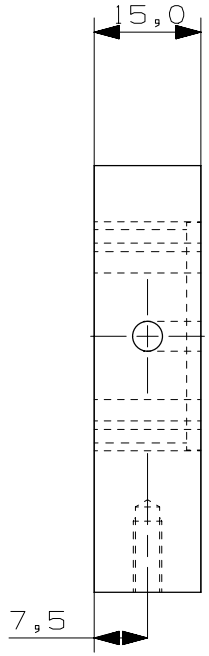
Det er også lagt ved en film av de ulike komponentene som ble maskinert i Deckel Maho DMU 50 eVolution.

Appendix B

Maskinteging motorbase







MOTORBASE

SIZE	MATERIAL:	REV
A4	ALUMINIUM	

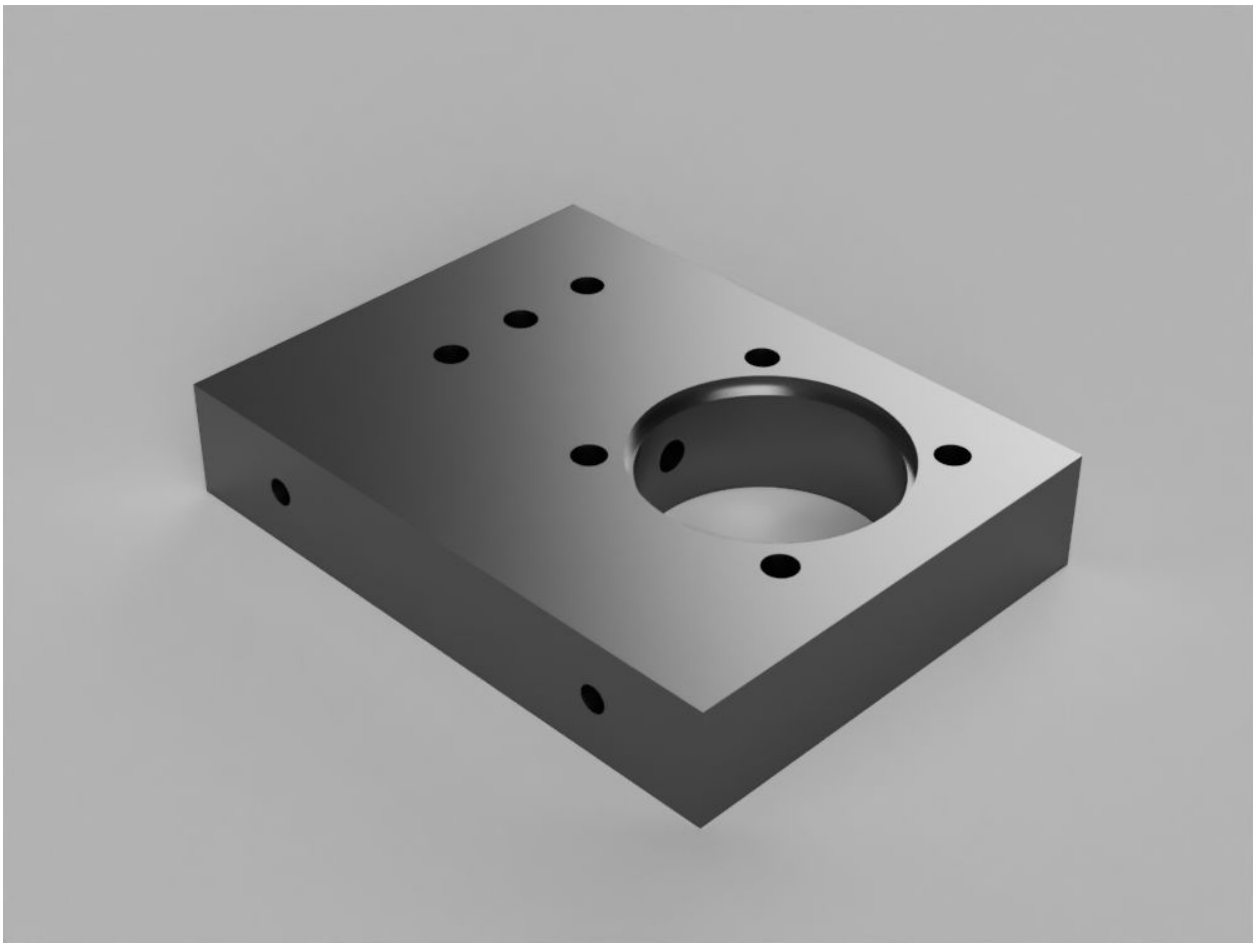
SCALE 1:1	DATE: 11.02.2016	NTNU-IPK

Appendix C

Manual for Stirling-motorbase

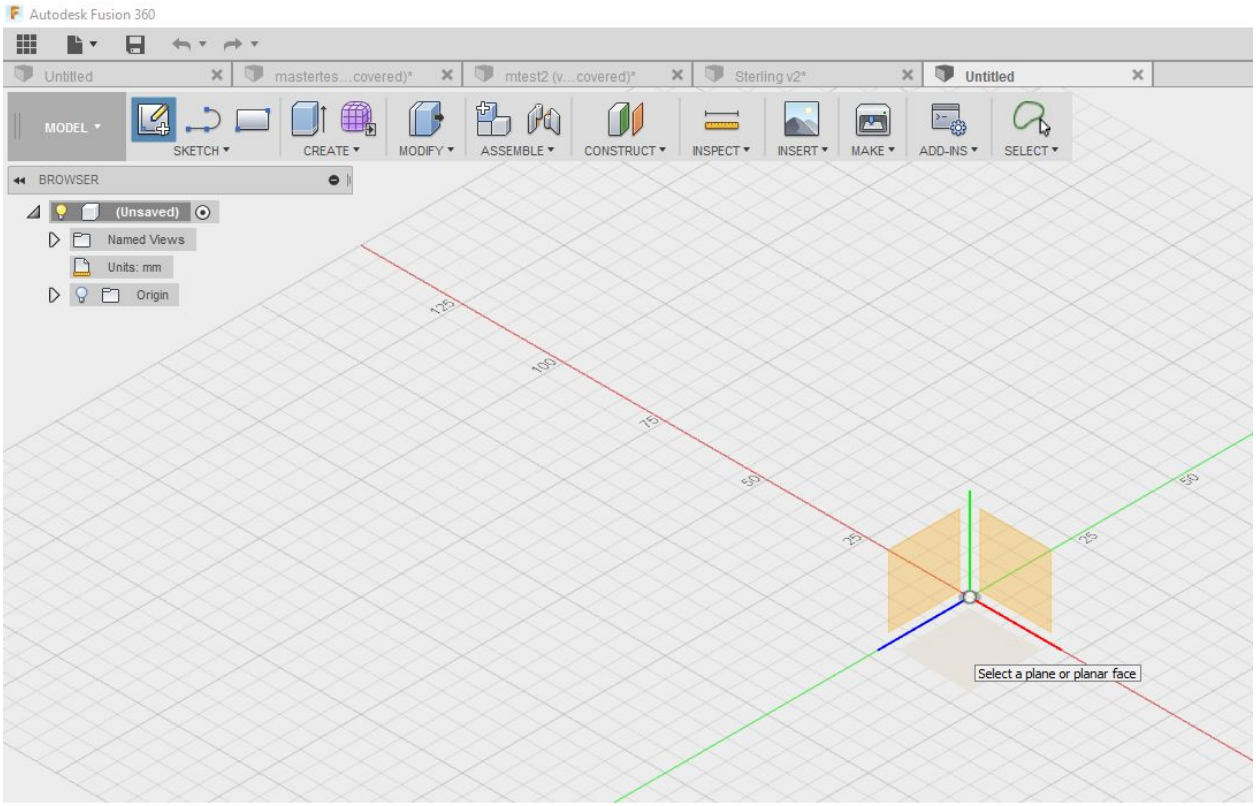
Introduction

Autodesk Fusion is a 3D CAD and CAM tool. It will connect your entire product development process in a single cloud-based platform that operates on both Mac and PC. Free educational access is available to students, sign in or create an Autodesk account to download and start running Fusion 360 on your Mac or PC.



This guide describes step-by-step the functions used to design a CAD model, as well as how to program the part using CAM (Computer Aided Manufacturing). At the end we will post process a G-Code to prepare the part for machining on a *Proxxon-FF 500CNC* machine.

Create a CAD model

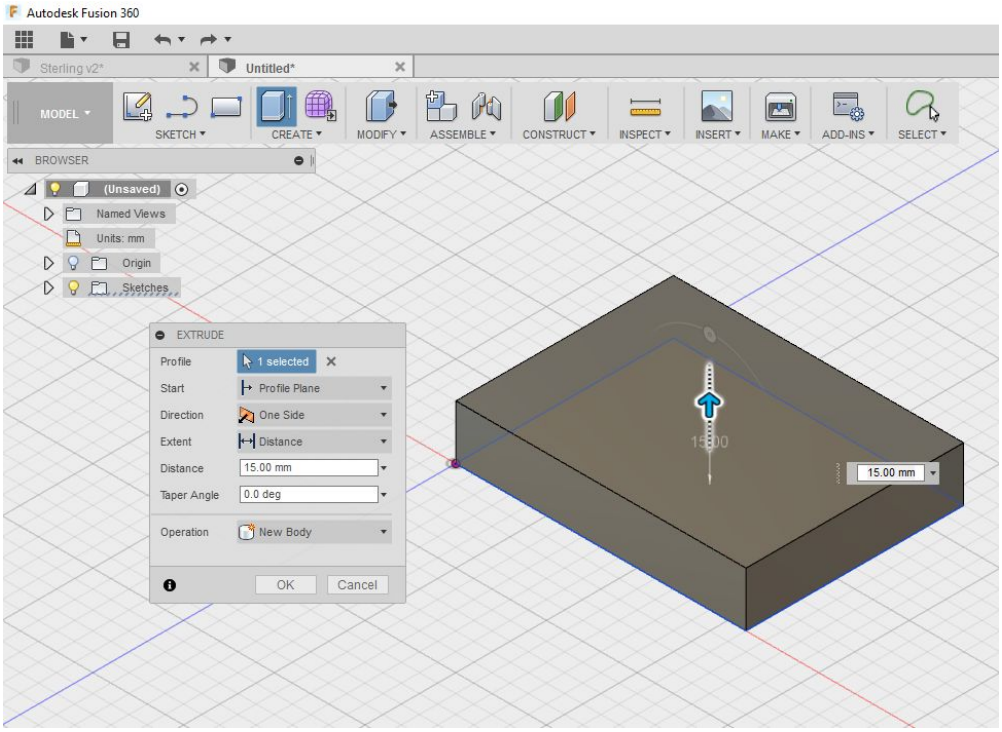


Step 1: Sketching

The features you create in Fusion 360 will often start with a 2D Sketch. In order to create intelligent designs it is necessary to have a good understanding of how to create sketches and how to apply dimensions. In this module we will cover the basic sketching tools needed to create a 2D sketch.

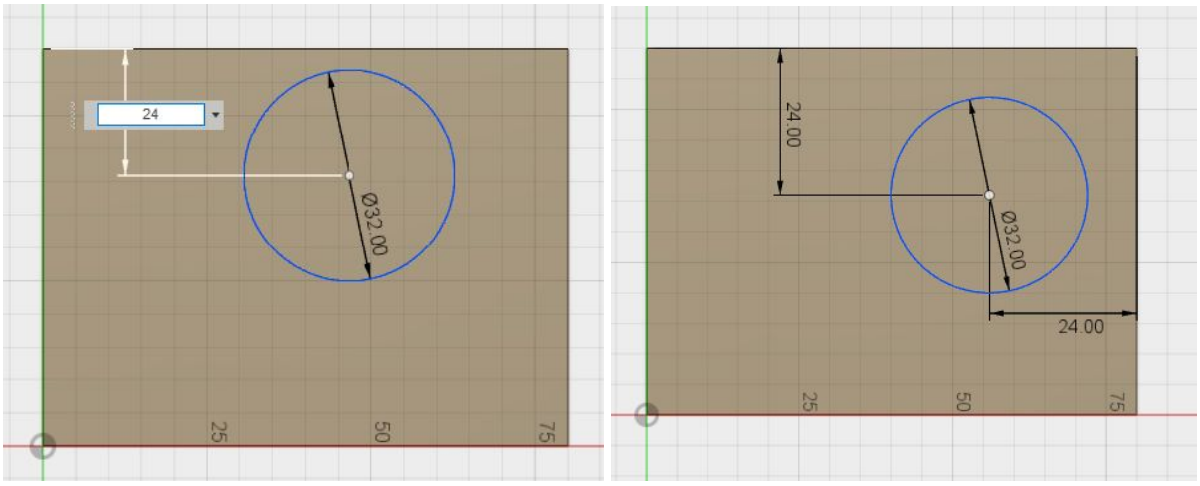
1: Select Sketch from the menu bar at the top of the screen and **Create Sketch.**
2: Select the XY sketch plane.
Note: Aside from the origin planes, you are also able to create sketches on one of the 3 default planes, on a custom construction plane or on an existing model face.
3: Select **Sketch>Rectangle** and click in the origin of the sketch, then draw a **60x80mm** rectangle by moving the mouse and click again when the rectangle has the correct dimensions.
*Note: You may choose the preferred size by typing a value of **60 mm** in the width box, press tab and then type a value of **80 mm** in the length box and press enter.*
4: Finish sketch by clicking **Stop Sketch**

Step 2: Extrusion



1: Select **Create>Extrude** or type “**e**”
2: Click on the rectangle and drag the blue arrow, or type a value of **15 mm** in order to set the depth.

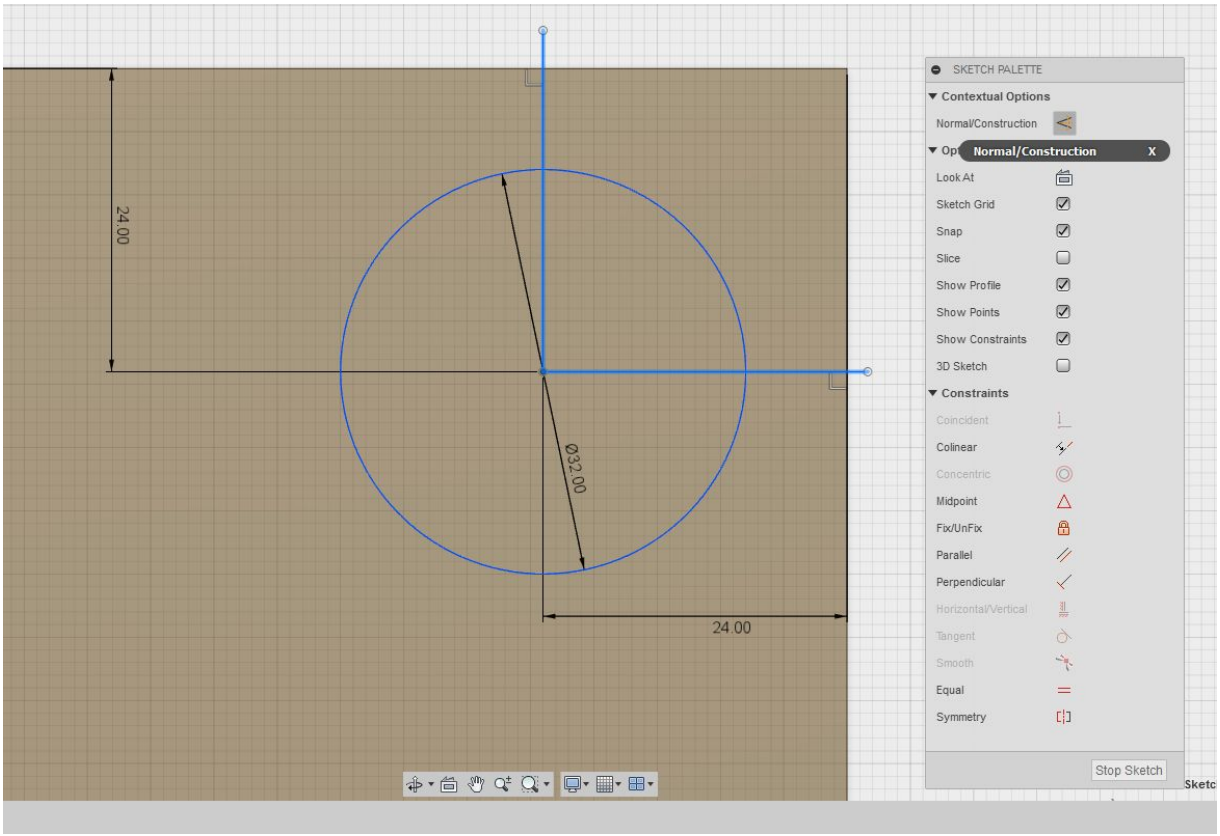
Step 3:



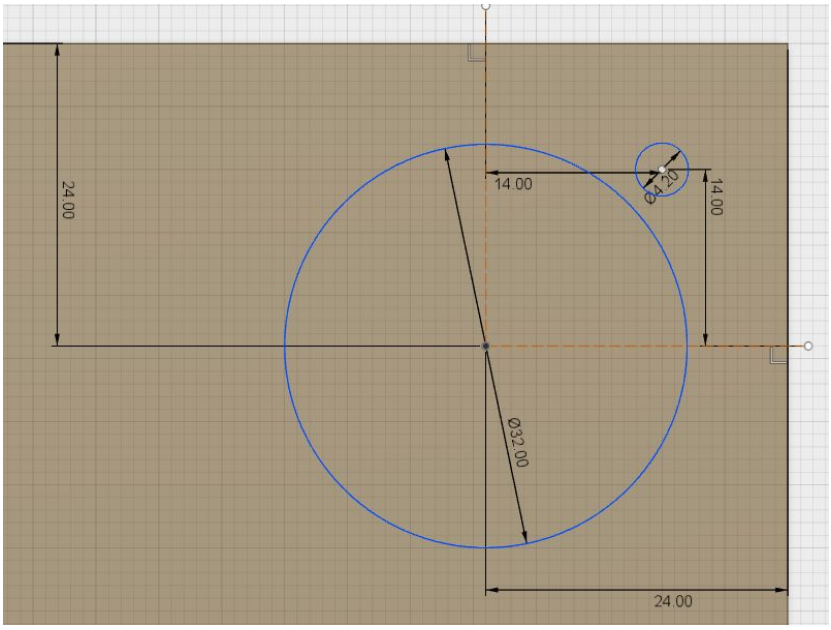
1: Select the top of the rectangle as the plane, and from the Sketch menu select **Sketch > Circle > Center diameter circle.**
2: Draw a Ø32.00 mm circle anywhere in the rectangle.

3: Select **Sketch > Sketch dimension** or type “**D”.**
4: Select the center of the circle first and then the top of the rectangle, drag the mouse to the side in order to enter the dimensions, type the value 24 in the pop-up box and enter. The center of the circle sketch should now be 24 mm from the top length of the rectangle.
5: Repeat step 4, but this time from the center and then over to the right hand side of the rectangle.

Step 4: Construction line to help position a sketch



1: Select **Sketch > Line**
2: Draw a line at an angle of 180 degrees, from the center of the circle and out to the perimeter on the right. Click once and then click the green checkmark to end.
3: Repeat step 2, but this time with a 90-degree angle, from the center and up to the perimeter at the top of the circle.
4: Now click **select** from the menu and select the first line, hold shift and select the second line.
5: From the **Sketch palette** box select **Normal/Construction** from the Contextual Options dropdown menu; the line now becomes an orange construction line.

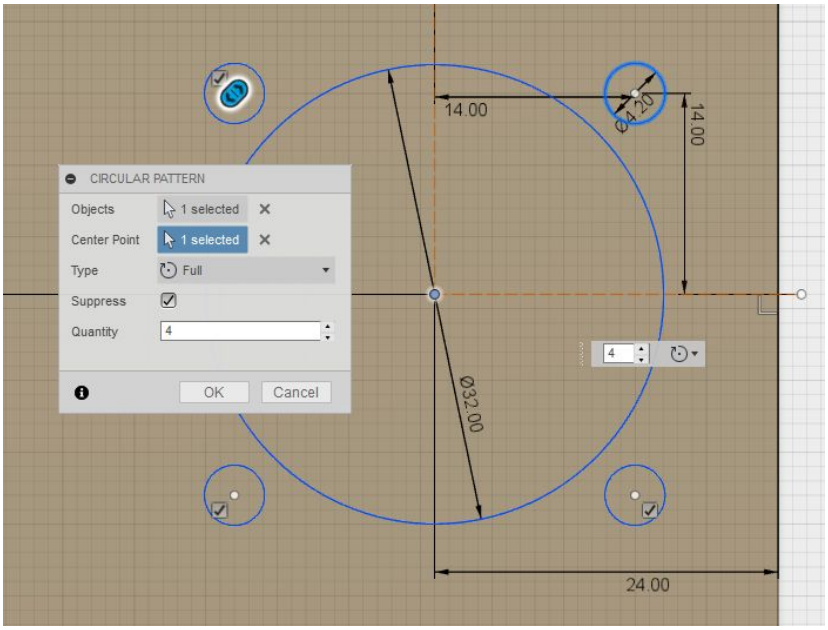


Step 5:

1: Sketch a new **Ø4.2 mm** circle at any point in the top right hand corner of the rectangle.

2: Select **Sketch > sketch dimension**, select the middle of the new circle and then the first construction line, click again and type 14.00 mm in the pop-up box.

3: Repeat step 2 for the second construction line.

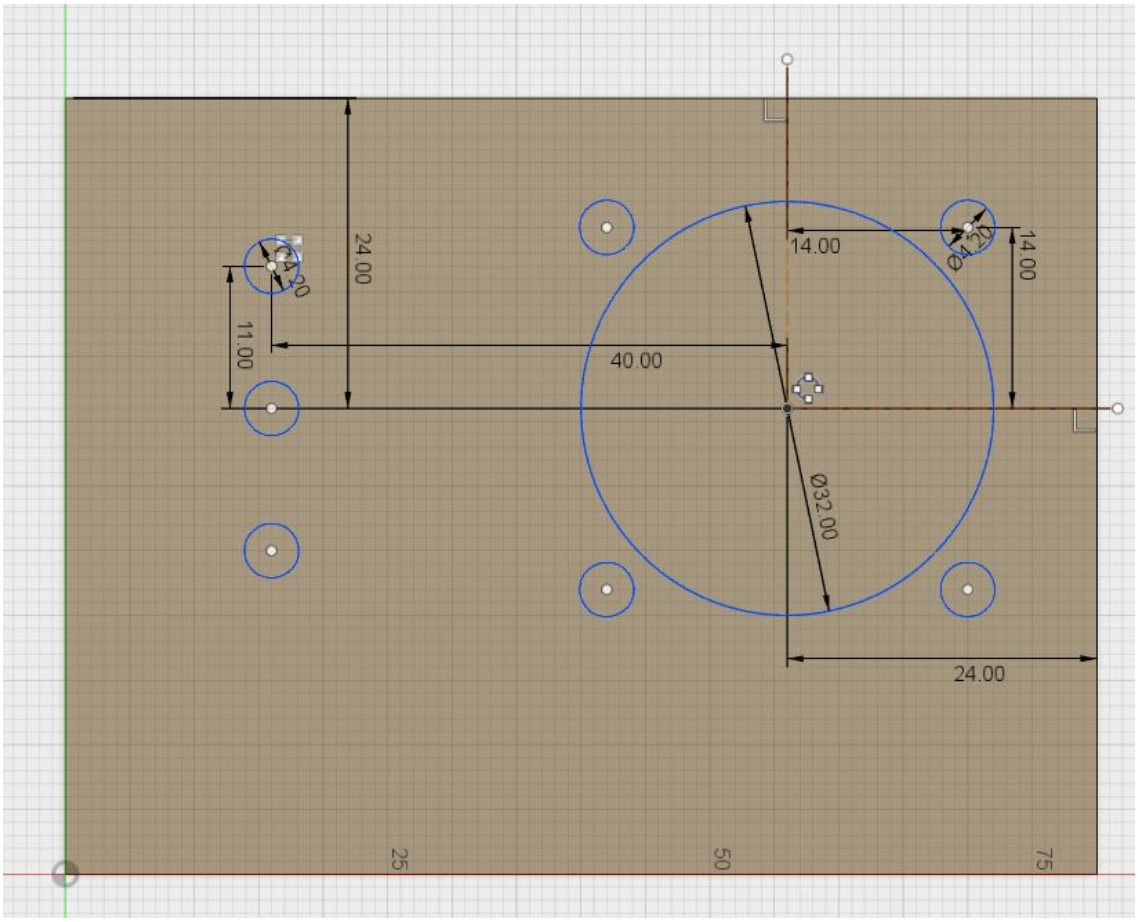


Step 6: Circular pattern

1: Select **Sketch > Circular pattern**

2: Click the Ø4.2mm circle on the drawing and the selection will show in the **object menu** inside the **Circular pattern** box. Next click the **Center point** menu and then click on the middle of the Ø32 mm circle. Select **Quantity 4** from the roll down menu and then enter. Four Ø4.2 circles will now show on the diagram.

