



**Høgskolen  
i Innlandet**

Fakultet for anvendt økologi, landbruksfag og bioteknologi

**Håvard Rokvam & Ola Olafsbye**

## **Bacheloroppgave**

**Automasjon av korntørke med luftsensor**

PLC Automation of grain dryer with air sensor

**Landbruksteknikk**

**6JB299**

**2024**

Samtykker til utlån hos høskolebiblioteket

JA  NEI

Samtykker til tilgjengeliggjøring i digitalt arkiv Brage

JA  NEI

## Sammendrag

Korntørking er en viktig del av å ta vare på de produktene vi lager på jorden. Samtidig er det en prosess som er tidkrevende og skal skje samtidig med andre tidskritiske aktiviteter. Derfor vil det være en stor fordel om korntørkene kunne styrt seg selv og at bonden fikk tid til å drive med innhøsting isteden. Dette er grunnen til at vi har jobbet med å finne en løsning for å enkelt og sikkert kunne måle fuktigheten i kornet ved hjelp av tørkeluften, slik at en styring kan skjønne når kornet er tørt nok og så stoppe tørkeprosessen og/eller varsle bonden, eller kanskje etter hvert tømme og fylle tørken slik at den går helt automatisk.

For å teste dette har vi bygget en modell av en satstørke. Hvor vi så har montert en styring som vil virke på den samme måten og til slutt har vi testet om dette prinsippet kan virke i praksis. Måten vi gjør det på er at vi måler lufttemperatur og -fuktighet inn på tørken og ut av tørken, også ser vi på forholdet mellom disse. For å få sammenlignbare data regner vi om temperaturen og den relative luftfuktigheten til spesifikk luftfuktighet.

Vi konkluderer med at vi klarer å se at kornet blir tørrere og vi klarer å sette ett punkt der kornet er tørt hvis inn-luften holder seg konstant både på temperatur og fuktighet. Veien videre blir å lage en modell som gjør at vi kan kompensere for variabel temperatur og fuktig inntaksluft.

# Abstract

Grain drying is a crucial part of preserving the products we harvest in the field. However, it is a time-consuming process that must occur alongside other time-sensitive tasks. Therefore, it would be highly beneficial if grain dryers could operate autonomously, allowing farmers to focus on harvesting instead. We have been working on a solution to accurately and reliably measure the moisture in the grain using the drying air. This would enable a control system to determine when the grain is sufficiently dry, stop the drying process, and/or alert the farmer, or maybe eventually automate the unloading and refilling of the dryer, making it fully automatic.

To test this, we built a model of a batch dryer and installed a control system designed to operate in the same manner as a real one. We then tested whether this principle could work in practice. Our approach involves measuring the air temperature and humidity, entering and exiting the dryer, and analyzing the relationship between these parameters. For comparable data, we convert the temperature and relative humidity to specific humidity.

We conclude that we can detect the drying progress of the grain and determine the point at which the grain is dry, provided the inlet air remains constant in both temperature and humidity. The next step is to develop a model that compensates for variable temperature and humidity of the intake air.

# Forord

Dette er en oppgave som er veldig personlig for oss. Fordi den prøver å løse en oppgave jeg Ola står personlig oppi hver høst og gjør at innhøstingsperioden blir veldig mye mer arbeidskrevende og slitsom. Klarer vi å løse det som denne oppgaven omhandler vil vi gi kornbonden mye enklere arbeidsmengde og ta vekk en stor byrde for mange flinke kornbønder. Samtidig håper vi den kan være med på å vise bønder og andre i landbruksnæringa at det har kommet mye ny teknologi som lar oss løse mange repetitive arbeidsoppgaver med automasjon. Og dermed gi oss som bønder mer frihet og en enklere arbeidshverdag.

Vi vil takke Hans Christian Endrerud for god veiledning gjennom hele prosessen. Samt Karl Daniel Ekker for å være en god sparringspartner.

Vi vil takke felleskjøpet, Strand Unikorn, Nibio, og Høgskolen i Innlandet for tilgang på korn til testtørking

Vi vil takke Anette Dølen og Eirik Andersen for korrekturlesing og gode innspill.