Step 7: Rectangular Pattern



Go back to the sketch menu

1: Sketch a new **Ø4.2mm** circle.

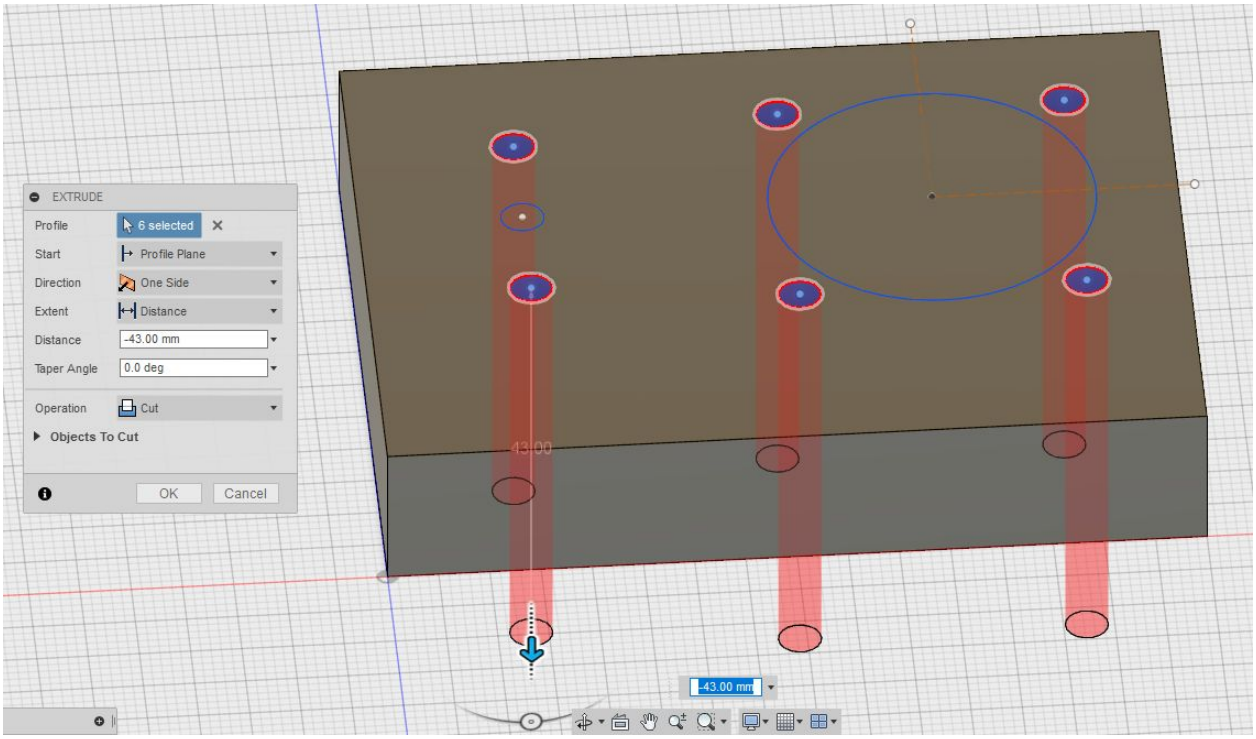
2: Type **D** and select first the middle of the circle, next the horizontal construction line and then type **11.00 mm** in the pop-up box.

3: Repeat step 2 but this time from the middle of the circle and out to the vertical construction line on the left, type **40.00mm** in the pop-up box**.**

4: **Select** Sketch>Rectangular pattern.

5: Click on the blue circle circumference line, select **Quantity 3** and drag the arrow **down to 22mm.** This will give a distance of **11.00mm between** the three holes.

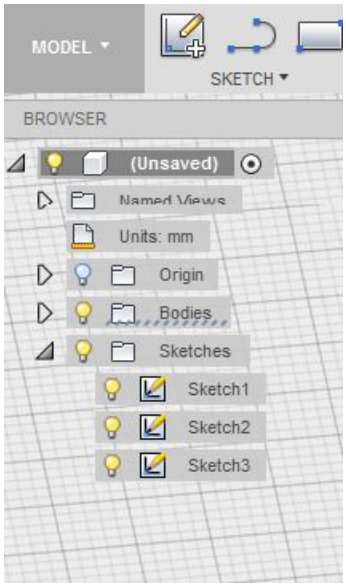
Step 8: Cut holes



1: To cut **6 x Ø4.2** holes that penetrate all the way through the rectangular block, select extrude by typing **E** and select the six circles.
2: Type a length **deeper** than the one of the work piece, for example distance -20mm and hit enter.

Note: *It is possible to rotate the work piece in order to cut the holes, by simultaneously holding in the shift button and dragging the blue arrow with the mouse wheel, through the length of the hole, until the cylinders become red. When the cylinders become red, a hole will be cut through the entire work piece.*

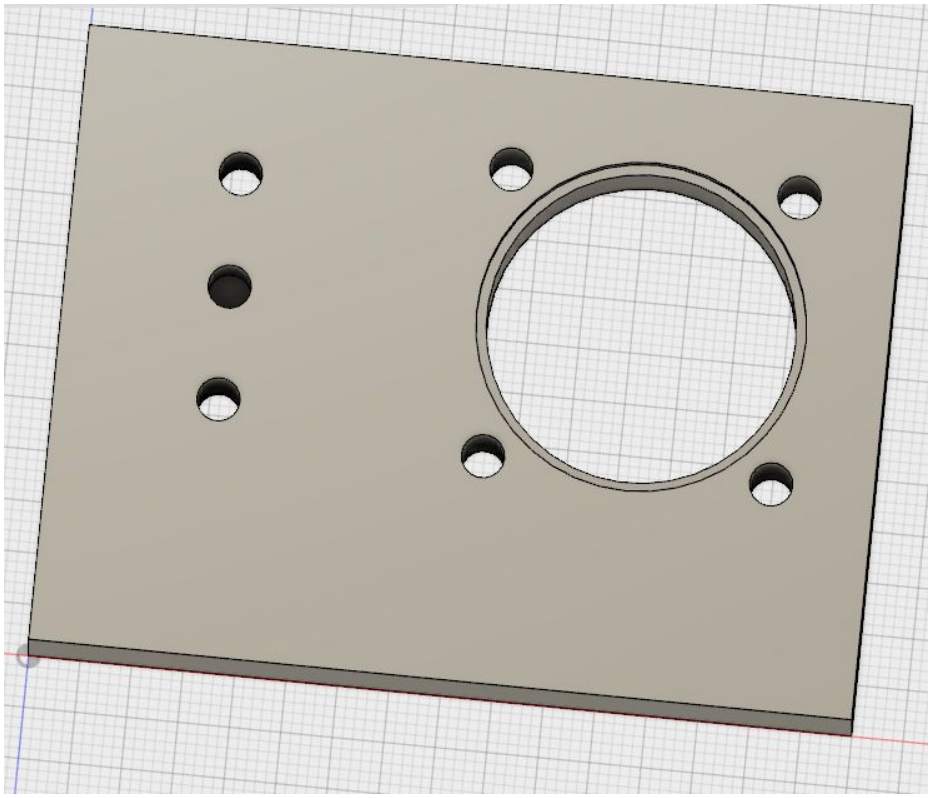
Step 9:



1: Select extrude.
Note: *If the sketches are hidden, click on the light bulb icons in the browser to the left to make the sketches visible.*
2: Select the remaining uncut **Ø4.2mm** circle and extrude **negative(-) 9.0mm.**
3: Select the **Ø32.0mm** circle and extrude **negative(-) 2.0mm** in order to cut a shallow hole.
4: Select **Create Sketch** and click on the plane inside the **Ø32.0mm** hole.
5: Sketch a smaller **Ø30mm** circle from the center of the hole.

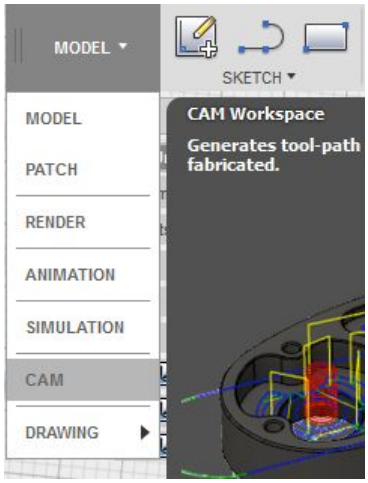
6: Extrude the **Ø30 mm** sketch all the way through the work piece to cut the rest of the hole. The top of the hole will have a 2.0mm rim remaining from the first extrusion.

The CAD model is now finished and should look similar to the model in the image:

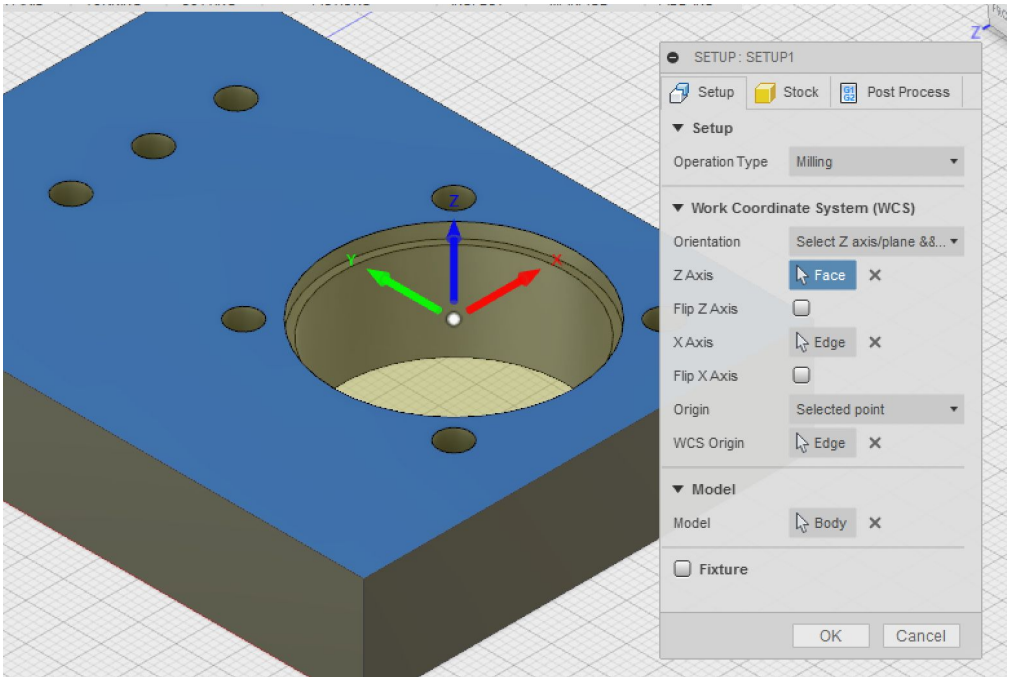


CAM

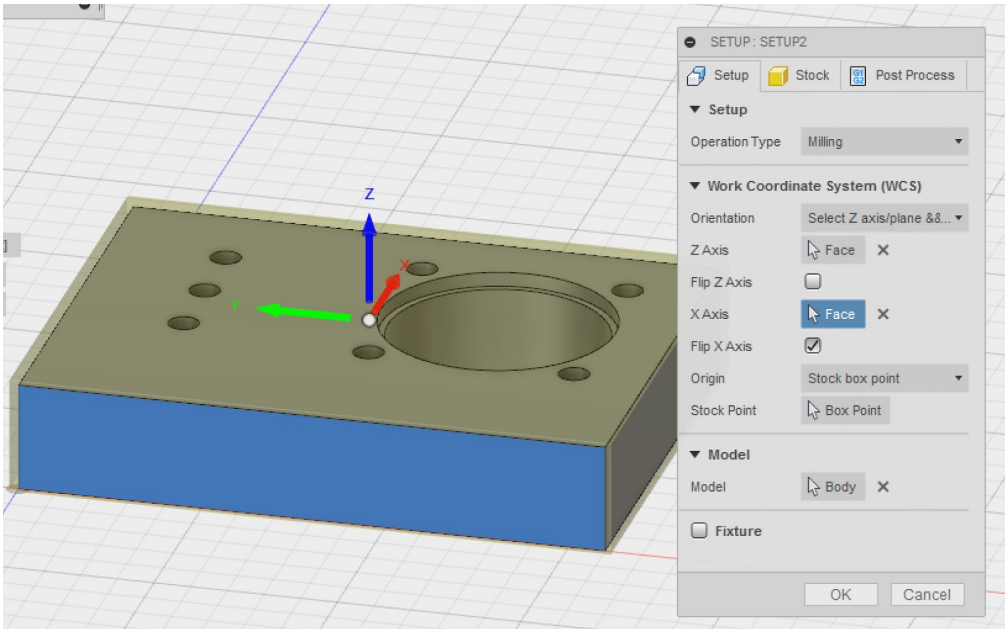
Save your file and go to the **CAM** section in Autodesk, by selecting **Model > CAM**.



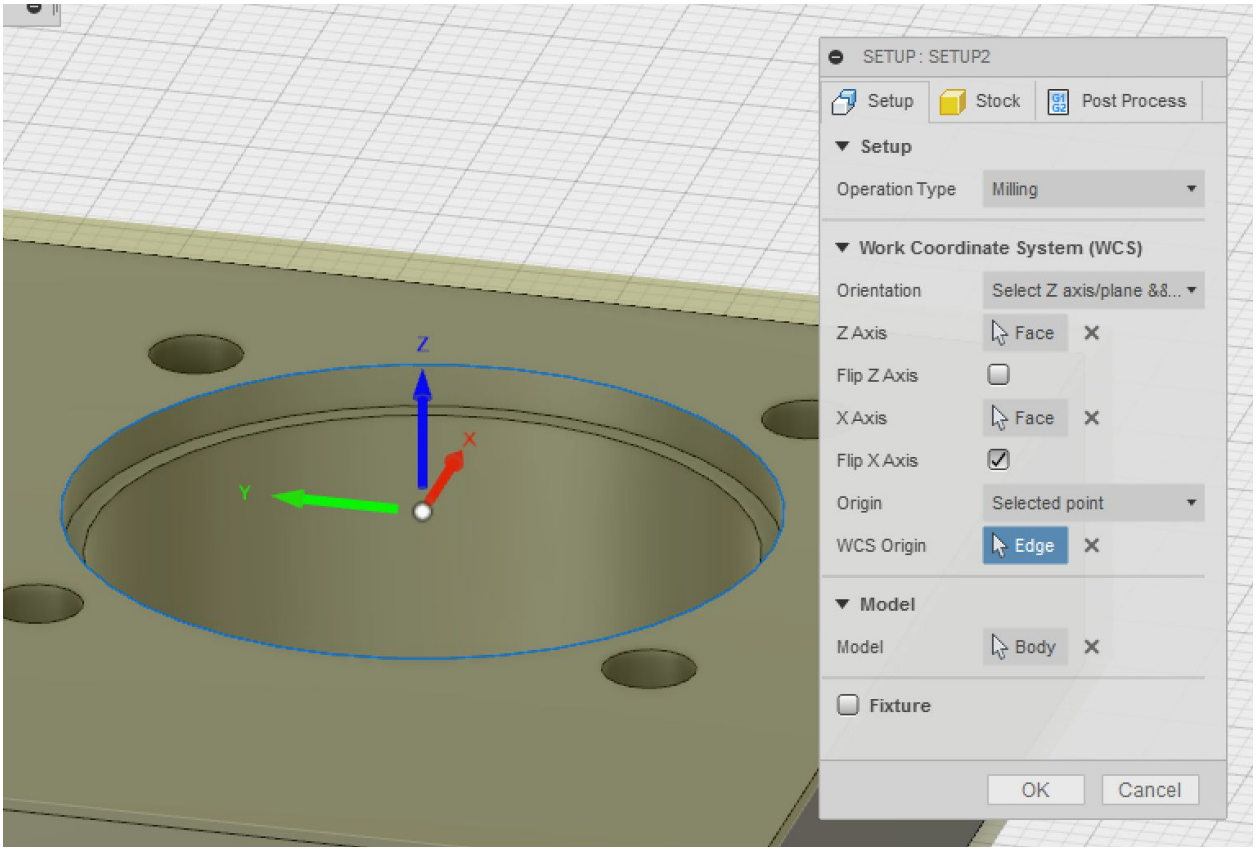
Step 1: Setup



1: Select **Setup > New setup**.



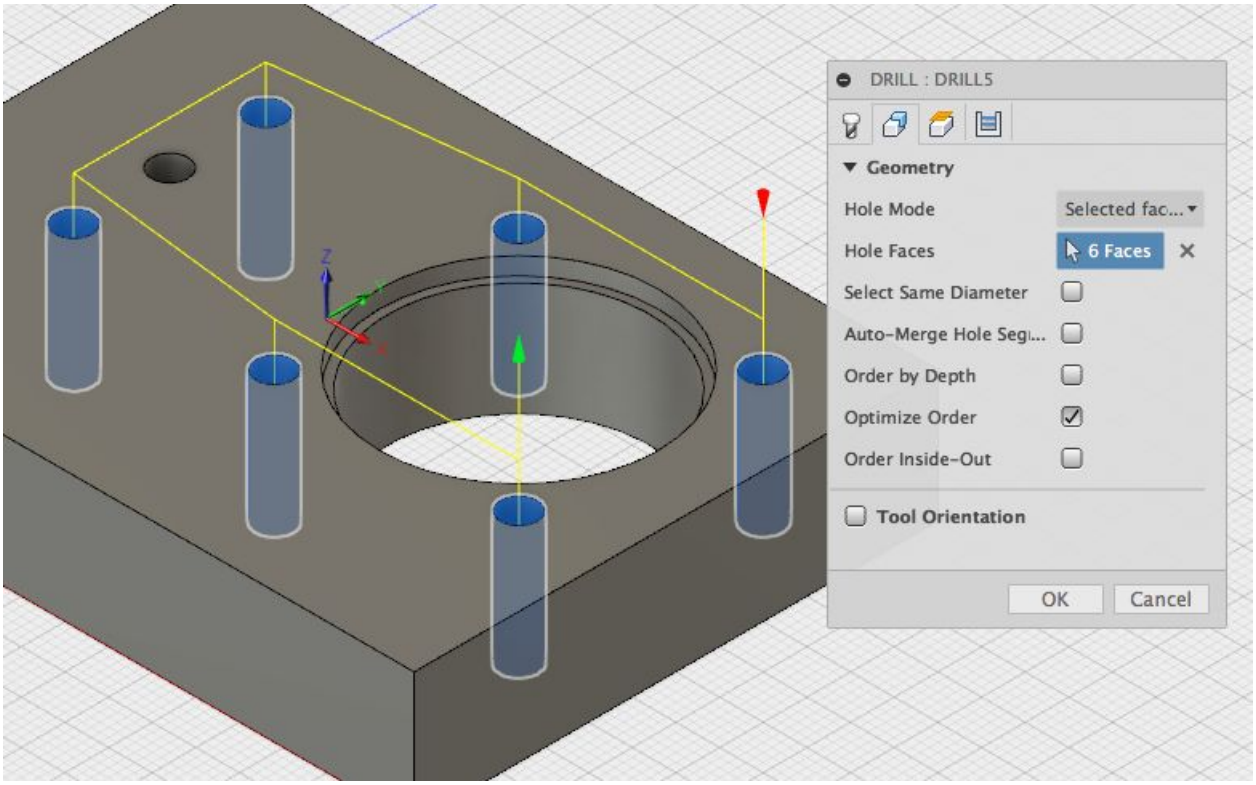
2: Choose orientation: Select **Z axis/plane & X axis**, click on the top of the workpiece to select the Z-Axis, and on the side face to select the X-Axis. Then check the “Flip X Axis” box.



3: Choose select point as Origin and click on the edge of the hole.

4: Choose the **Stock** tab and select **no additional stock**. Click **OK**.

Step 2: Drilling:

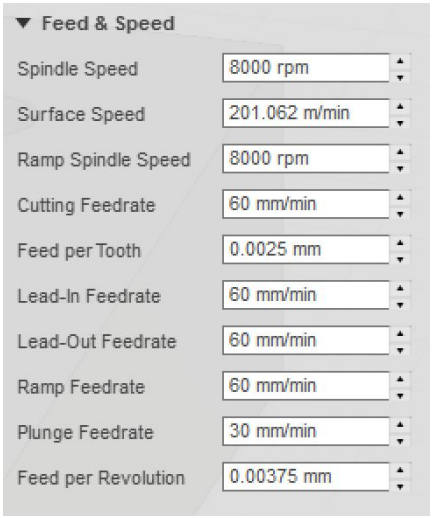


1: Select the drilling tab and then select tool.
2: Type **Ø4.2 mm** in the search bar and select the brass drill.
3: Select the **Geometry** tab and then select the **6 x Ø4.2mm hole faces**.
4: The drill tip is pointed and in order to clear the full hole, the drill tip must penetrate further than the bottom of the hole. Select the **Heights** tab, under **bottom height** check the box with **drill tip through bottom**, and the drill will automatically clear the full hole.
5: Under the tab **cycle,** switch from drill to deep drill full retract and click OK.
6: Repeat step 1 and 2, but now select the **Ø4.2 x 9mm hole**. Do **not** check the box with drill tip through bottom.
7: Repeat step 5.
Note: *Regular drill will push all the way through in one pass, while deep drill full retract is going peck the drill every 2.5 mm which is a much safer way to drill.*

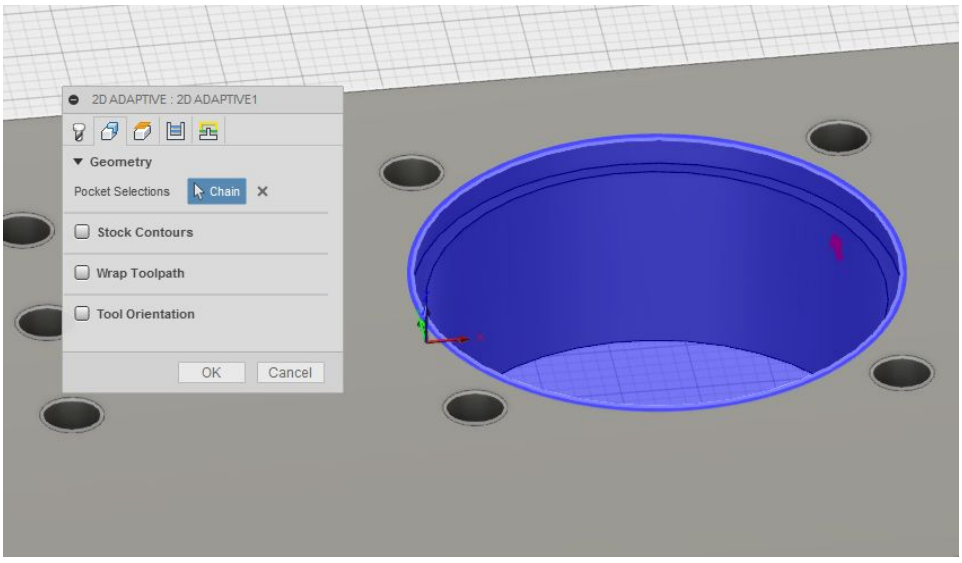
Step 3: Machine a pocket:

1: Select **2D> 2D adaptive clearing**

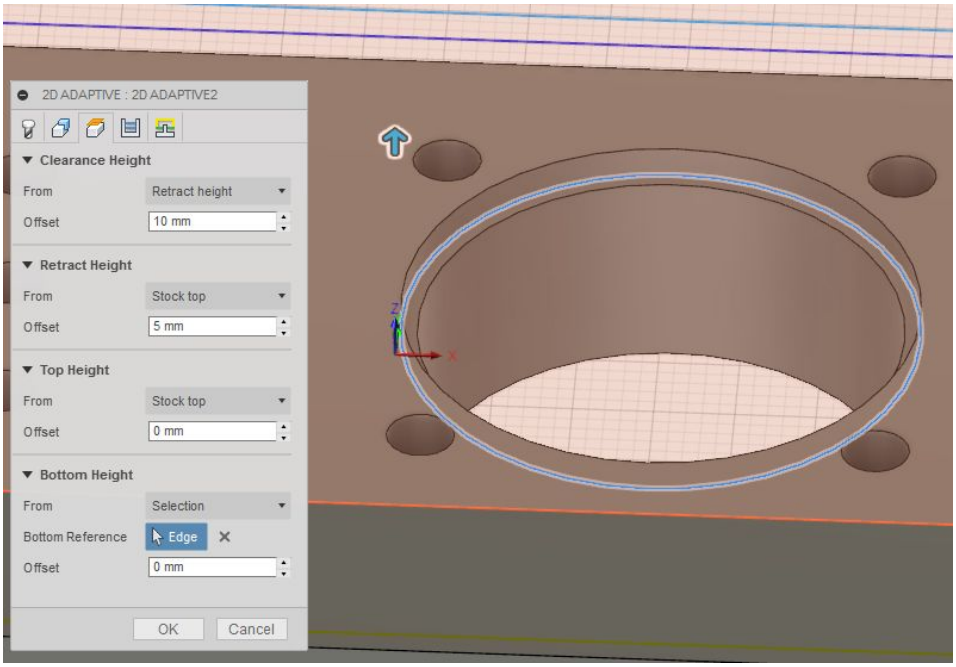
2: Click on **Select tool** and in the search bar type **Ø8 mm flat**, select the brass tool.



3: Choose all feedrates to be 60 mm/min and Plunge feedrate to be 30 mm/min

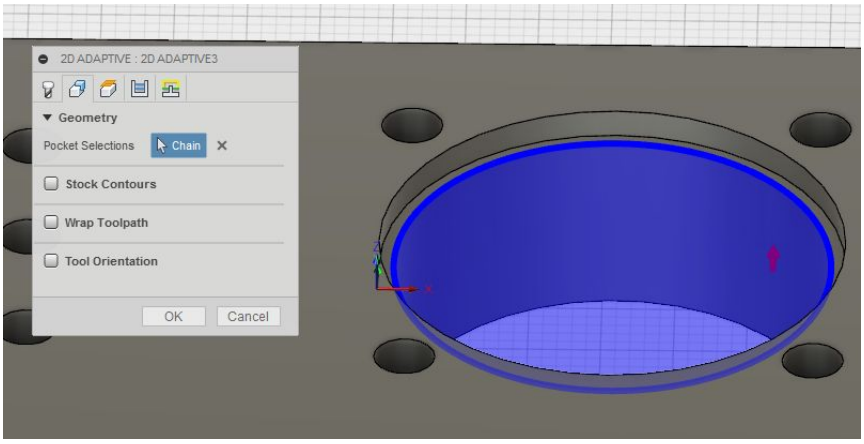


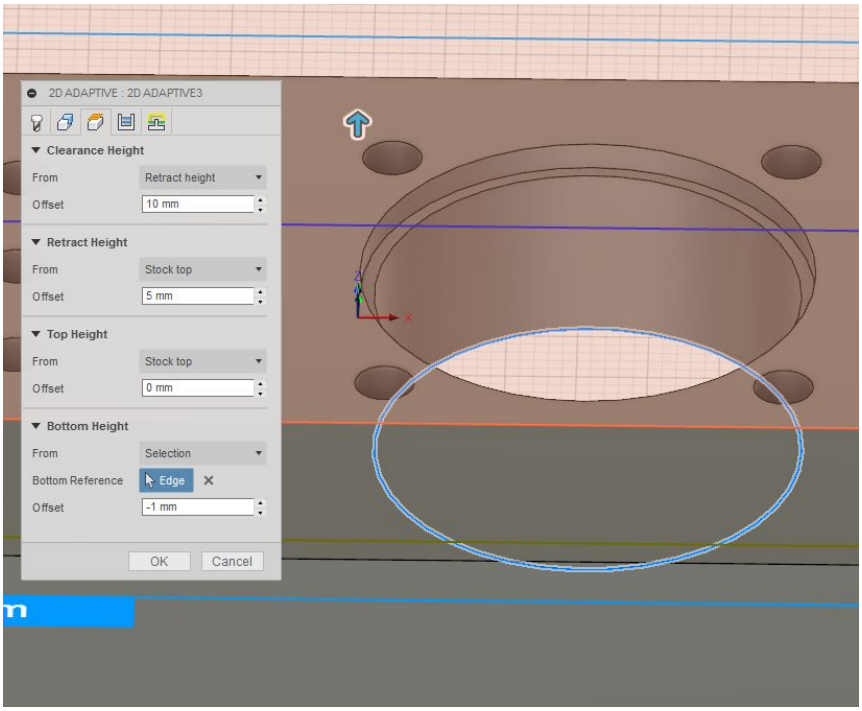
3: Select **Geometry** tab and click on the top edge of the Ø32.0mm hole



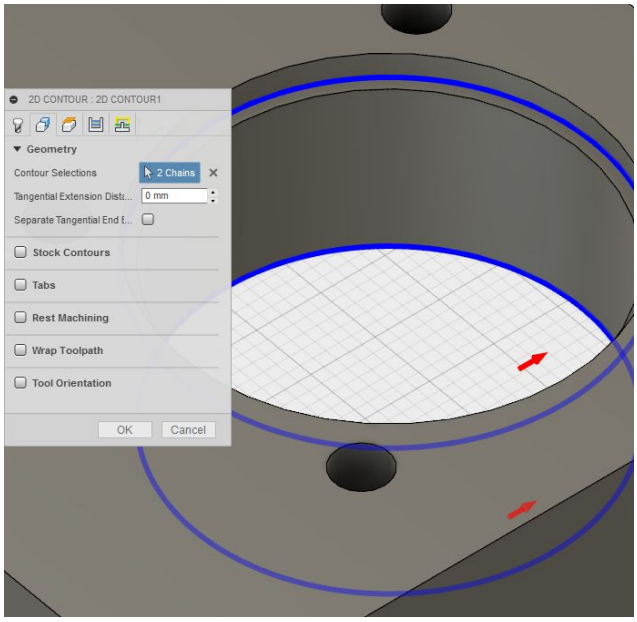
4: In the **heights** tab choose **Bottom height > from selection** and select the **edge** as the bottom reference.

5: Select the **linking** tab and under **Ramp** choose **Ramp Clearance Height** to be **0.5mm**





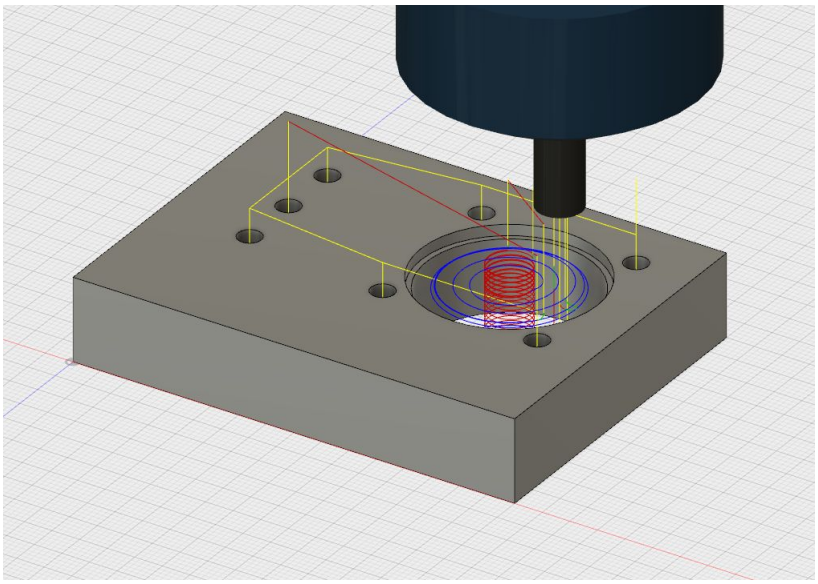
6: Repeat steps 1-5 for the pocket selection from the Ø30mm edge and choose bottom reference as the bottom edge, and set offset to negative 1 mm to be sure it goes all the way through.



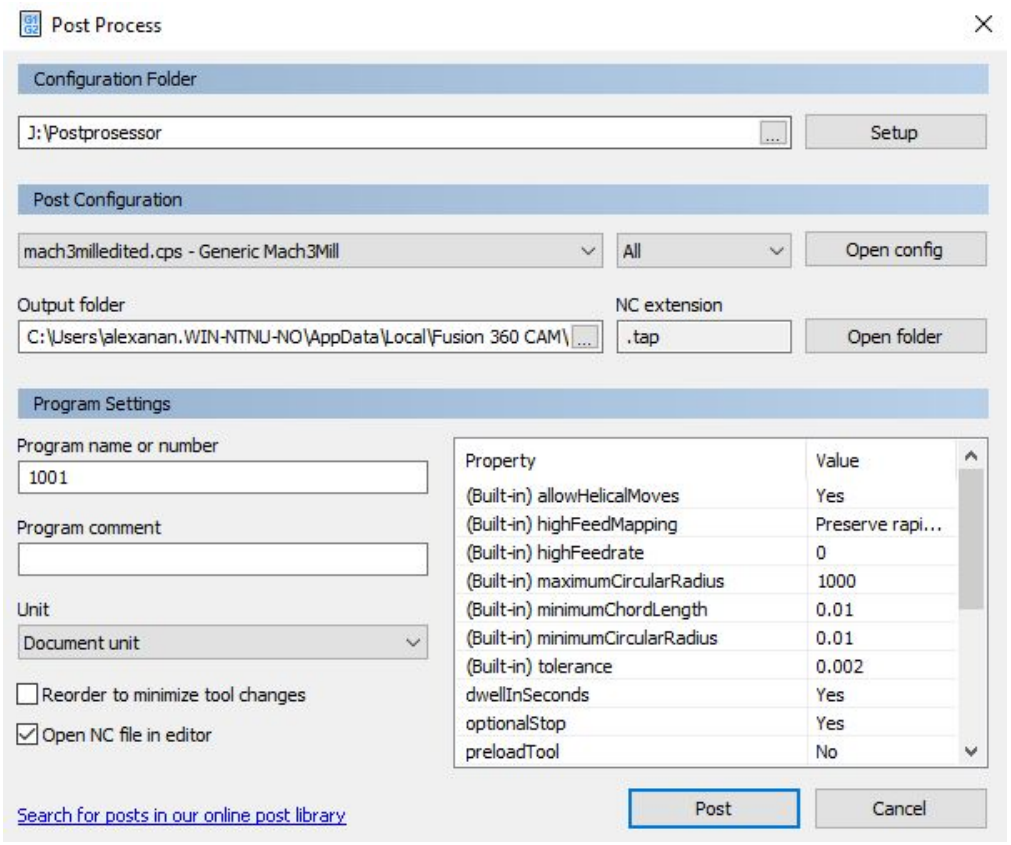
Step 4: 2D Contour.

1: Select **2D > Contour**
2: Click on the **geometry** tab and choose **contour sections** as the bottom edges of the **Ø32** mm hole and the **Ø30** mm hole.
3: Select **Heights > bottom height > selected contours** and click OK.
Note: *2D adaptive is used as a roughing strategy, and will leave material on the side, to clean up the holes we need to use 2D contour.*

Simulate.
To simulate the operations choose simulate in the actions tab, select the **setup folder** in the **browser**, and press play to simulate the entire process.



Post process
1: Select **Post Process** in the Actions tab
2: Find the configuration folder where the post processor **mach3milledited.cps** is stored.



3: Name the program and generate the G-Code

Appendix D

NC-kode programmert manuelt av studentene i TPK4190

```
Drilling
M3
G00 Z10
X14 Y-14
G01 Z-17 F50
G00 Z10
X14 Y14
G01 Z-17 F50
G00 Z10
X11 Y40
G01 Z-17 F50
G00 Z10
X0 Y40
G01 Z-9 F50
G00 Z10
X-11 Y40
G01 Z-17 F50
G00 Z10
X-14 Y14
G01 Z-17 F50
G00 Z10
X-14 Y-14
G01 Z-17 F50
G00 Z10
X0 Y0
G01 Z-17 F50
G00 Z10
M30
```

```
Fresing
M3
G00 Z10
X5 Y0
G00 Z1
G03 X5 Y0 Z-1 I0 J0 F60
G03 X5 Y0 Z-3 I0 J0 F60
G03 X5 Y0 Z-5 I0 J0 F60
G03 X5 Y0 Z-7 I0 J0 F60
G03 X5 Y0 Z-9 I0 J0 F60
G03 X5 Y0 Z-11 I0 J0 F60
G03 X5 Y0 Z-13 I0 J0 F60
G03 X5 Y0 Z-15 I0 J0 F60
G03 X5 Y0 Z-17 I0 J0 F60
```

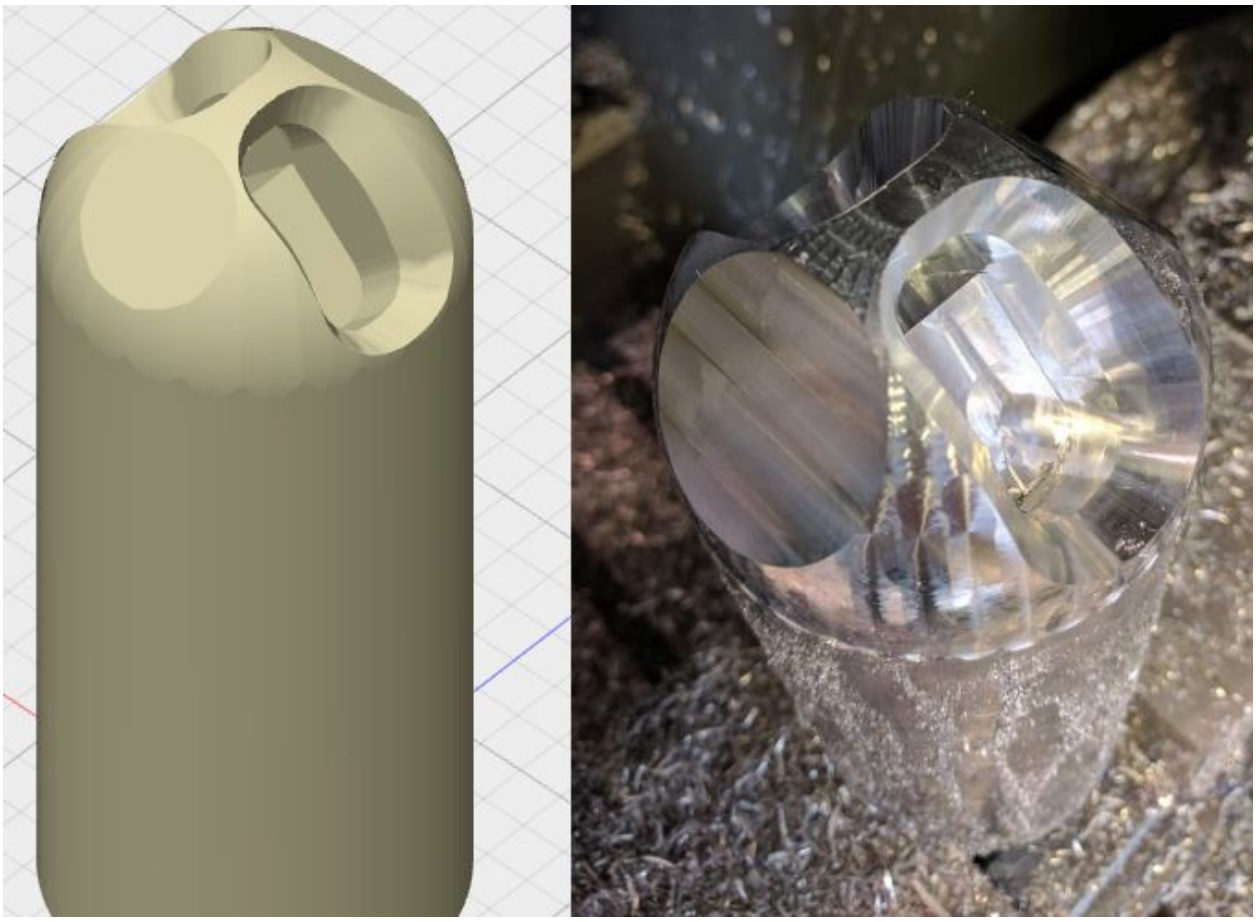
```
G00 Z1
X11 Y0
G03 X11 Y0 Z-1 I0 J0 F60
G03 X11 Y0 Z-3 I0 J0 F60
G03 X11 Y0 Z-5 I0 J0 F60
G03 X11 Y0 Z-7 I0 J0 F60
G03 X11 Y0 Z-9 I0 J0 F60
G03 X11 Y0 Z-11 I0 J0 F60
G03 X11 Y0 Z-13 I0 J0 F60
G03 X11 Y0 Z-15 I0 J0 F60
G03 X11 Y0 Z-17 I0 J0 F60
G00 Z-2
G01 X12 Y0
G03 X12 Y0 I0 J0 F60
G00 Z10
M30
```

Appendix E

Tutorial PhD Students

Introduction

Autodesk Fusion 360 is a 3D CAD & CAM tool. It connects your entire product development process in a single cloud-based platform that works on both Mac and PC. Students get free educational access, sign in or create an Autodesk account to download and start running Fusion 360 on your Mac or PC.



This is a step-by-step guide on how to design a CAD model and program a CAM setup for 5-Axis milling using Fusion 360. The image to the left is from Fusion 360 and the image to the right is the final result after milling the workpiece. We are milling the workpiece using five axis milling on the Deckel Maho 50 eVolution machine in the workshop at NTNU Valgrinda.

CAD Model

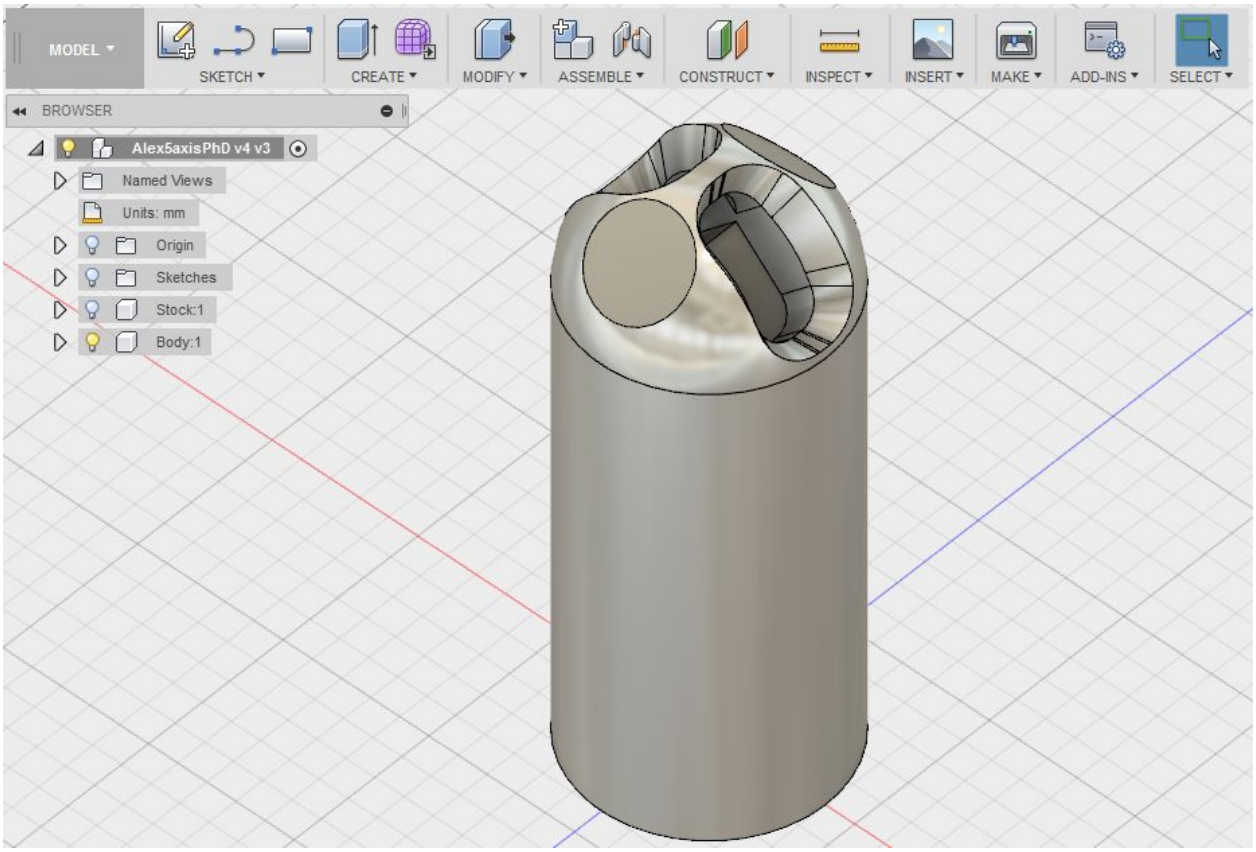
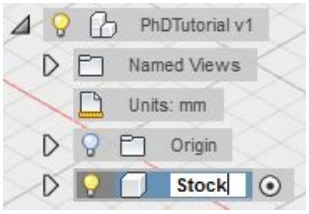


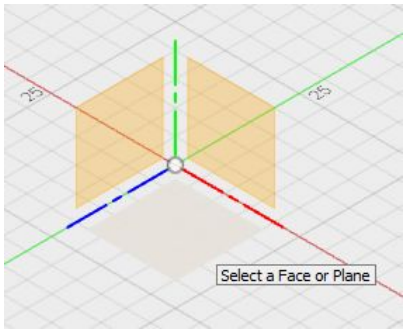
Image of the final CAD model in this guide

Step 1

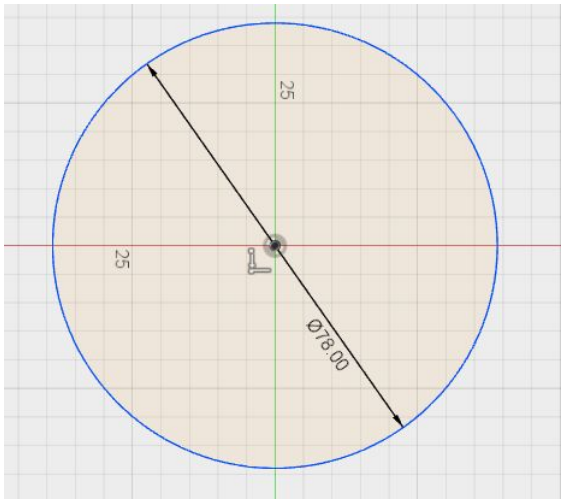
- Open a new design in Fusion 360 by selecting **file** → **new design**
- Save the file
- Right click on the file name in the left corner, and choose **new component**
- Double click on the new component and name it **Stock**