Håvard og Ola

29.05.2024 Blæstad.

# Innhold

<b>SAMMENDRAG</b> .....	<b>1</b>
<b>ABSTRACT</b> .....	<b>2</b>
<b>FORORD</b> .....	<b>3</b>
<b>INNHold</b> .....	<b>4</b>
<b>1. INNLEDNING</b> .....	<b>6</b>
1.1.1 Styring.....	7
1.1.2 Målinger .....	9
1.1.3 Tørketeknikk.....	10
1.1.4 Løsningen.....	11
1.2 Problemstilling.....	12
1.3 Delmål og avgrensinger .....	12
1.3.1 Konstruksjon.....	12
1.3.2 Styring.....	12
1.3.3 Testing .....	12
<b>2. MATERIAL OG METODE</b> .....	<b>13</b>
2.1 Elektronikk.....	13
2.1.1 Temperatur- og fuktighetssensor.....	13
2.1.2 Arduino UNO R3 .....	14
2.1.3 LCD-skjerm.....	15
2.1.4 Relemodul.....	15
2.1.5 Strømforsyning .....	16
2.1.6 Elektromotor .....	16
2.1.7 Hårføner .....	16
2.2 Digitale verktøy .....	17
2.2.1 ChatGPT.....	17
2.2.2 Inventor .....	18
2.2.3 Cura .....	18
2.2.4 Arduino IDE.....	18
2.2.5 Tinkercad .....	19
2.2.6 PuTTY .....	19
2.2.7 R.....	19
2.3 Utstyr verktøy og materialer .....	20
2.3.1 3D-printer .....	20
2.3.2 Plasttype PLA .....	20

2.3.3	Lim .....	21
2.4	Metode .....	21
2.4.1	Modellering .....	22
2.4.2	Styring og koding .....	22
2.4.3	Testing .....	22
<b>3.</b>	<b>RESULTAT .....</b>	<b>24</b>
3.1	3D modell .....	24
3.2	Styring og kode .....	29
3.2.1	Ledningsnett .....	31
3.3	Testing .....	34
<b>4.</b>	<b>DISKUSJON .....</b>	<b>37</b>
4.1	Modellering .....	37
4.2	Kobling og koding .....	38
4.2.1	Brytere .....	38
4.2.2	LCD skjerm .....	38
4.2.3	Koding og Kl .....	39
4.2.4	Beregninger .....	39
4.3	Testing .....	40
4.3.1	Kornets fuktighet .....	40
4.3.2	Sensorer .....	41
4.4	Tidsbruk .....	42
4.4.1	Tegning og 3D-printing .....	42
4.4.2	Styring og koding .....	42
4.4.3	Testing .....	42
4.5	Veien videre .....	43
<b>5.</b>	<b>KONKLUSJON .....</b>	<b>44</b>
<b>6.</b>	<b>LITTERATURLISTE .....</b>	<b>45</b>
<b>7.</b>	<b>VEDLEGG .....</b>	<b>46</b>

# 1. Innledning

Kornproduksjonen i Norge har ikke hatt stor utvikling siden 70-tallet. Ja, traktorene har blitt litt større, arbeidsbredden har blitt større, og vi har fått noen flere tekniske løsninger som ikke før de siste fem årene har blitt gode nok til å være anvendbare. Selv om jordarbeidingsredskapen og såmaskinene jobber med de samme prinsippene, har de fått noen tekniske oppgraderinger, og blitt mer presise. Treskerne ser også ganske like ut som de gjorde, men det er kommet nyere og mer tekniske løsninger som gir bedre resultater under tresking, ved at det reduserer spill og registrerer avling i sanntid. De maskinene med kanskje størst utvikling er sprøyte og kunstgjødselspreder, spesielt kunstgjødselspredere har fått mer avansert teknologi for å bli bedre til å kaste små kuler presist utover et jorde. Dette er teknologi som sparer bonden store kostnader ved å bruke innsatsfaktorene nøyaktig og der vi har behov for dem. Det samme begynner å komme på sprøyter nå. Vi kan i samtid punktsprøyte ugress istedenfor å måtte sprøyte hele åkrene flatt. Til slutt har autostyring kommet for fullt som et ekstremt godt hjelpemiddel for bonden.

Men hva er det som utviklingen har hatt til felles de siste årene? Det er digitale løsninger som gjør at vi blir mer effektive. Dette trengs for at bonden skal gjøres sine oppgaver bedre og raskere. De er med på å ta vekk noe av arbeidsbyrden slik at bonden kan slappe mer av i traktoren, følge mer med på det som skjer i redskapen og identifisere feil tidligere, og dermed redusere ting som går galt. Bonden er også mer utvilt til å løse utfordringer når de dukker opp. Spørsmålet er da, hvor henger vi igjen? Hvor har ikke den digitale utviklingen kommet frem?

Det er nok mange steder vi kan få god hjelp av digitale løsninger som vi ikke har tenkt på, men i denne oppgaven skal vi snakke om korntørking og logistikk inne på låven, og hvordan vi kan gjøre livet lettere for en bonde som har eget tørkeanlegg. Problemet med tørkeanlegg på mange gårder er at de må kjøres manuelt. Det vil si at en person må gå rundt og stille om spjeld, starte motorer og følge med på at alt kornet går dit det er tenkt at det skal gå. Og så må han sitte og følge med, enten til det han fyller er fullt eller til det går tomt der han tar kornet i fra, slik at han får stoppet alt når det er ferdig kjørt rundt. For å gjøre ting enda verre så er dette ofte gamle anlegg med små elevatorer og skruer som gjør at bare å flytte et kornlass kan ta over en time.



### 1.1.1 Styring

Så hva kan vi gjøre? Jo vi kan lage en digital løsning for dette. Men hvordan kommer vi dit? Da må vi først tenke på hva vi vil løse. Transportsystemet i et kornanlegg består i hovedsak av motorer, koblet til en skrue eller en elevator / redler. Videre er det spjeld og flerveisvelgere. Moderne spjeld er mer gunstig enn eldre spjeld, fordi de kan gå av selv ved bruk av aktuatorer, istedenfor at man må dra de for hånd. Flerveisvelgere er runde, og har flere enn tre nedslipp. Derfor krever det at man kan snurre på et spjeld inne i dem, noe som kan gjøres med en trinnmotor. Deretter er det behov for en sensor som leser når siloene er fulle eller tomme. I siloer kan man bruke nivåradarer som leser høyden på det som er i siloen, eller en kapasitiv sensor som merker når den kommer i kontakt med korn. Den vil fungere både i siloer og i transportørene. Til slutt trenger vi en måte å lese når kornet i tørka er tørt. Her tenker vi å bruke en sensor for temperatur og relativ luftfuktighet på inn og en på ut-lufta på tørka for å se når kornet slutter å avgi fuktighet til luften (Jokiniemi & Ahokas, 2014).

Hvorfor kan vi ikke bruke veieceller, eller en tradisjonell vannmåler til å måle når kornet er tørt? Årsaken til at vi må benytte lufta i tørka til å måle om kornet er tørt er fordi vi må ha en metode som i minst mulig grad påvirkes av hva som blir tørket, samt hvor mye som skal tørkes ned. Dette er et krav automasjonen vil stille for å kunne fungere relativt bra i alle situasjoner. Veieceller fungerer dårlig med bakgrunn i at de må vite hvor mye vannvekt den skal fjerne. Da må det kunne tas vannprøver av lasset, for å finne et gjennomsnitt, slik at tørka vet hvor mye vekt som skal fjernes. Dersom det ikke gjøres, kan det oppstå utfordringer knyttet til forskjeller i partier, eksempelvis hvis noe av lasset er på 30% fuktighet, mens en annen del har 25% fuktighet. En tradisjonell vannmåler har den åpenbare utfordringen at den krever at kornet blir presset sammen inne i sylindren for å kunne få et resultat. En slik prosess er vanskelig å automatisere, og det anses derfor som en lite hensiktsmessig metode. I tillegg tar den bare et lite utvalg av hele partiet som blir tørket.

Det krever ikke mye å bygge en styringsenhet som kan transportere kornet rundt, så lenge spjeldene er satt i riktig posisjon. Da trenger man bare en måte å starte de motorene som går dit man vil, og en sensor som registrer når det er tomt der man tømmer eller fullt der man fyller, og som da stopper prosessen. Dette kan gjøres med en knapp, kontaktorer til motorene og sensorer som sender signal, enten når det er fullt eller tomt, om at alt skal stoppe. Dette er en enkel form for automasjon som enhver elektriker kan legge opp, men det løser egentlig ikke problemet. Vi må fortsatt stille om spjeldene manuelt og vi må starte prosessen. Det eneste dette

løser er at vi kan gå fra det når det er startet, men det vil ikke kunne sette i gang den neste prosessen som skal gjøres. Tørkinga av kornet er også en stor del av utfordringen, fordi det må til for å kunne utnytte tørkeanlegget. Tørkeanlegget må gå hele tiden, og derfor skulle man ofte gjort andre ting samtidig, som man ikke får gjort, fordi man må følge med på tørka. Samtidig skal man være ute på jordet og høste eller transportere korn, og om natta burde man helst sove litt i innhøstingen. Alle arbeidsoppgavene fører til at man ikke har mulighet til å følge med på tørka hele tiden, noe som resulterer i lav effektivitet, samt over- og undertørka korn.