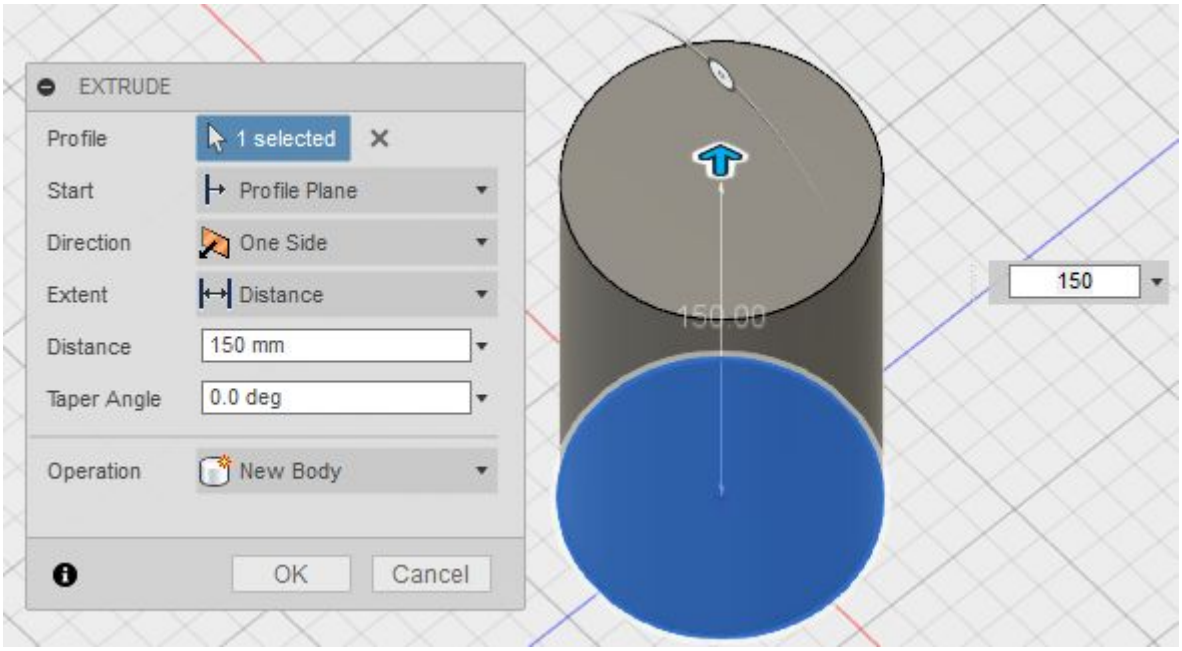


Step 2



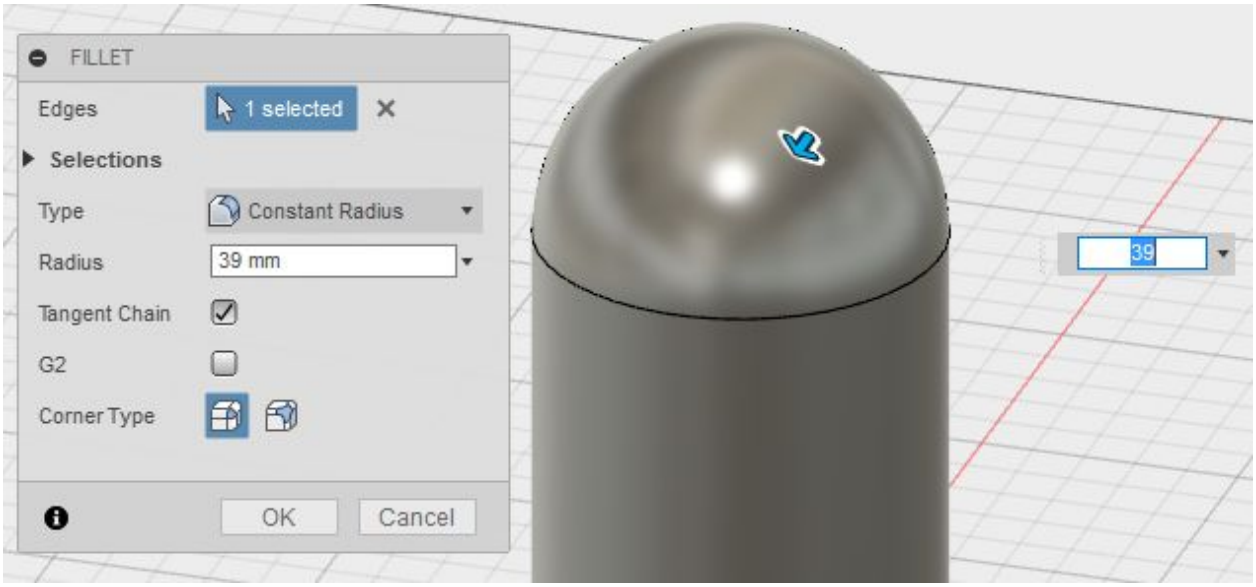
- From the top menu go to **Sketch → Circle → Center diameter circle** (in the rest of the guide we will use the shortcut “C” to sketch a center diameter circle)
- Select the **ZX-plane** as the sketch plane. (Note the coordinate axis are shown in the top right corner)
- Click in the **origin** to place the center of the circle
- Drag the mouse and type **78** to get a circle diameter of 78 mm, and hit enter twice.
- From the top menu go to **Create → Extrude**. (in the rest of this guide we will use the shortcut “E” to extrude.)
- Click on the circle sketch and type **150** to extrude 150 mm.





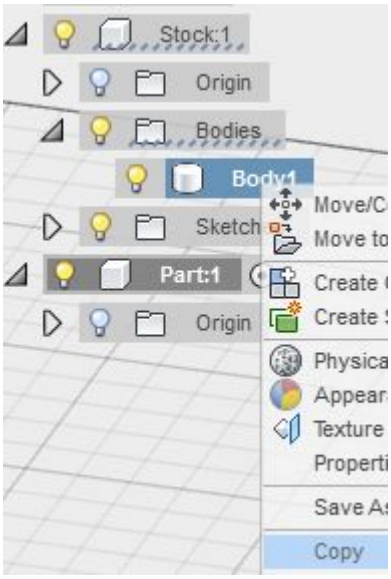
Step 3 Fillet

- From the top menu go to **Modify** → **Fillet** (We will use the shortcut “F” for fillet in the rest of this guide)
- Choose the corner of the cylinder as Edge and type **39** to get a radius of 39 mm

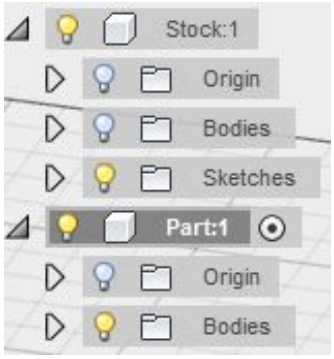


Step 4 New component

- Right click on the model name in the top folder to the left.
- Select **new component**
- Name the new component **Part**
- Open the subfolders in the stock component by clicking on the **arrow** next to Stock:1
- Open the **bodies** folder, select **Body1**, right click and select copy.
- Select the **Part:1** folder, right click and paste, click OK.
- Now **hide** the body in the **Stock:1** folder by **turning off** the light bulb next to the bodies folder. To turn it off just click on the light bulb.



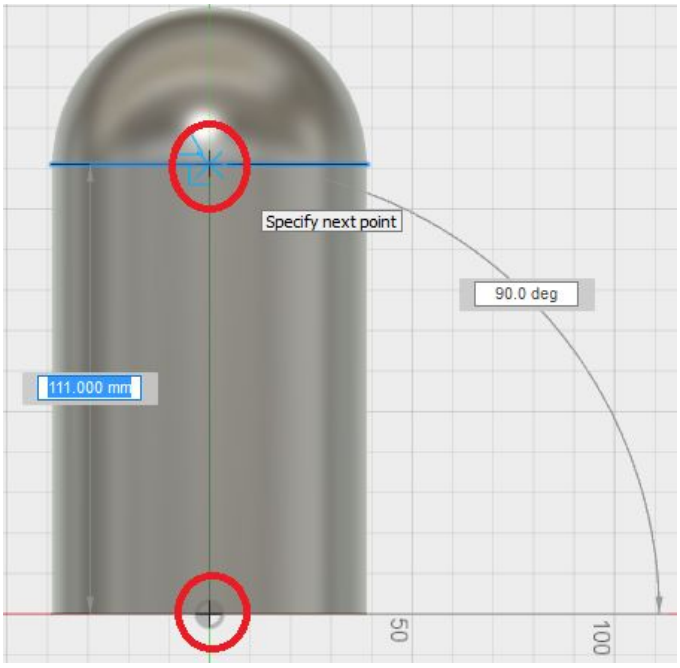
The folders should now look similar to the image, with only the light bulb next to the **Bodies** folder in the **Part:1** folder turned on.



Step 5

- From the top menu menu select **Sketch→ Line** (Further on in this guide we will use the shortcut “L”)
- Select the **XY-plane** as Sketch plane

- Draw a line from the **origin** of the cylinder to the **origin** of the sphere (marked with red circles in the image) set the angle to be **90 degrees** and the length to be **111 mm.**



- Type **L** to sketch a new line and draw it from the origin of the sphere **45 degrees** and a length of **39 mm**. Use tab to switch between angle and length and hit enter when the right values are chosen

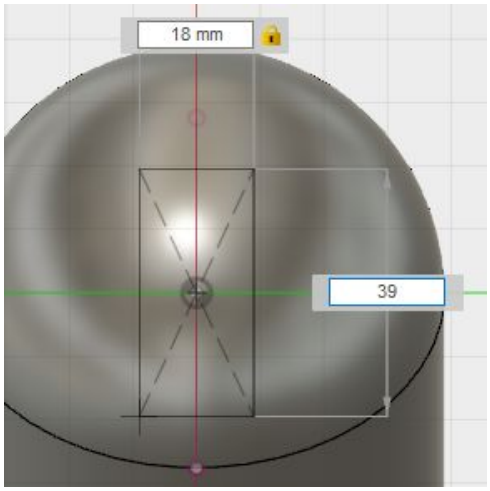
Step 6 Plane

- From the top menu menu go to **Construct** → **Plane along path**.

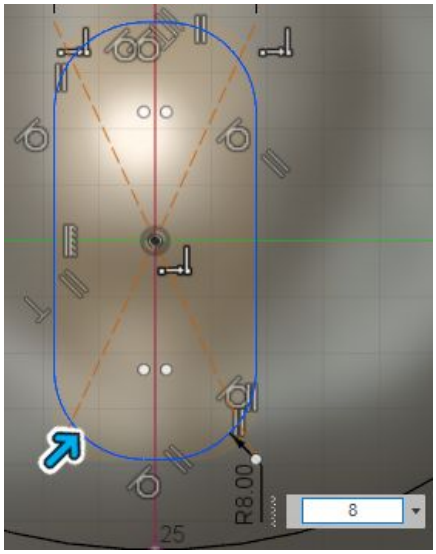
- Select the 45-degree line as the path.
- Set the distance to be **1** and press OK.

We have now made a new plane we can use to sketch on. To view the plane, orbit the model by holding shift + the mouse wheel whilst moving the mouse. Note: If you want to use Solidworks settings to Pan, Zoom and Orbit shortcuts go to your username in the top right corner and choose preferences.

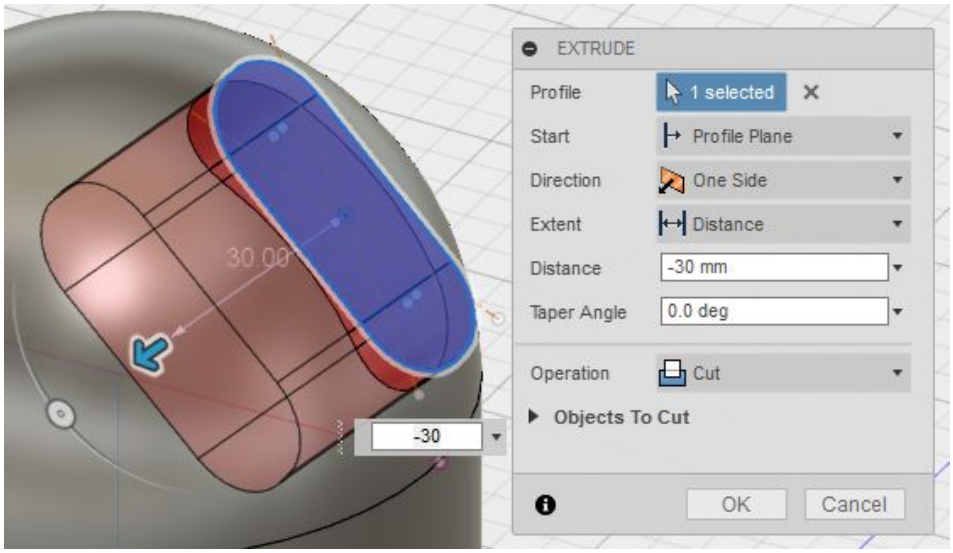
Step 7



- Select **Sketch→ Rectangle → Center rectangle.**
- Select the new plane as sketch plane. (If the plane is hidden open the construction folder on the left by clicking the arrow, and click on the light bulb next to Plane1)
- Select the center of the sphere as the midpoint and type **18 mm** as width and **39 mm** as height and hit enter.



- From the top menu select **Sketch→Fillet.**
- Select the lines that make the corners, type **8** and enter.

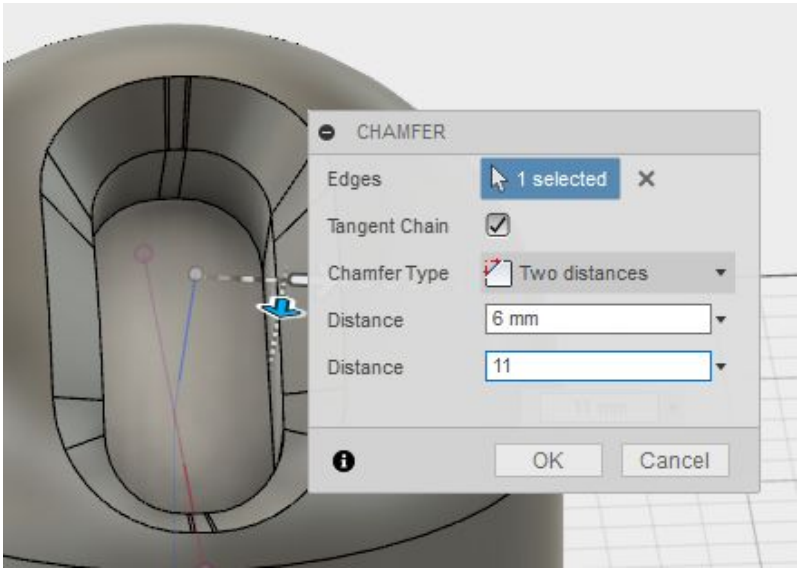


- Press **E** to extrude
- Select the sketch and **cut -30mm**

down and press OK

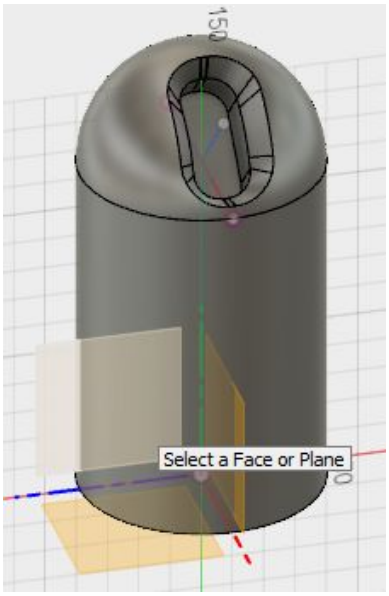
Step 8

- From the top menu select **modify→ chamfer**
- Select the edge of the sketch
- Choose chamfer type: **Two distances**
- Type **6** and **11 mm** as distances

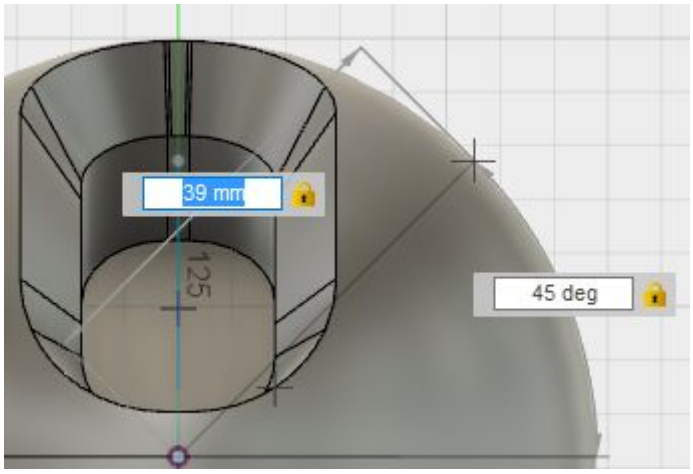


Step 9

- Press **L** to sketch a new line
- Choose the **YZ-plane**. (If the origin is hidden, turn on the light bulb next to the origin folder to the left)

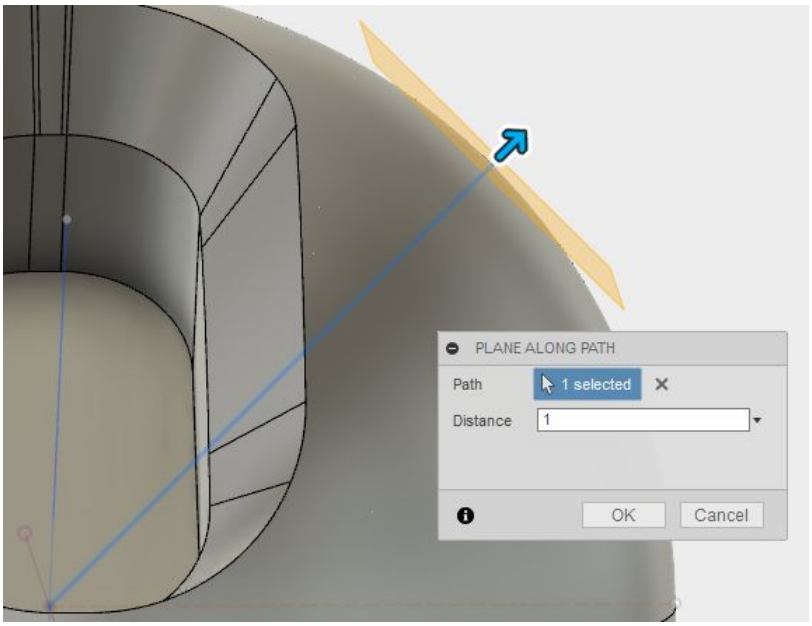


- Draw a **39mm** and **45-degree** line from the origin of the sphere



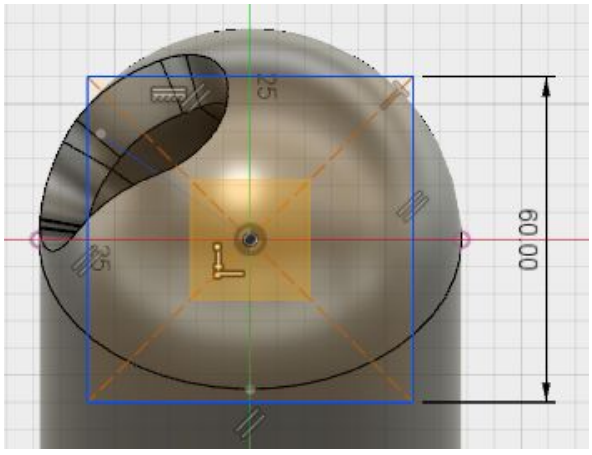
Step 10

- From the top menu select **Create → Plane along path**.
- Select the sketch line as path and set the distance to **1.**

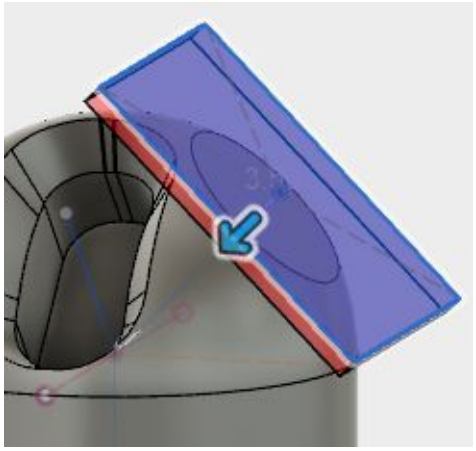


Step 11

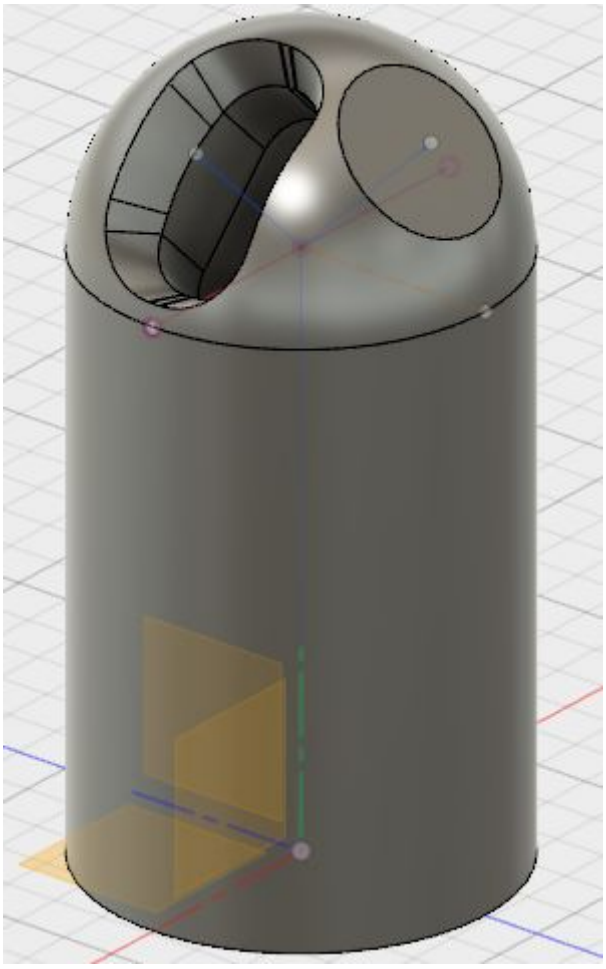
- From the top menu select **Sketch→ Rectangle → Center rectangle**
- Choose the new plane as sketch plane
- Draw a **60x60mm** rectangle



- Press **E** to extrude and select the rectangle
- Set the distance to **-3.5mm** and click OK

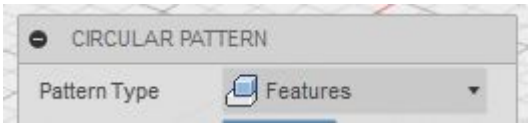


Your model should now be similar to the model in the image.



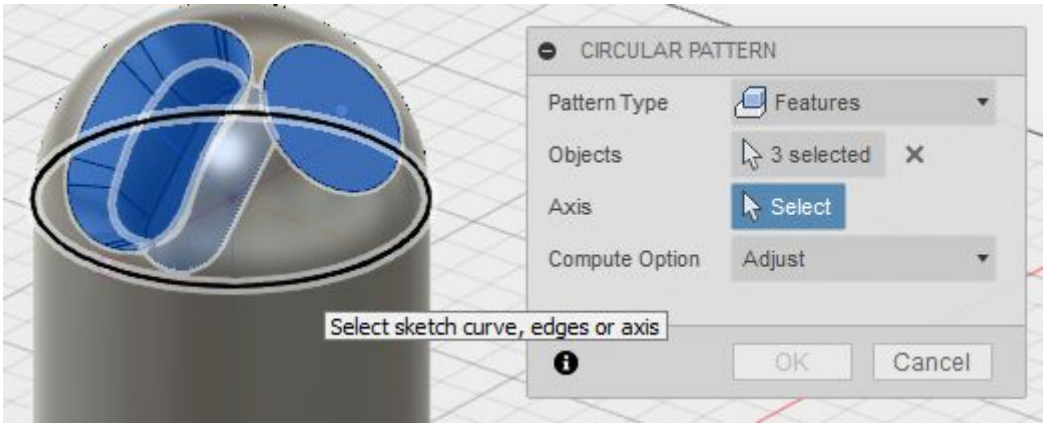
Step 12 Circular pattern

- From the top menu select **Create→ Pattern → Circular pattern**
- In the pop-up box select **Pattern type: Features.**



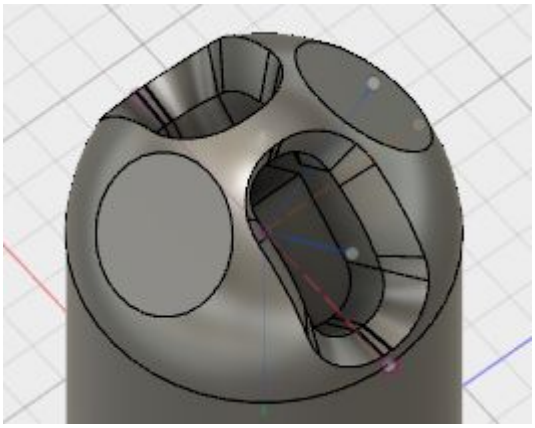
- From the timeline at the bottom of the screen select the objects highlighted in the image below

- Choose the end of the cylinder as axis by clicking on the line



- Select quantity **2** and OK

Your model should now be similar to the one in the image



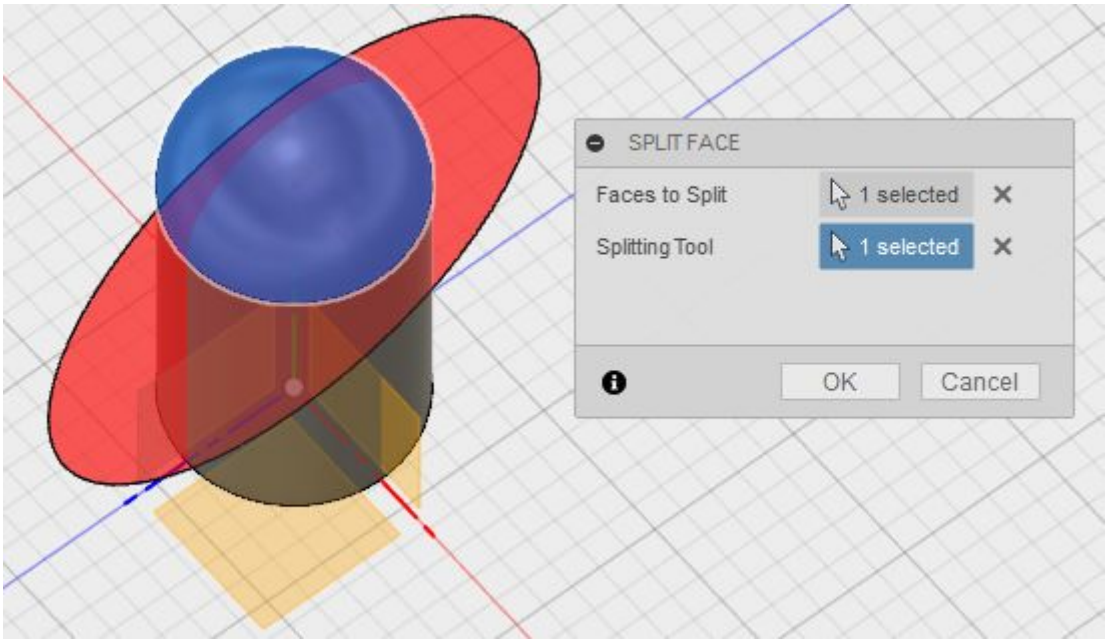
Step 13

- Hide the model by turning off the light bulb next to Bodies in the **Part:1** component folder

- Activate the **Stock:1** component by clicking in the **target** midpoint next to the folder
- Turn on the **Origin and Bodies** light bulbs in the Stock:1 component folder

Step 14

- From the top menu menu go to **Modify→ Split face**
- Select the **Sphere** as **Face to split**

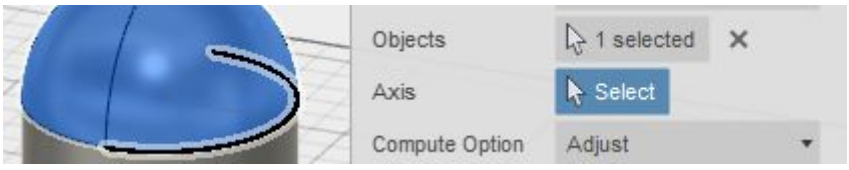


- Select the **YZ origin plane** as the splitting tool and click OK. *Note: if the origin axis planes are hidden they can be accessed by either zooming out using the mouse wheel.*

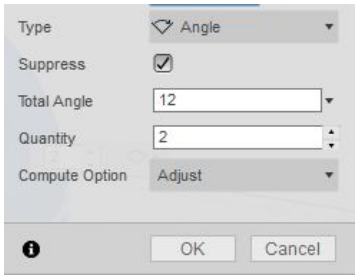
Step 15

- From the top menu go to **Create→Pattern→ Circular pattern**
- Choose **Pattern type: Features**
- Select the split face feature from the timeline in the bottom of the screen as object

- Select the line at the edge of the rectangle as axis

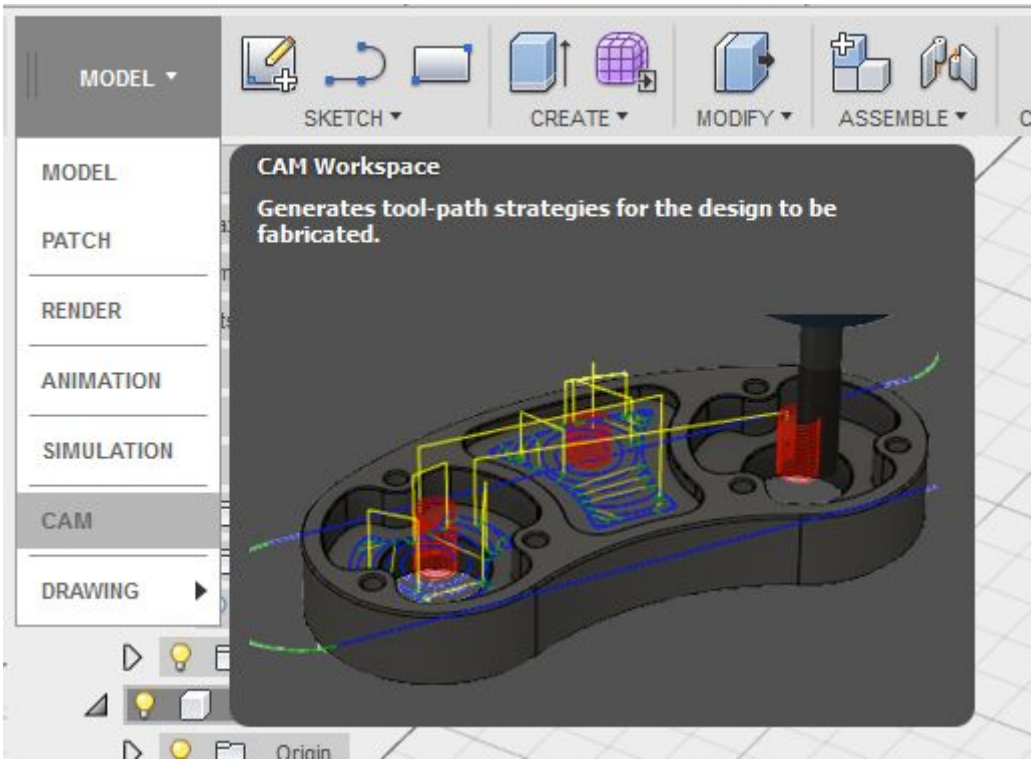


- Choose **Type: Angle**
- **Total Angle: 12**
- **Quantity: 2**



*Note: We use the split faces as toolpaths in when we program the CAM program later. We use a 12 degree angle so we can cut the sphere half in 15 operations (2*12*15=360) in step 29.*

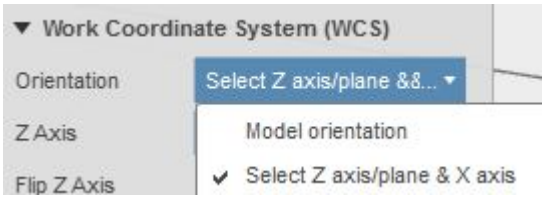
CAM



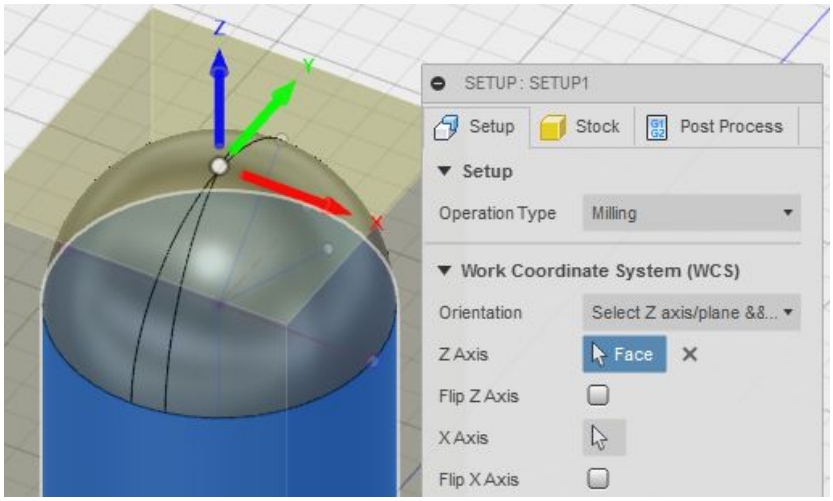
Go to the **CAM** section of Fusion 360 by selecting CAM in from the drop down menu in the top right corner.

Step 16

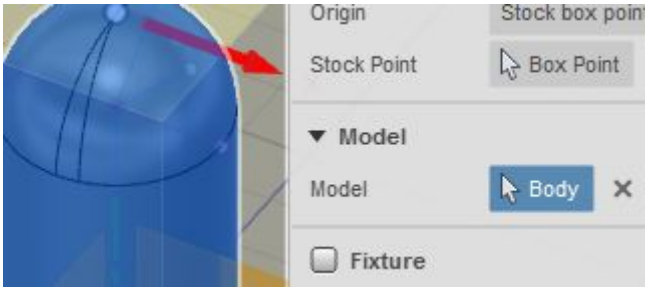
- From the top menu select **Setup→ new setup**
- In the **Work Coordinate System** choose **orientation: Select Z axis/plane & X-Axis**



- Select the **face** of the tube as **Z axis**

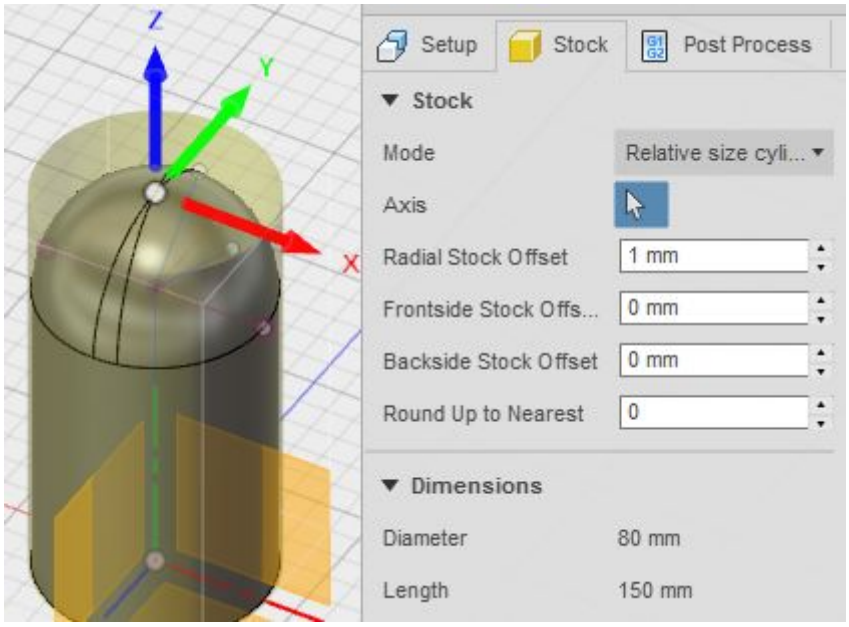


- Select the **body** as **Model**



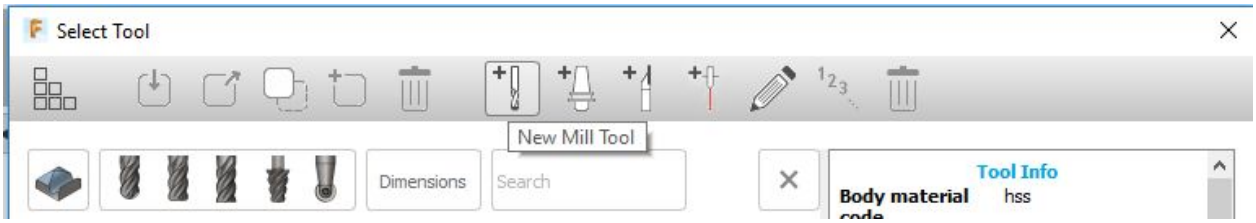
- Go to the **Stock tab** and select **Mode: Relative size cylinder**
- Choose **Radial Stock Offset** to be **1mm**
- Choose **Round Up To Nearest** to be **0**

The setup should now be similar to the image on the right

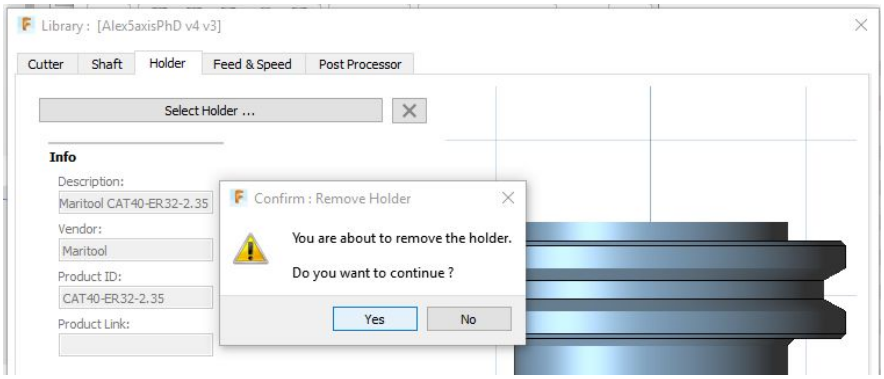


Step 17

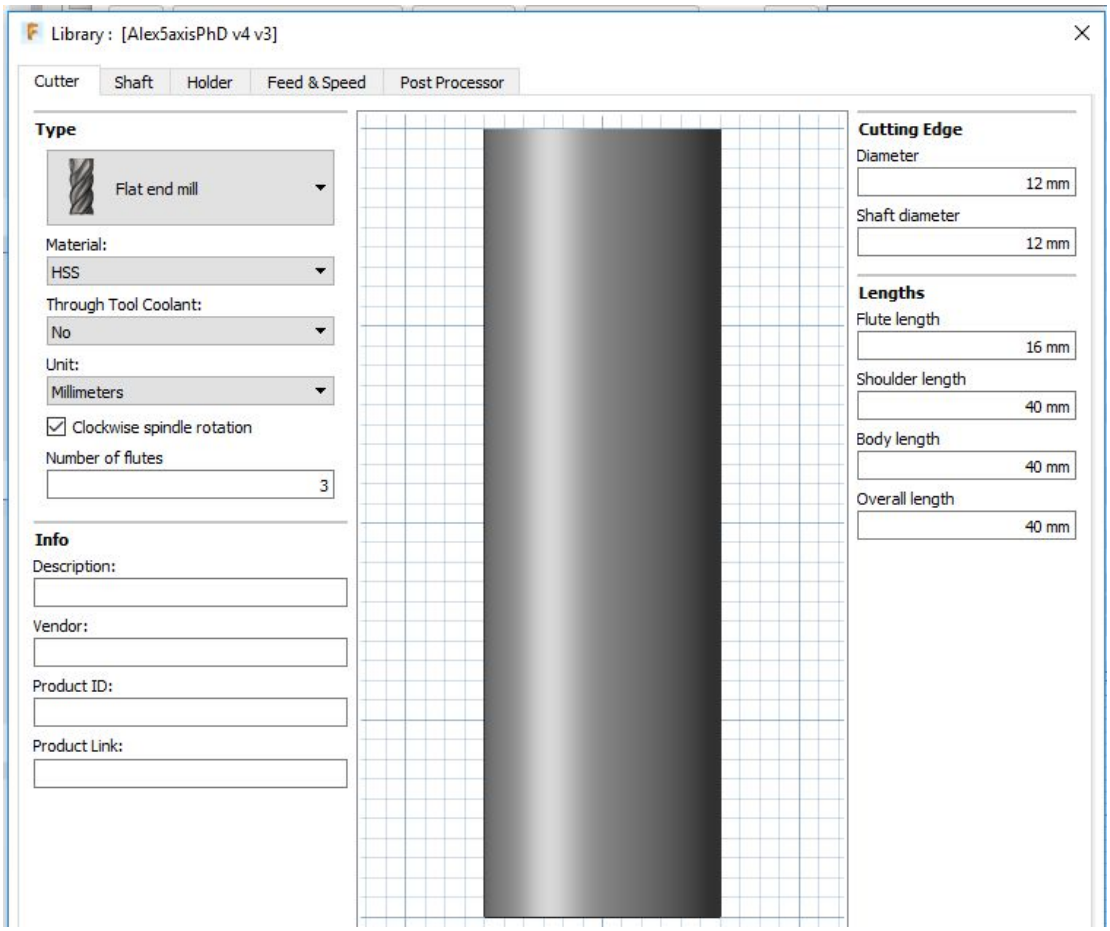
- From the top menu select **3D→Adaptive clearing**.
- From the pop up window press select tool



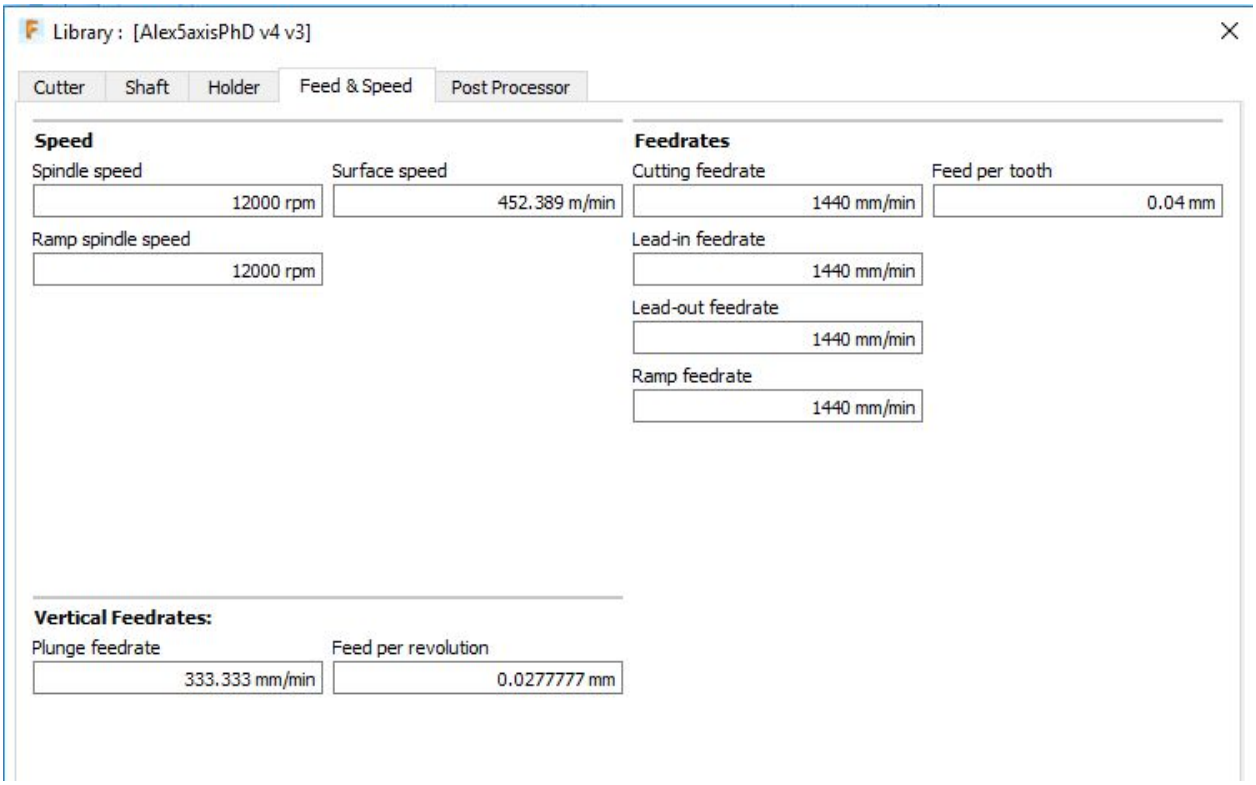
- Choose **new mill tool** from the top menu



- Go to the **Holder tab**
- Press the X button next to select holder and **remove** the holder.

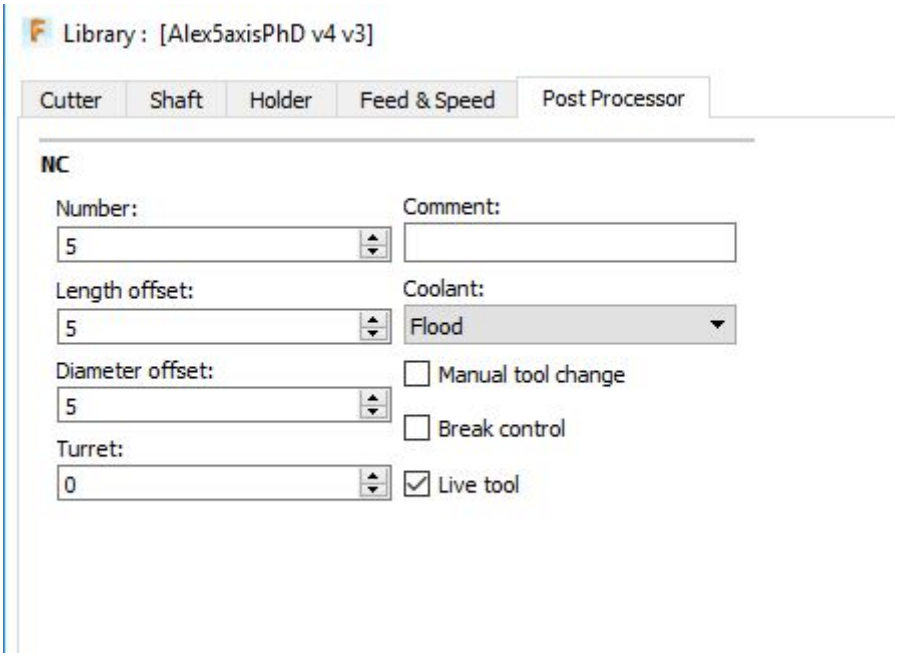


- Go to the **Cutter tab** and insert the values from the image above

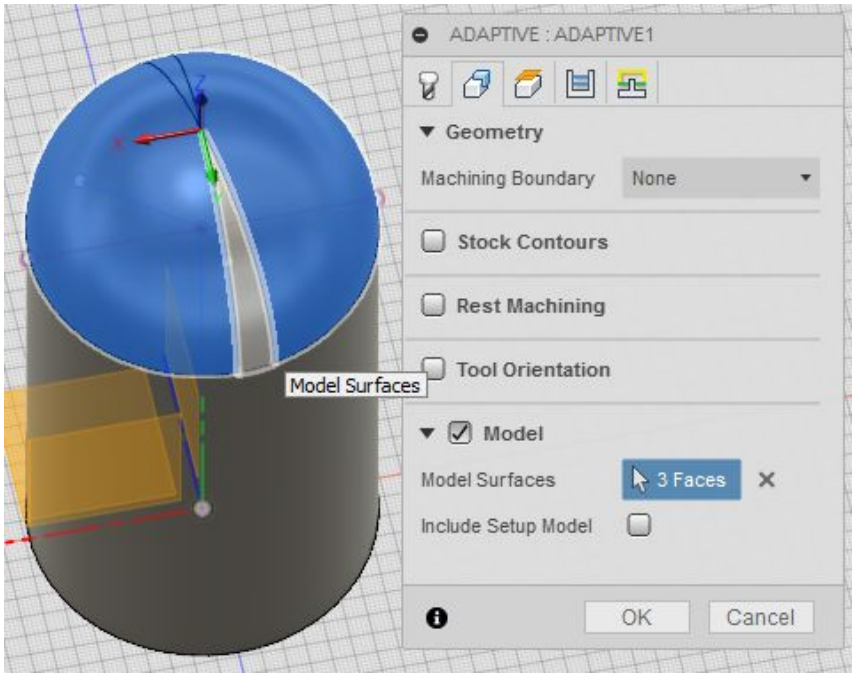


- Go to the **Feed & Speed tab** and insert the values from the image above

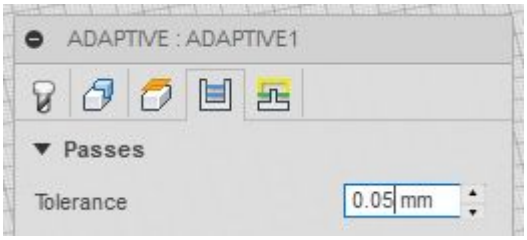
- Go to the **Post Processor tab** and insert the values from the image to the right
- Click OK and choose the tool



- In the Adaptive window go to the **Geometry tab**
- **Uncheck Stock contours** and **rest machining**
- Select the **4 faces** on the sphere as model surfaces, by clicking on the faces. Make sure the correct face is highlighted before clicking.
- **Uncheck** the **Include Setup model** box

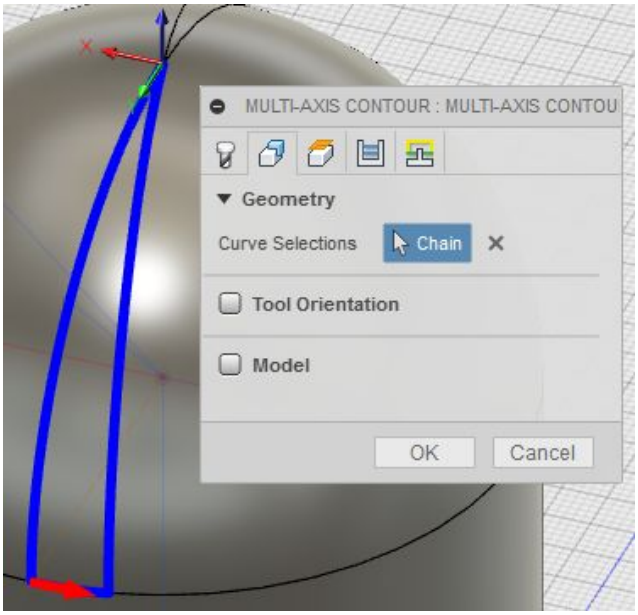


- Go to the **passes tab** and choose the Tolerance to be **0.05 mm** and click OK

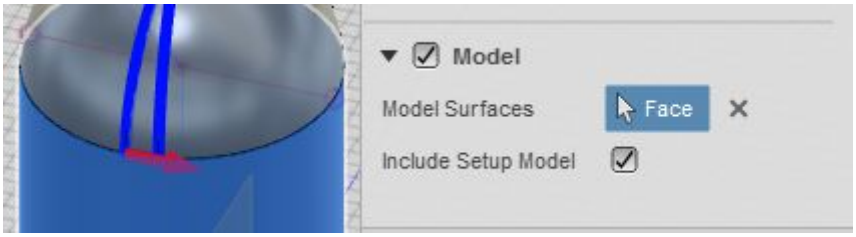


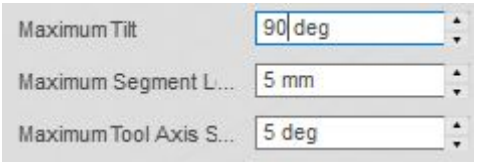
Step 18

- From the top menu menu choose **Multi-Axis → Multi-Axis Contour**
- Go to the **Geometry tab** and select the split face triangle as curve selection.



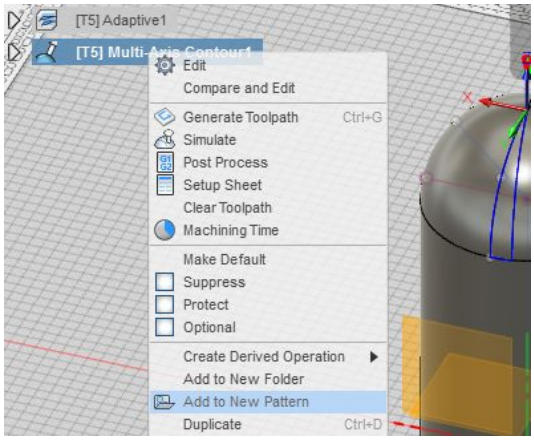
- **Check** the **model** box and select the cylinder face as model surface and include setup model. Click OK
- Go to the **Passes tab**
- set **Maximum Tilt** to **90 degrees**



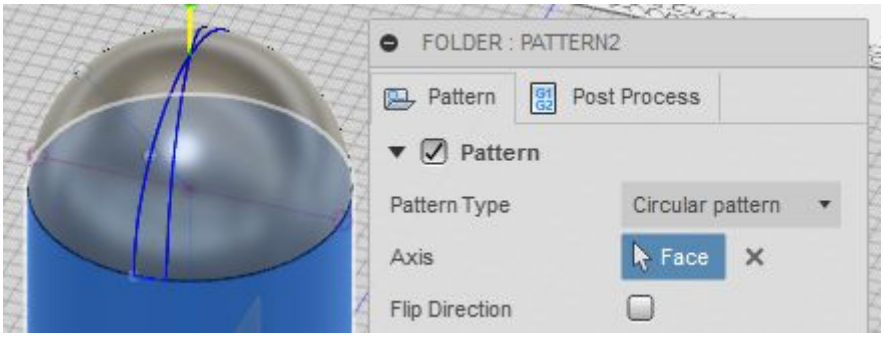


Step 19

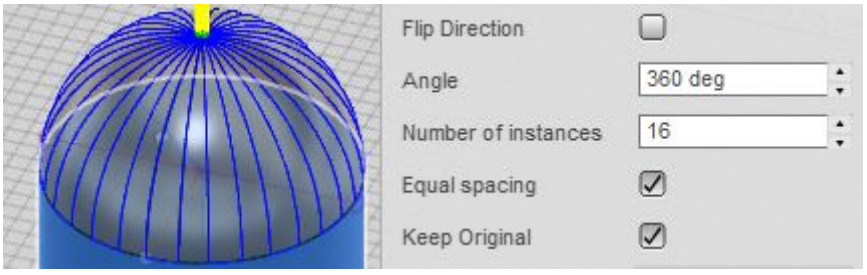
- Right click on the
- **Multi-Axis Contour** folder to the left and select **Add to new pattern**



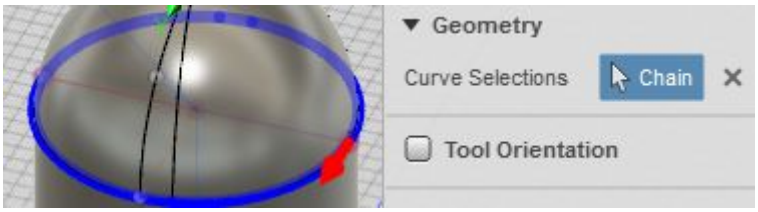
- Choose **pattern type: Circular pattern**
- Select the cylinder face as Axis



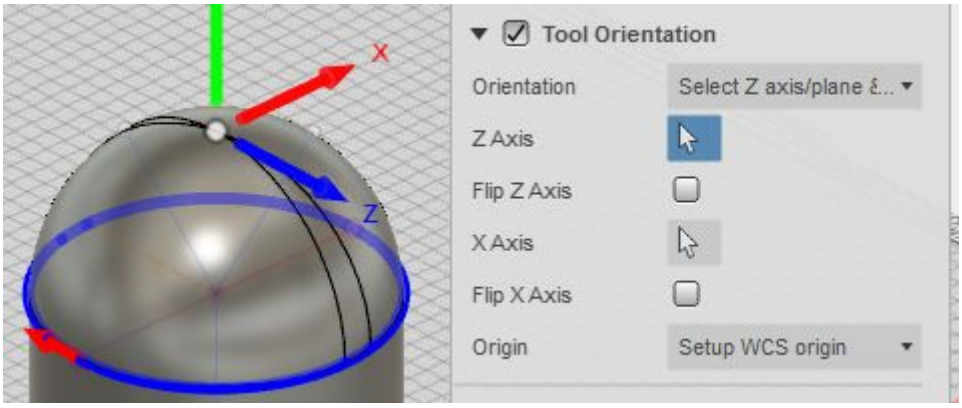
- Let the Angle be **360 Degrees**
- Set **Number of instances** to **16**. (15 would be enough to cover the full sphere, but we use 16 to get some overlap)



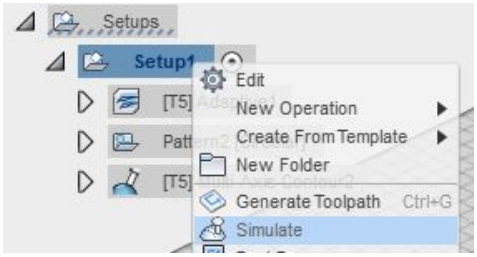
Step 20



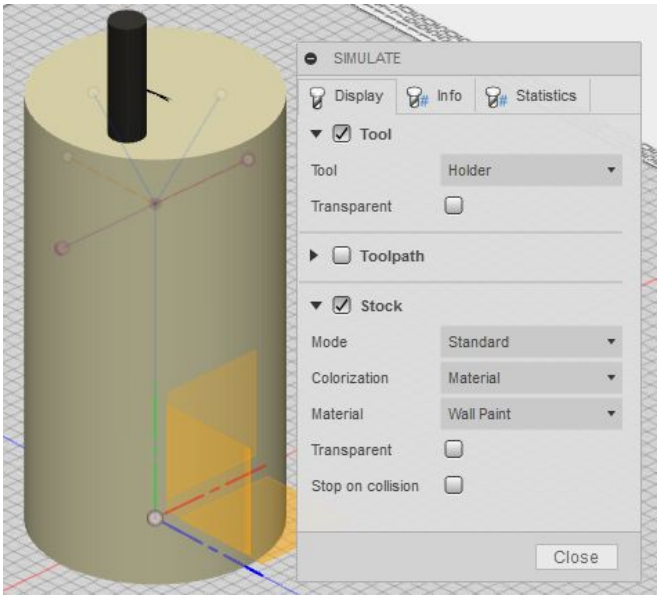
- From the top menu choose **Multi-Axis → Multi-Axis Contour**
- Go to the **Geometry tab** and select the chain in the image
- **Check** the **Tool orientation** box and if the axis are similar to the ones in the image below click OK



Step 21 Simulation

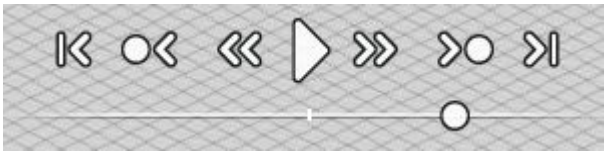


- Right click on the **Setup folder** to the right and choose simulate

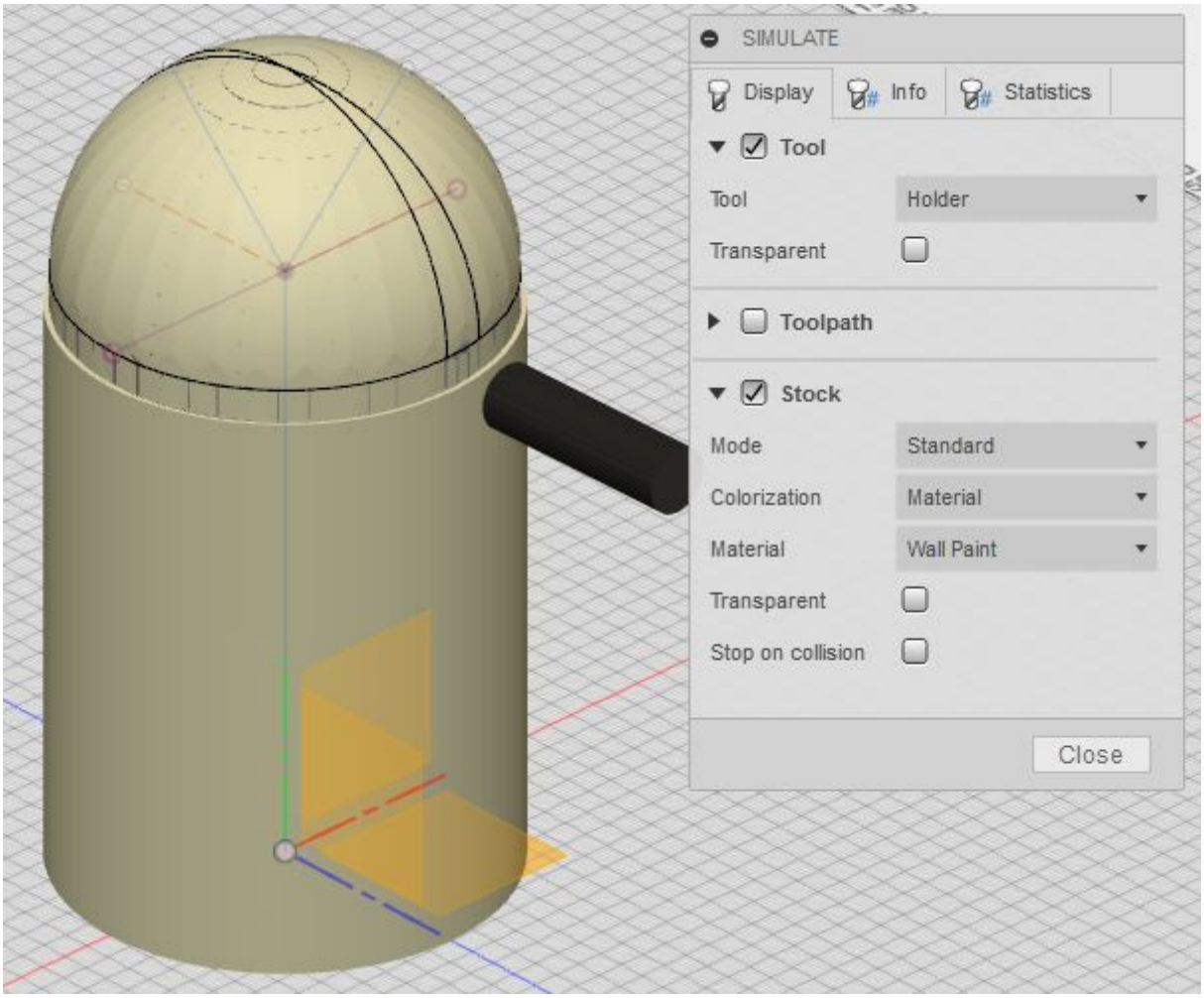


- **Uncheck** the **Toolpath** box
- **Check** the **Stock** box and if you wish to change the stock appearance settings you can change the colorization and material.

- Start the simulation by pressing the play button at the bottom of the screen and adjust the speed by dragging the speed control beneath the play button

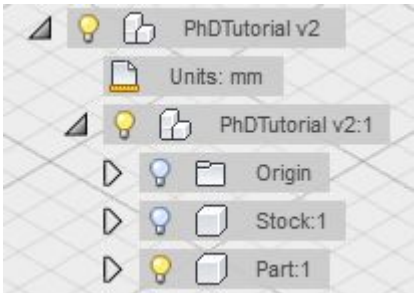


Once the simulation is over the model should be similar to the model below

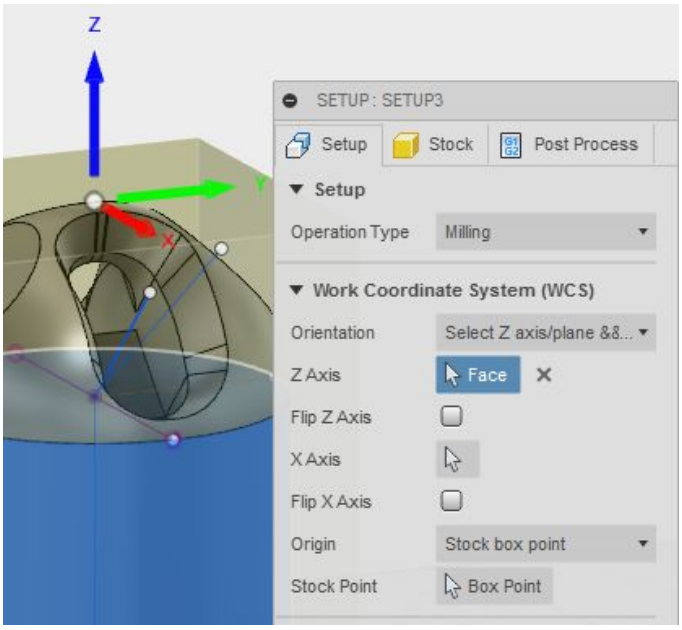


Step 22

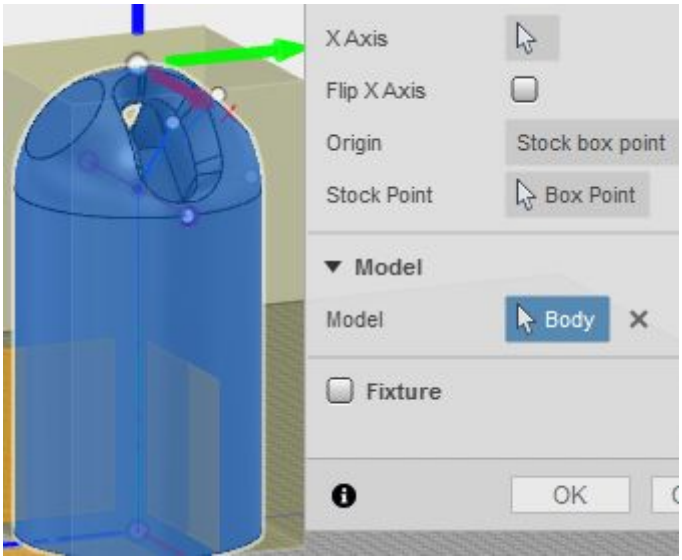
- Go to the component folder and turn **off** the **Stock:1** light bulb and turn **on Part:1**.



- From the top menu select **Setup→ New setup**
- Choose the face of the cylinder as **Z axis**

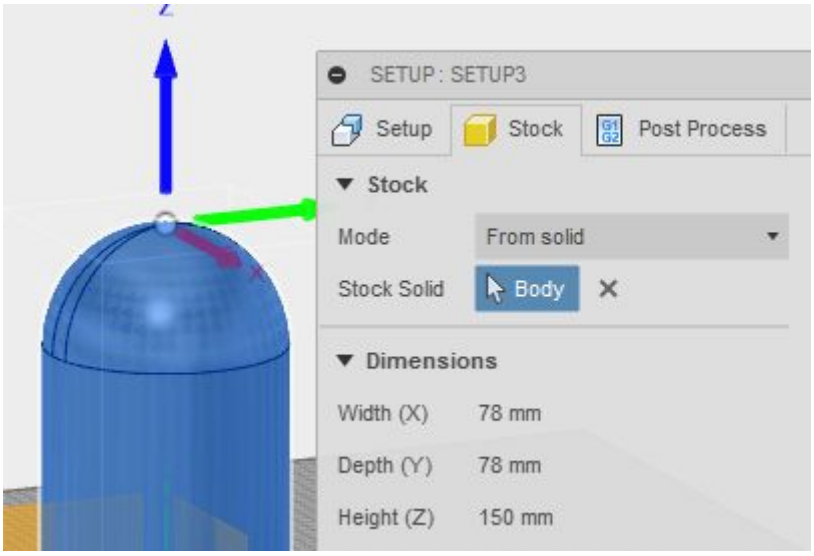


- Choose the body as model



- Go back and turn **on** the **Stock:1** Light bulb, and turn **off** the **Part:1**

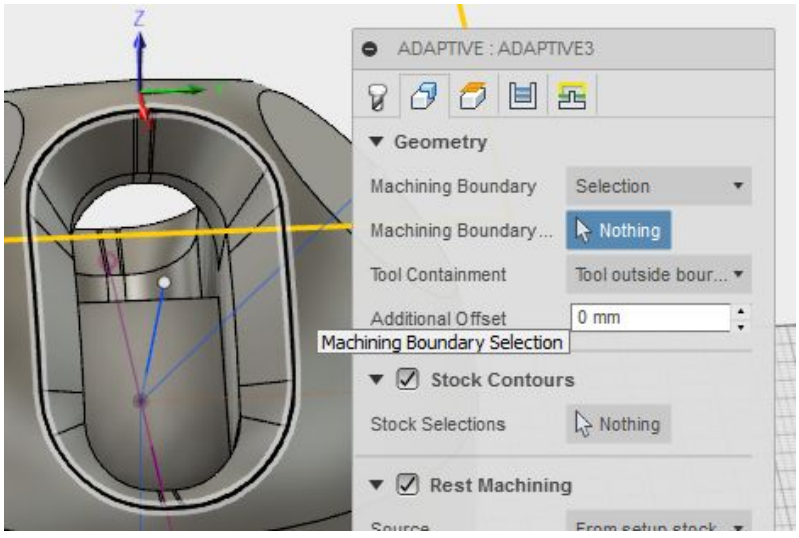
- Go to the **Stock tab** and select **Mode: From solid**
- Choose the body as Stock solid and click OK



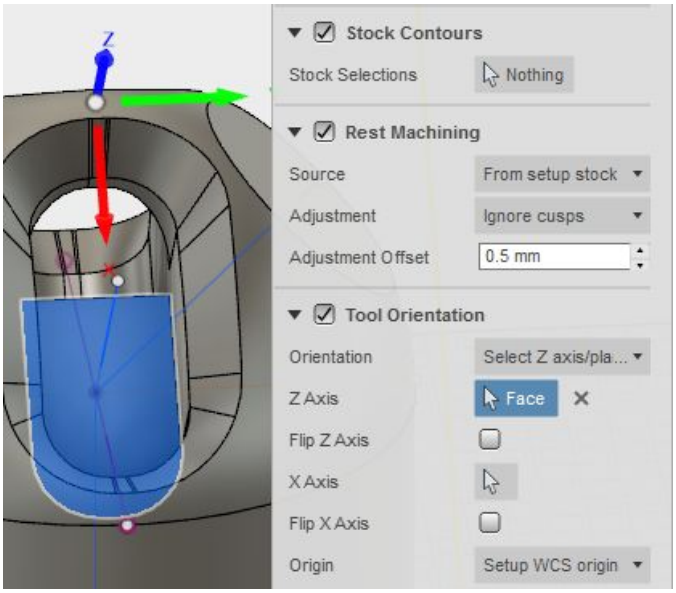
- Turn on the **Part:1** model again, and turn off Stock:1.

Step 23

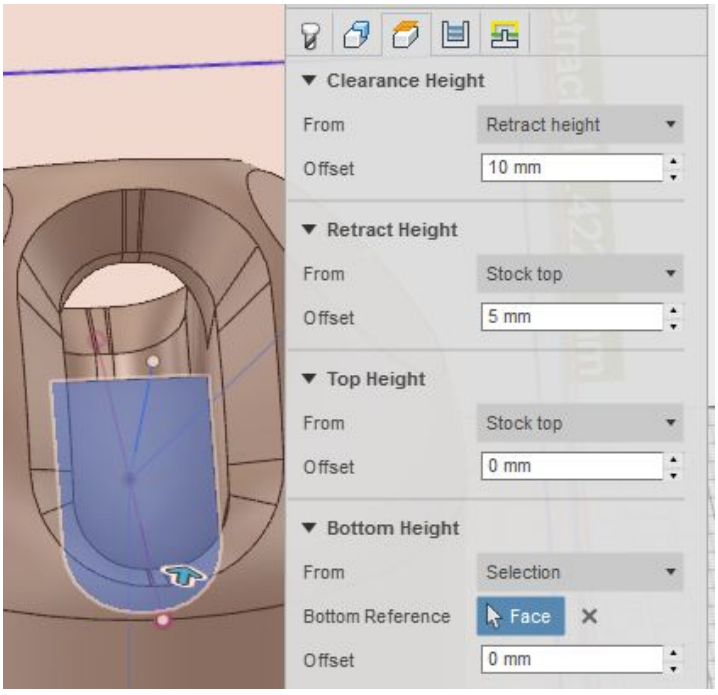
- From the top menu select **3D→Adaptive clearing**
- Go to the **Geometry tab**
- Select **Machining Boundary: Selection**
- Choose the edge shown in the image as selection
- Select tool containment: **Tool inside boundary**



- Go to tool orientation and select **Z axis** from the face at the bottom of the hole as shown in the image



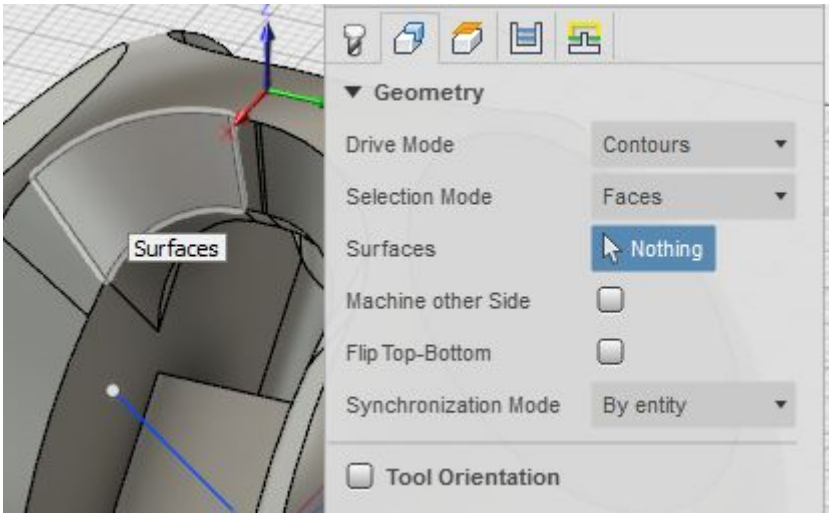
- Go to the **Heights tab** and go to **Bottom Height**
- Choose **From: Selection**
- Select the same face as bottom reference



- Go to the **Linking tab**
- Choose **Ramp Type: Plunge**

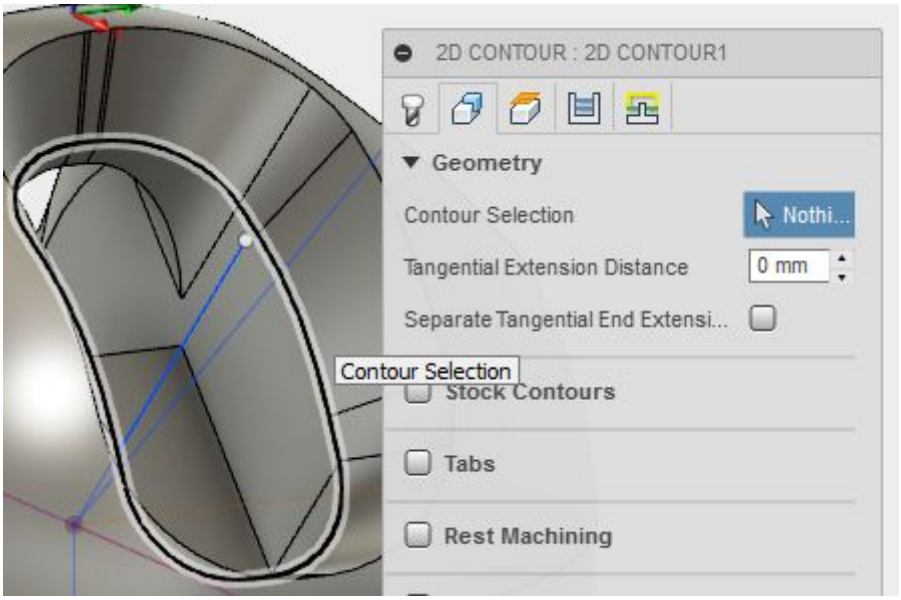
Step 24

- From the top menu select **Multi-Axis→ Swarf**
- Go to the **Geometry tab**
- Select one of the chamfered faces as surface and click OK

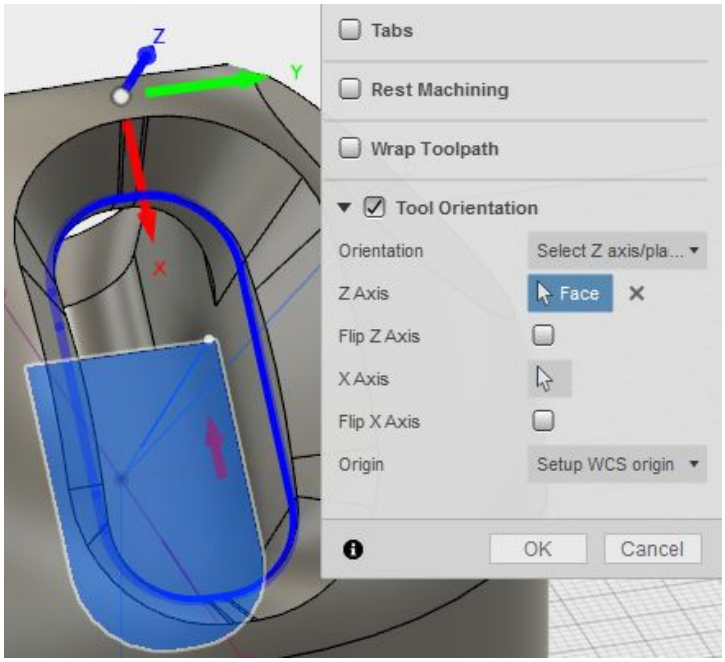


Step 25

- From the top menu go to **2D → Contour**
- Choose the **Geometry tab** and select the chain in the image as contour selection

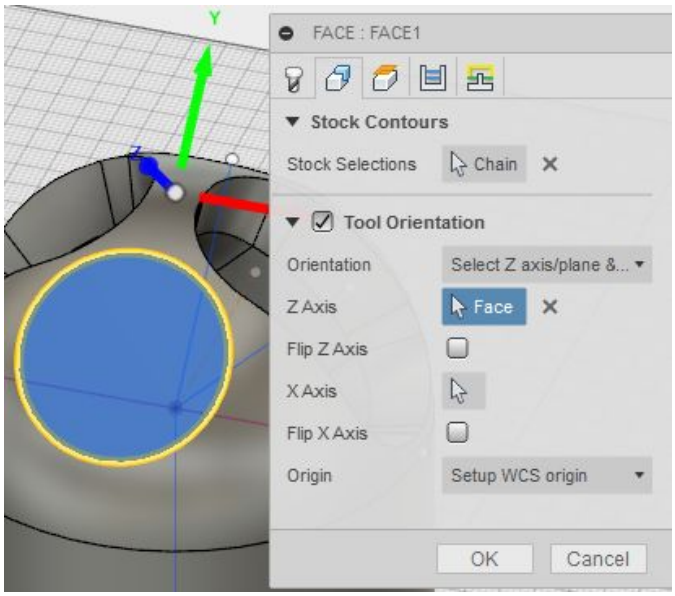


- Check the **tool orientation** box
- Select Z axis from the face highlighted in the image (The same face as in step 23)
- Go to the **Heights tab**
- Select **Bottom height: From selection**
- Choose the same face as the one highlighted in the image as Z axis for bottom reference and click OK



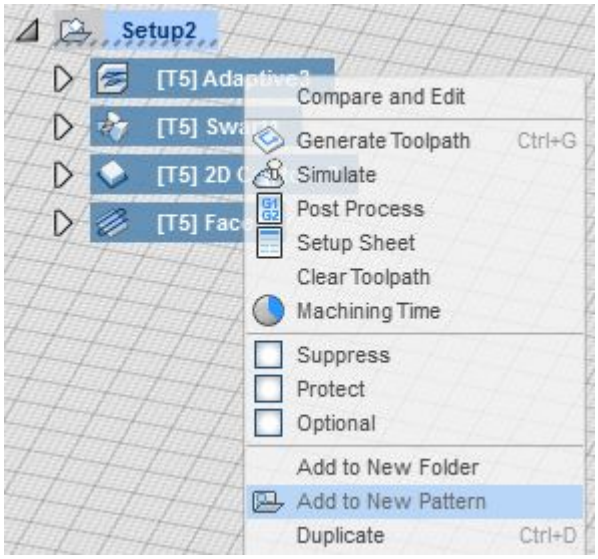
Step 26

- From the top menu select **2D→ Face**
- Go to the **Geometry tab**
- Select the yellow circle in the image as **Stock selection**
- Go to **tool orientation**
- Choose the face of the circle as Z axis and click OK

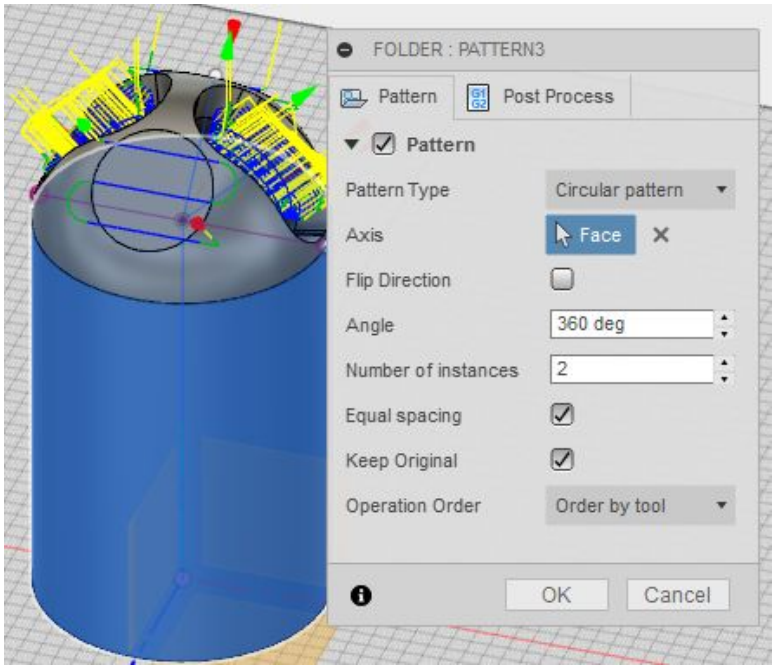


Step 27

- Select all the operations in the Setup2 folder (Hold the Ctrl key while clicking on each operation)
- Right click and select **Add to new pattern**

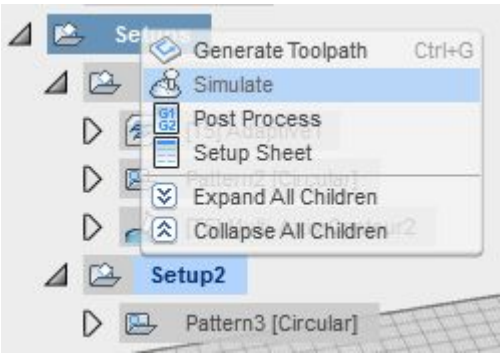


- Choose **Pattern Type: Circular pattern**
- Select the face of the cylinder as Axis
- Choose **number of instances** to be **2** and click OK

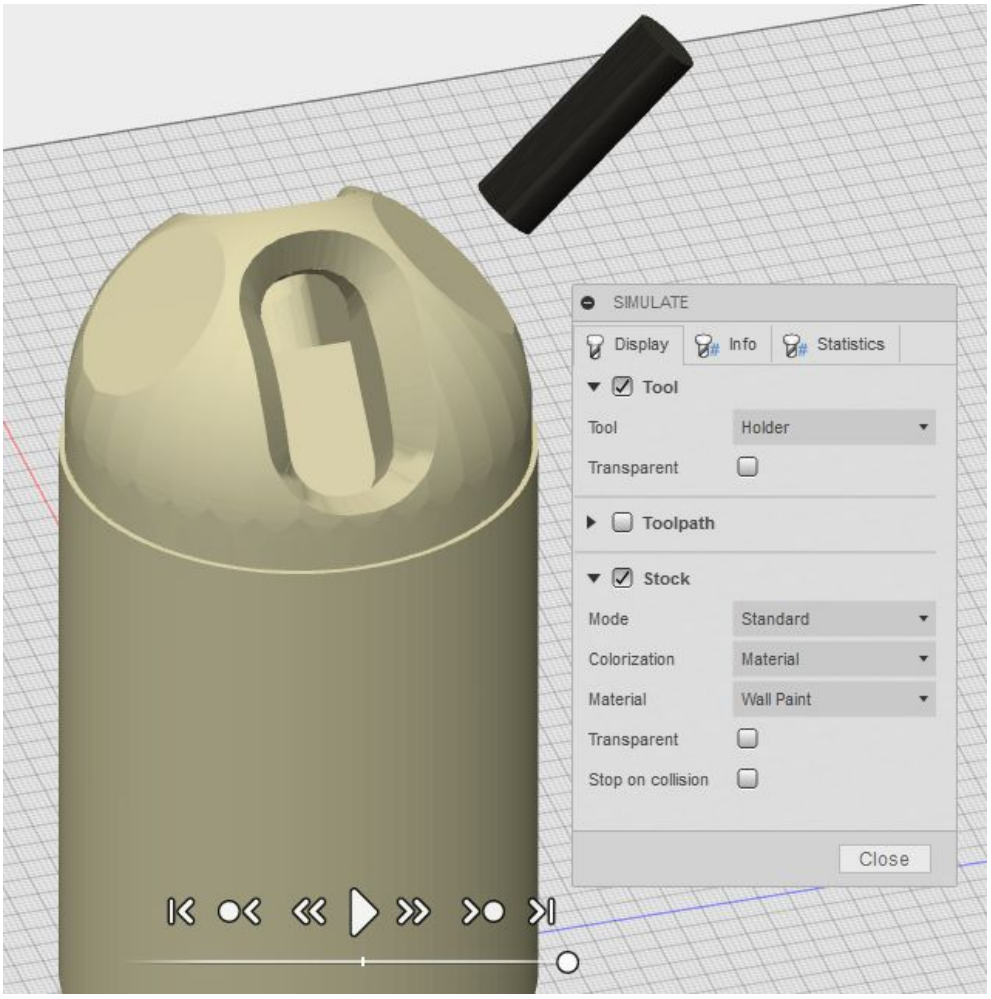


Step 28

- Right click on the **Setups** folder and select **Simulate**



- Start the simulation (To view the workpiece without the CAD model turn off the light bulb)
- When the simulation is finished, the model should be similar to the model in the image below



NTNU Valgrinda Workshop

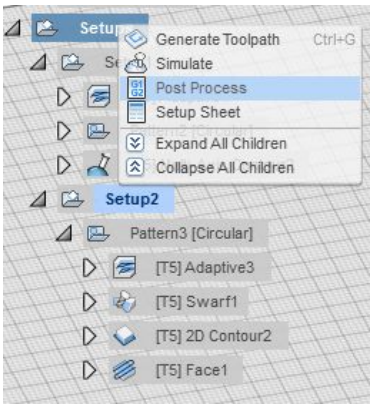
Step 29

- Insert the workpiece in a vise in the Deckel Maho 50 eVolution machine
- Use a probe to measure the offset values from the machine table origin (x = -250.658 and Y = -155.058 from the machine reference value)
- Calculate the offset values using the values from the probe measurement

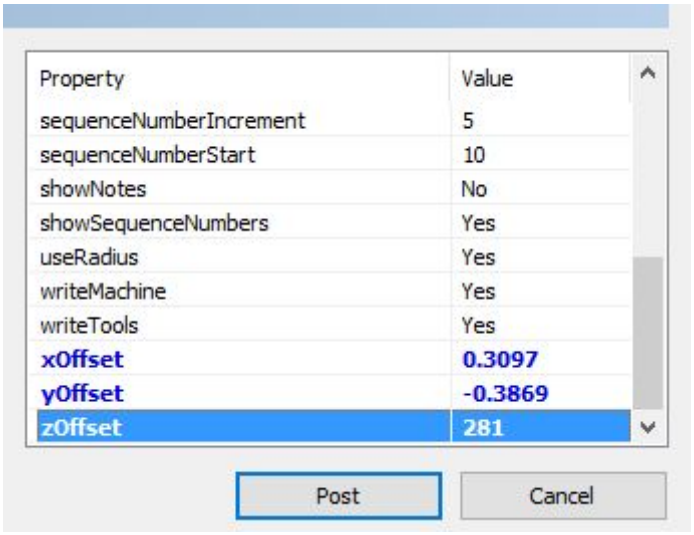
	Negative Value	Positive Value	Abs(negative-positive)/2	Negative Value + Previous Ans
X	-39.4726	40.092	39.7823	0.3097
Y	-40.1668	39.3931	39.77995	-0.38685
Z				281

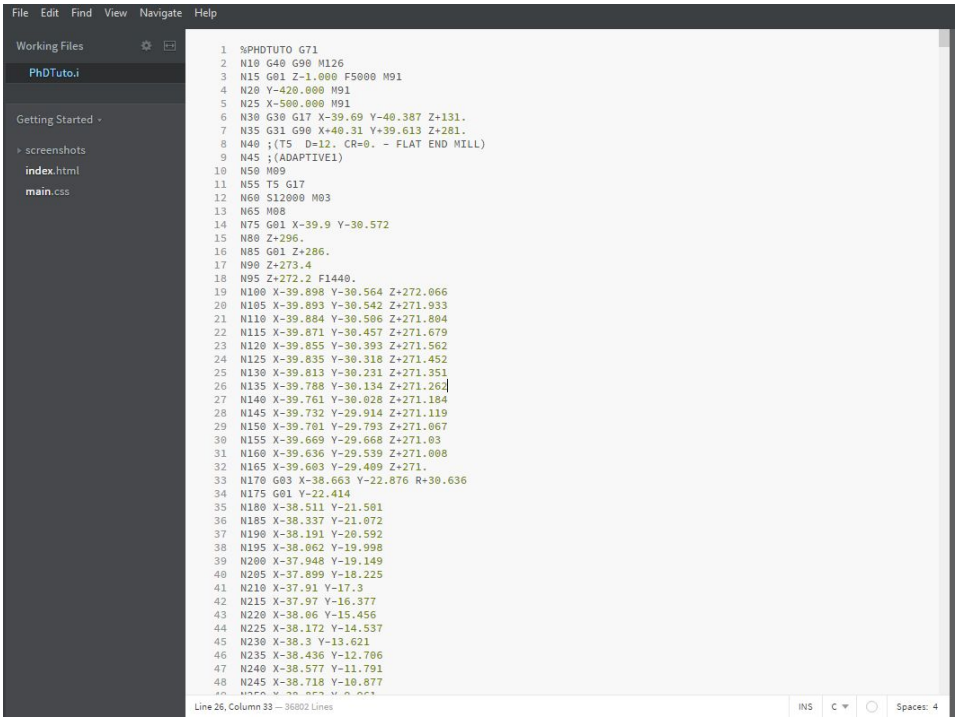
Step 30 Post processing in Fusion 360

- Right click on the **Setups** folder in Fusion
- Select **Post Process**
- Click on the three dots button to find the Configuration Folder where the post processor file **AADM50** is stored



- Insert the offset values (from the calculation of the probe values in step 29) in the **Property section**
- Name the program and click Post
- The program is now ready to be uploaded to the machine.





Appendix F

CNC-kode med og uten linearisering-og singularitetskontroll

F.1 CNC-kode med linearisering-og singularitetskontrol

Av plasshensyn er kodene kortet ned og erstattet med "..." der det ikke går utover verdifull kode.

```
%MEDSING3 G71
N10 ;(Offset values: X: 0.2945 ,Y:  0.646 ,Z:  228 )
N15 G40 G90 M126
N20 G01 Z-1.000 F5000 M91
N25 Y-420.000 M91
N30 X-500.000 M91
N35 G30 G17 X-39.706 Y-39.354 Z+96.5
N40 G31 G90 X+40.294 Y+40.646 Z+228.
N45 ;(T5  D=12. CR=0. - FLAT END MILL)
N50 ;(MULTI-AXIS CONTOUR3)
N55 M09
N60 T5 G17
N65 S12000 M03
N80 G01 Z+243.
N85 X+20.293 Y+36.816
N90 G01 X-35.117 Y-48.845 Z+231.761 B+88.456 C+124.589 F2000.
N95 X-10.557 Y-31.941 Z+213.92
N100 X-10.416 Y-31.843 Z+213.795 F1080.
N105 X-10.294 Y-31.76 Z+213.642
N110 X-10.197 Y-31.693 Z+213.465
N115 X-10.127 Y-31.645 Z+213.27
N120 X-10.087 Y-31.617 Z+213.063
N125 X-10.077 Y-31.61 Z+212.851
...
N155 X-10.614 Y-31.979 Z+211.822
```

```
N160 X-10.776 Y-32.091 Z+211.743
N165 X-10.948 Y-32.209 Z+211.699
N170 X-10.987 Y-32.199 Z+211.709 C+124.54 //Lineariseringskontroll slaar
    inn
N175 X-11.109 Y-31.84 Z+212.171 B+86.742 C+123.757
N180 X-11.232 Y-31.482 Z+212.633 B+85.029 C+122.975
N185 X-11.055 Y-31.082 Z+212.976 B+83.717 C+122.426
N190 X-10.877 Y-30.683 Z+213.32 B+82.405 C+121.877
N195 X-11.48 Y-30.895 Z+213.451 B+81.98 C+121.601
N200 X-11.282 Y-30.5 Z+213.785 B+80.713 C+121.059
N205 X-11.084 Y-30.105 Z+214.118 B+79.447 C+120.517
N210 X-11.599 Y-30.482 Z+213.995 B+79.937 C+120.679
N215 X-11.673 Y-30.311 Z+214.247 B+79.007 C+120.263
N220 X-11.74 Y-30.025 Z+214.634 B+77.544 C+119.619
N225 X-11.807 Y-29.739 Z+215.02 B+76.082 C+118.974
N230 X-11.746 Y-29.35 Z+215.467 B+74.371 C+118.233
N235 X-11.684 Y-28.961 Z+215.913 B+72.659 C+117.492
N240 X-11.69 Y-28.688 Z+216.275 B+71.24 C+116.889
N245 X-11.697 Y-28.415 Z+216.638 B+69.82 C+116.286
N250 X-11.677 Y-28.146 Z+216.99 B+68.415 C+115.695
N255 X-11.657 Y-27.876 Z+217.342 B+67.009 C+115.105
N260 X-11.612 Y-27.61 Z+217.684 B+65.618 C+114.526
N265 X-11.567 Y-27.345 Z+218.025 B+64.226 C+113.947
N270 X-11.499 Y-27.086 Z+218.356 B+62.846 C+113.381
N275 X-11.43 Y-26.827 Z+218.686 B+61.467 C+112.815
N280 X-11.34 Y-26.576 Z+219.004 B+60.098 C+112.262
N285 X-11.249 Y-26.324 Z+219.322 B+58.73 C+111.71
N290 X-11.137 Y-26.08 Z+219.629 B+57.372 C+111.168
N295 X-11.025 Y-25.836 Z+219.935 B+56.014 C+110.626
N300 X-10.893 Y-25.599 Z+220.229 B+54.665 C+110.093
N305 X-10.76 Y-25.362 Z+220.523 B+53.317 C+109.56
N310 X-10.609 Y-25.134 Z+220.805 B+51.977 C+109.036
N315 X-10.457 Y-24.905 Z+221.087 B+50.638 C+108.512
N320 X-10.287 Y-24.685 Z+221.356 B+49.306 C+107.997
N325 X-10.118 Y-24.465 Z+221.625 B+47.975 C+107.482
N330 X-9.931 Y-24.255 Z+221.881 B+46.651 C+106.974
N335 X-9.744 Y-24.044 Z+222.137 B+45.327 C+106.465//Lineariseringskontroll
    avsluttes
N340 X-9.337 Y-23.641 Z+222.623 B+42.692 C+105.463
N345 X-8.901 Y-23.258 Z+223.082 B+40.071 C+104.473
N350 X-8.435 Y-22.896 Z+223.513 B+37.46 C+103.495
N355 X-7.943 Y-22.556 Z+223.916 B+34.861 C+102.529
N360 X-7.427 Y-22.238 Z+224.291 B+32.271 C+101.575
...
N400 X-2.576 Y-20.538 Z+226.239 B+11.715 C+94.182
```

```
N405 X-1.859 Y-20.435 Z+226.349 B+9.089 C+93.26
N410 X-1.137 Y-20.36 Z+226.426 B+6.466 C+92.341
N415 X-0.42 Y-20.314 Z+226.469 B+3.844 C+91.451
N420 X+0.232 Y-20.3 Z+226.48 B+1.223 C+90.752
N425 X+0.484 Z+226.479 B+0.865 C+90.68 //Singularitetskontrollen slaar inn
N430 X-0.558 Y-20.327 Z+226.471 B+0.014 C+95.131
N435 X-0.714 Y-20.322 C+95.569
N440 X-0.87 Y-20.316 Z+226.47 B+0.017 C+96.006
N445 X-1.026 Y-20.309 C+96.444
N450 X-1.182 Y-20.301 B+0.02 C+96.882
N455 X-1.338 Y-20.291 C+97.32
N460 X-1.494 Y-20.281 B+0.022 C+97.757
N465 X-1.65 Y-20.269 C+98.195
N470 X-1.805 Y-20.256 B+0.025 C+98.633
N475 X-1.961 Y-20.242 C+99.07
...
N515 X-3.201 Y-20.086 C+102.572
N520 X-3.355 Y-20.061 B+0.039 C+103.01
...
N535 X-3.816 Y-19.98 C+104.323
N540 X-3.97 Y-19.951 Z+226.468 B+0.045 C+104.761
...
N560 X-4.581 Y-19.822 B+0.05 C+106.511
N565 X-4.733 Y-19.787 C+106.949
...
N645 X-7.125 Y-19.069 C+113.952
N650 X-7.272 Y-19.015 B+0.075 C+114.39
...
N665 X-7.708 Y-18.845 C+115.703
N670 X-7.853 Y-18.787 B+0.081 C+116.141
...
N720 X-9.273 Y-18.139 B+0.095 C+120.518
N725 X-9.412 Y-18.069 Z+226.459 C+120.956
...
N780 X-10.906 Y-17.224 B+0.112 C+125.771
N785 X-11.039 Y-17.141 Z+226.455 C+126.208
...
N805 X-11.561 Y-16.8 C+127.959
N810 X-11.69 Y-16.712 Z+226.453 B+0.12 C+128.397
...
N860 X-12.941 Y-15.782 Z+226.449 B+0.134 C+132.774
N865 X-13.062 Y-15.683 C+133.212
...
N885 X-13.539 Y-15.282 Z+226.447 C+134.962
N890 X-13.657 Y-15.179 B+0.143 C+135.4
```

```
...
N955 X-15.108 Y-13.766 C+141.09
N960 X-15.214 Y-13.651 Z+226.44 B+0.162 C+141.528
...
N985 X-15.729 Y-13.067 C+143.717
N990 X-15.829 Y-12.948 Z+226.437 B+0.17 C+144.154
...
N1040 X-16.782 Y-11.717 B+0.184 C+148.531
N1045 X-16.872 Y-11.59 Z+226.432 C+148.969
...
N1080 X-17.475 Y-10.682 B+0.196 C+152.033
N1085 X-17.557 Y-10.55 Z+226.428 C+152.471
...
N1190 X-19.042 Y-7.644 B+0.226 C+161.663
N1195 X-19.101 Y-7.501 Z+226.417 C+162.1
...
N1215 X-19.326 Y-6.921 Z+226.415 C+163.851
N1220 X-19.38 Y-6.775 B+0.235 C+164.289
...
N1260 X-19.767 Y-5.594 B+0.246 C+167.791
N1265 X-19.81 Y-5.445 Z+226.41 C+168.228
...
N1290 X-20.009 Y-4.695 Z+226.408 B+0.254 C+170.417
N1295 X-20.046 Y-4.544 C+170.855
...
N1370 X-20.453 Y-2.253 Z+226.401 B+0.277 C+177.42
N1375 X-20.471 Y-2.099 C+177.858
...
N1390 X-20.517 Y-1.636 B+0.282 C+179.171
N1395 X-20.53 Y-1.481 Z+226.399 C+179.609
...
N1440 X-20.595 Y-0.088 B+0.296 C+183.548
N1445 X-20.597 Y+0.067 Z+226.395 C+183.986
...
N1570 X-20.246 Y+3.917 B+0.333 C+194.929
N1575 X-20.217 Y+4.069 C+195.366
...
N1620 X-19.902 Y+5.425 B+0.347 C+199.306
N1625 X-19.862 Y+5.575 C+199.743
...
N1735 X-18.684 Y+8.76 C+209.373
N1740 X-18.618 Y+8.9 B+0.38 C+209.811
...
N1755 X-18.413 Y+9.315 C+211.124
N1760 X-18.343 Y+9.453 B+0.386 C+211.561
```

```
...
N1815 X-17.502 Y+10.927 C+216.376
N1820 X-17.419 Y+11.058 B+0.402 C+216.814
...
N1920 X-15.567 Y+13.521 B+0.43 C+225.568
N1925 X-15.464 Y+13.636 C+226.006
N1930 X-15.361 Y+13.751 Z+226.39 B+0.433 C+226.444
...
N1965 X-14.615 Y+14.53 C+229.508
N1970 X-14.505 Y+14.638 B+0.444 C+229.945
N1975 X-14.394 Y+14.745 Z+226.393 C+230.383
...
N2075 X-12.017 Y+16.7 Z+226.402 C+239.137
N2080 X-11.891 Y+16.788 B+0.475 C+239.575
...
N2110 X-11.119 Y+17.295 Z+226.406 B+0.484 C+242.201
N2115 X-10.988 Y+17.376 C+242.639
N2120 X-10.856 Y+17.456 Z+226.407 B+0.486 C+243.076
N2125 X-10.724 Y+17.535 Z+226.408 C+243.514
...
N2215 X-8.251 Y+18.778 Z+226.42 C+251.393
N2220 X-8.109 Y+18.837 Z+226.421 B+0.514 C+251.831
N2225 X-7.967 Y+18.894 Z+226.422 C+252.268
N2230 X-7.824 Y+18.951 Z+226.423 B+0.517 C+252.706
...
N2250 X-7.248 Y+19.167 Z+226.426 B+0.523 C+254.457
N2255 X-7.102 Y+19.218 Z+226.427 C+254.895
N2260 X-6.957 Y+19.268 B+0.525 C+255.332
...
N2340 X-4.584 Y+19.915 Z+226.442 B+0.548 C+262.336
N2345 X-4.434 Y+19.945 Z+226.443 C+262.773
N2350 X-4.283 Y+19.975 Z+226.444 B+0.551 C+263.211
...
N2370 X-3.677 Y+20.081 Z+226.448 B+0.556 C+264.962
N2375 X-3.525 Y+20.105 Z+226.449 C+265.4
N2380 X-3.373 Y+20.128 Z+226.45 B+0.559 C+265.837
N2385 X-3.221 Y+20.149 Z+226.451 C+266.275
N2390 X-3.068 Y+20.169 Z+226.452 B+0.562 C+266.713
N2395 X-2.916 Y+20.188 Z+226.453 C+267.15
N2400 X-2.763 Y+20.206 Z+226.454 B+0.565 C+267.588
N2405 X-2.61 Y+20.223 Z+226.455 C+268.026
N2410 X-2.457 Y+20.238 Z+226.456 B+0.567 C+268.464
N2415 X-2.304 Y+20.252 Z+226.457 C+268.901
N2420 X-2.151 Y+20.266 Z+226.458 B+0.57 C+269.339
N2425 X-1.997 Y+20.278 Z+226.459 C+269.777
```

```
204 N2430 X-1.844 Y+20.288 Z+226.46 B+0.573 C+270.214//
        Singularitetskontrollen avsluttes
205 N2435 X-2.346 Y+20.26 Z+226.432 B+2.296 C+270.774
206 N2440 X-2.848 Y+20.232 Z+226.403 B+4.019 C+271.333
207 N2445 X-4.219 Y+20.141 Z+226.315 B+6.64 C+272.296
208 N2450 X-5.593 Y+20.01 Z+226.195 B+9.263 C+273.237
209 N2455 X-6.875 Y+19.849 Z+226.047 B+11.802 C+274.147
210 N2460 X-8.15 Y+19.653 Z+225.869 B+14.343 C+275.054
211 ...
212 N2500 X-17.89 Y+16.82 Z+223.382 B+34.861 C+282.505
213 N2505 X-19.027 Y+16.312 Z+222.942 B+37.46 C+283.47
214 N2510 X-20.142 Y+15.769 Z+222.474 B+40.071 C+284.445
215 N2515 X-21.232 Y+15.193 Z+221.98 B+42.692 C+285.433
216 N2520 X-22.296 Y+14.583 Z+221.459 B+45.327 C+286.434//
        Lineariseringskontrollen slaar inn
217 N2525 X-22.814 Y+14.262 Z+221.186 B+46.651 C+286.942
218 N2530 X-23.332 Y+13.941 Z+220.913 B+47.975 C+287.449
219 N2535 X-23.836 Y+13.603 Z+220.628 B+49.306 C+287.964
220 N2540 X-24.34 Y+13.265 Z+220.342 B+50.638 C+288.479
221 N2545 X-24.829 Y+12.91 Z+220.044 B+51.977 C+289.003
222 N2550 X-25.317 Y+12.556 Z+219.745 B+53.317 C+289.526
223 N2555 X-25.789 Y+12.185 Z+219.435 B+54.665 C+290.059
224 N2560 X-26.262 Y+11.814 Z+219.125 B+56.014 C+290.592
225 N2565 X-26.716 Y+11.427 Z+218.803 B+57.372 C+291.135
226 N2570 X-27.171 Y+11.039 Z+218.481 B+58.73 C+291.678
227 N2575 X-27.609 Y+10.636 Z+218.148 B+60.098 C+292.232
228 N2580 X-28.047 Y+10.232 Z+217.815 B+61.467 C+292.786
229 N2585 X-28.466 Y+9.813 Z+217.47 B+62.846 C+293.349
230 N2590 X-28.886 Y+9.394 Z+217.124 B+64.226 C+293.912
231 N2595 X-29.287 Y+8.958 Z+216.769 B+65.618 C+294.487
232 N2600 X-29.688 Y+8.523 Z+216.413 B+67.009 C+295.062
233 N2605 X-30.068 Y+8.071 Z+216.047 B+68.415 C+295.651
234 N2610 X-30.448 Y+7.619 Z+215.681 B+69.82 C+296.24
235 N2615 X-30.806 Y+7.151 Z+215.305 B+71.24 C+296.846
236 N2620 X-31.164 Y+6.682 Z+214.929 B+72.659 C+297.452
237 N2625 X-31.667 Y+6.045 Z+214.468 B+74.371 C+298.193
238 N2630 X-32.169 Y+5.408 Z+214.006 B+76.082 C+298.933
239 N2635 X-32.481 Y+4.898 Z+213.608 B+77.544 C+299.576
240 N2640 X-32.793 Y+4.389 Z+213.209 B+79.007 C+300.218
241 N2645 X-32.961 Y+4.072 Z+212.949 B+79.937 C+300.629
242 N2650 X-32.351 Y+4.583 Z+213.076 B+79.447 C+300.355
243 N2655 X-32.863 Y+3.958 Z+212.733 B+80.713 C+300.946
244 N2660 X-33.375 Y+3.333 Z+212.389 B+81.98 C+301.537
245 N2665 X-32.935 Y+3.557 Z+212.255 B+82.405 C+301.645
246 N2670 X-33.424 Y+2.886 Z+211.901 B+83.717 C+302.289
```

```
N2675 X-33.913 Y+2.215 Z+211.548 B+85.029 C+302.934
N2680 X-34.229 Y+1.537 Z+211.075 B+86.742 C+303.737
N2685 X-34.546 Y+0.859 Z+210.601 B+88.456 C+304.54
N2690 X-34.54 Y+0.898 Z+210.59 C+304.49
N2695 X-34.369 Y+1.016 Z+210.634
N2700 X-34.207 Y+1.128 Z+210.714
N2705 X-34.059 Y+1.23 Z+210.827
N2710 X-33.93 Y+1.319 Z+210.97
N2715 X-33.823 Y+1.392 Z+211.139
N2720 X-33.743 Y+1.448 Z+211.328
N2725 X-33.691 Y+1.483 Z+211.531
N2730 X-33.67 Y+1.498 Z+211.742
N2735 X-33.68 Y+1.491 Z+211.955
N2740 X-33.721 Y+1.463 Z+212.161
N2745 X-33.79 Y+1.415 Z+212.356
N2750 X-33.888 Y+1.348 Z+212.533
N2755 X-34.009 Y+1.264 Z+212.686
N2760 X-34.15 Y+1.167 Z+212.811
N2765 X-58.71 Y-15.738 Z+230.652 F2000.
N2775 M09
N2780 M30
N2785 %medsing3 G71
```

F.2 CNC-kode uten linearisering-og singularitetskontrol

```
%UTENSING3 G71
N10 ;(Offset values: X: 0.2945 ,Y:  0.646 ,Z:  228 )
N15 G40 G90 M126
N20 G01 Z-1.000 F5000 M91
N25 Y-420.000 M91
N30 X-500.000 M91
N35 G30 G17 X-39.706 Y-39.354 Z+96.5
N40 G31 G90 X+40.294 Y+40.646 Z+228.
N45 ;(T5  D=12. CR=0. - FLAT END MILL)
N50 ;(MULTI-AXIS CONTOUR3)
N55 M09
N60 T5 G17
N65 S12000 M03
N70 M08
N80 G01 Z+243.
N85 X+20.293 Y+36.816
N90 G01 X-35.117 Y-48.845 Z+231.761 B+88.456 C+124.589 F2000.
```

```
N95 X-10.557 Y-31.941 Z+213.92
N100 X-10.416 Y-31.843 Z+213.795 F1080.
N105 X-10.294 Y-31.76 Z+213.642
N110 X-10.197 Y-31.693 Z+213.465
N115 X-10.127 Y-31.645 Z+213.27
N120 X-10.087 Y-31.617 Z+213.063
N125 X-10.077 Y-31.61 Z+212.851
N130 X-10.098 Y-31.625 Z+212.64
N135 X-10.15 Y-31.66 Z+212.437
N140 X-10.23 Y-31.715 Z+212.248
N145 X-10.336 Y-31.789 Z+212.079
N150 X-10.466 Y-31.878 Z+211.936
N155 X-10.614 Y-31.979 Z+211.822
N160 X-10.776 Y-32.091 Z+211.743
N165 X-10.948 Y-32.209 Z+211.699
N170 X-10.987 Y-32.199 Z+211.709 C+124.54
N175 X-11.232 Y-31.482 Z+212.633 B+85.029 C+122.975
N180 X-10.877 Y-30.683 Z+213.32 B+82.405 C+121.877
N185 X-11.48 Y-30.895 Z+213.451 B+81.98 C+121.601
N190 X-11.084 Y-30.105 Z+214.118 B+79.447 C+120.517
N195 X-11.599 Y-30.482 Z+213.995 B+79.937 C+120.679
N200 X-11.673 Y-30.311 Z+214.247 B+79.007 C+120.263
N205 X-11.807 Y-29.739 Z+215.02 B+76.082 C+118.974
N210 X-11.684 Y-28.961 Z+215.913 B+72.659 C+117.492
N215 X-11.697 Y-28.415 Z+216.638 B+69.82 C+116.286
N220 X-11.657 Y-27.876 Z+217.342 B+67.009 C+115.105
N225 X-11.567 Y-27.345 Z+218.025 B+64.226 C+113.947
N230 X-11.43 Y-26.827 Z+218.686 B+61.467 C+112.815
N235 X-11.249 Y-26.324 Z+219.322 B+58.73 C+111.71
N240 X-11.025 Y-25.836 Z+219.935 B+56.014 C+110.626
N245 X-10.76 Y-25.362 Z+220.523 B+53.317 C+109.56
N250 X-10.457 Y-24.905 Z+221.087 B+50.638 C+108.512
N255 X-10.118 Y-24.465 Z+221.625 B+47.975 C+107.482
N260 X-9.744 Y-24.044 Z+222.137 B+45.327 C+106.465
N265 X-9.337 Y-23.641 Z+222.623 B+42.692 C+105.463
N270 X-8.901 Y-23.258 Z+223.082 B+40.071 C+104.473
N275 X-8.435 Y-22.896 Z+223.513 B+37.46 C+103.495
N280 X-7.943 Y-22.556 Z+223.916 B+34.861 C+102.529
N285 X-7.427 Y-22.238 Z+224.291 B+32.271 C+101.575
N290 X-6.889 Y-21.943 Z+224.637 B+29.69 C+100.631
N295 X-6.329 Y-21.671 Z+224.954 B+27.117 C+99.694
N300 X-5.751 Y-21.422 Z+225.242 B+24.551 C+98.767
N305 X-5.156 Y-21.199 Z+225.5 B+21.991 C+97.848
N310 X-4.546 Y-20.998 Z+225.728 B+19.437 C+96.933
N315 X-3.921 Y-20.822 Z+225.928 B+16.888 C+96.021
```

```
63  N320 X-3.286 Y-20.671 Z+226.096 B+14.343 C+95.115
64  N325 X-2.576 Y-20.538 Z+226.239 B+11.715 C+94.182
65  N330 X-1.859 Y-20.435 Z+226.349 B+9.089 C+93.26
66  N335 X-1.137 Y-20.36 Z+226.426 B+6.466 C+92.341
67  N340 X-0.42 Y-20.314 Z+226.469 B+3.844 C+91.451
68  N345 X+0.232 Y-20.3 Z+226.48 B+1.223 C+90.752 //Singularitet
69  N350 X-1.528 Y+20.282 Z+226.458 B+1.398 C+270.214
70  N355 X-2.848 Y+20.232 Z+226.403 B+4.019 C+271.333
71  N360 X-4.219 Y+20.141 Z+226.315 B+6.64 C+272.296
72  N365 X-5.593 Y+20.01 Z+226.195 B+9.263 C+273.237
73  N370 X-6.875 Y+19.849 Z+226.047 B+11.802 C+274.147
74  N375 X-8.15 Y+19.653 Z+225.869 B+14.343 C+275.054
75  N380 X-9.417 Y+19.421 Z+225.661 B+16.888 C+275.964
76  N385 X-10.673 Y+19.154 Z+225.422 B+19.437 C+276.878
77  N390 X-11.916 Y+18.852 Z+225.155 B+21.991 C+277.798
78  N395 X-13.144 Y+18.515 Z+224.858 B+24.551 C+278.727
79  N400 X-14.357 Y+18.143 Z+224.531 B+27.117 C+279.66
80  N405 X-15.554 Y+17.737 Z+224.177 B+29.69 C+280.599
81  N410 X-16.732 Y+17.296 Z+223.793 B+32.271 C+281.548
82  N415 X-17.89 Y+16.82 Z+223.382 B+34.861 C+282.505
83  N420 X-19.027 Y+16.312 Z+222.942 B+37.46 C+283.47
84  N425 X-20.142 Y+15.769 Z+222.474 B+40.071 C+284.445
85  N430 X-21.232 Y+15.193 Z+221.98 B+42.692 C+285.433
86  N435 X-22.296 Y+14.583 Z+221.459 B+45.327 C+286.434
87  N440 X-23.332 Y+13.941 Z+220.913 B+47.975 C+287.449
88  N445 X-24.34 Y+13.265 Z+220.342 B+50.638 C+288.479
89  N450 X-25.317 Y+12.556 Z+219.745 B+53.317 C+289.526
90  N455 X-26.262 Y+11.814 Z+219.125 B+56.014 C+290.592
91  N460 X-27.171 Y+11.039 Z+218.481 B+58.73 C+291.678
92  N465 X-28.047 Y+10.232 Z+217.815 B+61.467 C+292.786
93  N470 X-28.886 Y+9.394 Z+217.124 B+64.226 C+293.912
94  N475 X-29.688 Y+8.523 Z+216.413 B+67.009 C+295.062
95  N480 X-30.448 Y+7.619 Z+215.681 B+69.82 C+296.24
96  N485 X-31.164 Y+6.682 Z+214.929 B+72.659 C+297.452
97  N490 X-32.169 Y+5.408 Z+214.006 B+76.082 C+298.933
98  N495 X-32.793 Y+4.389 Z+213.209 B+79.007 C+300.218
99  N500 X-32.961 Y+4.072 Z+212.949 B+79.937 C+300.629
100 N505 X-32.351 Y+4.583 Z+213.076 B+79.447 C+300.355
101 N510 X-33.375 Y+3.333 Z+212.389 B+81.98 C+301.537
102 N515 X-32.935 Y+3.557 Z+212.255 B+82.405 C+301.645
103 N520 X-33.913 Y+2.215 Z+211.548 B+85.029 C+302.934
104 N525 X-34.546 Y+0.859 Z+210.601 B+88.456 C+304.54
105 N530 X-34.54 Y+0.898 Z+210.59 C+304.49
106 N535 X-34.369 Y+1.016 Z+210.634
107 N540 X-34.207 Y+1.128 Z+210.714
```

```
N545 X-34.059 Y+1.23 Z+210.827
N550 X-33.93 Y+1.319 Z+210.97
N555 X-33.823 Y+1.392 Z+211.139
N560 X-33.743 Y+1.448 Z+211.328
N565 X-33.691 Y+1.483 Z+211.531
N570 X-33.67 Y+1.498 Z+211.742
N575 X-33.68 Y+1.491 Z+211.955
N580 X-33.721 Y+1.463 Z+212.161
N585 X-33.79 Y+1.415 Z+212.356
N590 X-33.888 Y+1.348 Z+212.533
N595 X-34.009 Y+1.264 Z+212.686
N600 X-34.15 Y+1.167 Z+212.811
N605 X-58.71 Y-15.738 Z+230.652 F2000.
N615 M09
N620 M30
N625 %utensing3 G71
```

Referanser

[1] Tony L. Schmitz, Kevin S. Smith *Machining Dynamics*. 2009

[2] Industriskolen *Grunnleggende innføring i CNC-teknikk*

[3] Suk-Hwan Suh, Seong Kyoon Kang, Dae-Hyuk Chung, Ian Stroud *Theory and Design of CNC Systems*. 2008.

[4] https://www.makino.com/about/history/

[5] E.L.J. Bohez *International Journal of Machine Tools and Manufacture*. Volume 42, Issue 4, March 2002, Pages 505–520.

[6] Bjarte Halvorsen *CAM for avanserte verktøymaskiner*. juni 2015.

[7] Alan C. Lin, Tzu-Kuan Lin, Tsong Der Lin *Deriving Generic Transformation Matrices for Multi-Axis Milling Machine* . International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering Vol:8, No:7, 2014.

[8] Autodesk *Post Processor Manual*. 2013 Autodesk, Inc.

[9] https://www.youtube.com/watch?v=iO-KkegGiqQ&feature=youtu.be&t=544

[10] Professor Olav Egeland *TPK4170 Robotics: Rotation Matrices* August 18, 2014

[11] Knut Sørby *International Journal of Machine Tools and Manufacture*. Volume 47, Issue 2, February 2007, Pages 299–306.

[12] Peter Corke. *Robotics, Vision and Control*. volume 73 of Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2011.

[13] Bilder og tekst hentet fra programmet Autodesk Fusion 360 *Autodesk Fusion 360*.

[14] Tran Duc Tang, Nguyen Phu Thuy, Vuong Si Kong Le Quy Don Technical University *Tool path linearization algorithm and postprocessor for five-axis CNC machine* . VCCA-2015

[15] Autodesk Post Library http://cam.autodesk.com/posts/

[16] G-Code and M-Code Reference List for Milling *http://www.cnccookbook.com/CCCNCGCodeList.html*

[17] HSM Post Processor Customization *https://www.youtube.com/watch?v=Z-EibPEQ8W8*

[18] Post Processors 101. *https://forums.autodesk.com/t5/computer-aided-machining-cam/post-processors-101/td-p/5916925*

[19] APT.CPS post that ships with HSMWorks. *https://forums.autodesk.com/t5/machining-discussions/cl-data/td-p/6104963*

[20] Post Processor Information for Fusion 360 and HSM CAM *http://cam.autodesk.com/posts/reference*

[21] Post Processor Information for Fusion 360 and HSM CAM *http://www2.i-logic.com/manuals/Heidenhain_programming_manual_TNC.pdf*

[22] Delcam 5 axis Machining Demo at DMG MORI *https://youtu.be/ezGoVZ7MF7M?t=29*