For å kunne gjøre prosessen helautomatisk, er det nødvendig med en hjerne. Det er ikke like dyrt som man kanskje tenker, og krever ikke en kraftig PC. Det holder med en liten mikrokontroll, som vil ha kontroll på når det skal starte og stanse, samt hvilke sensorer den skal lese av, for å forstå når den skal stoppe, samt starte neset oppgave.

I praksis er det samme som bonden gjør, bare at man unngår menneskelige feil. Eksempler på hjerner er Arduino UNO (*Arduino Uno Rev3*, 2024) eller Raspberry Pi (Ltd, 2024). Hjernene er billige og enkle komponenter, som lar deg styre forskjellige elektriske komponenter. Det finnes også mer profesjonelle hjerner som heter PLC, «Programmable Logic Controllers». PLC-hjerner kan fungere bedre, men da er man mer prisgitt sin leverandør på hvilke muligheter dem har, og de kan være vesentlig dyrere.

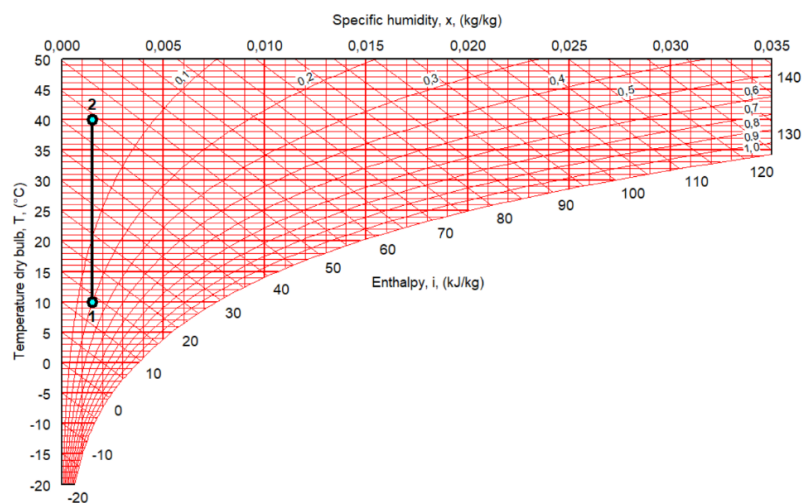
Begge hjernene er aktuelle til å benytte som styring, men de har visse ulikheter. Arduinoen er en mikrokontroller som krever en annen PC til å lage koden (*Arduino Uno Rev3*, u.å.). Deretter kan man laste programmet over på Arduinoen, slik at den kjører samme program hver gang den blir startet. Raspberry Pi er annerledes ved at det er en fullverdig PC der man laster opp et operativsystem og skriver koden direkte på PC-en. Fordelen med en raspberry Pi, er at det er enklere å se hva man skal gjøre, samt hva som skjer. Det gjør man fordi styringen hele tiden viser hva den gjør, samtidig er det mer utfordrende å kode en Raspberry Pi.

Det hjernene har til felles, og som gjør de egnet til enkel automasjon, er at de har gpio pinner, general purpose input/output. Man kan koble seg til pinnene og lage en kode som enten leser hva pinnene får av signaler, eller de kan sende ut binær verdi. høy eller lav. Metoden gjør det enkelt å styre oppgaver som er enten av eller på, noe et kornanlegg er. Om motorer skal gå eller ikke, eller om spjeld skal være åpne eller lukket.

## 1.1.2 Målinger

I tillegg til en hjerne trenger dette systemet sensorer for fungerer som følere til hjernen og sender signaler hjernen kan tolke. I denne oppgaven skal vi se på luftfuktighetene og da trenger vi sensorer som kan lese dette. Det finnes det masser av på elektronikk markedet av varierende kvalitet. Men det disse sensorene har til felles er at de alle leser temperatur og relativ luftfuktighet. Og den må lese begge deler siden den relative luftfuktigheten er helt avhengig av temperaturen siden temperaturen bestemmer hvor mye vann luften klarer å holde på. Det vil si at når luften når 100% luftfuktighet er luften vannmettet og den begynner å frigjøre vann (Samuelsen, 2023).

Men en korntørke vil vi ofte få forskjellige temperaturer på luften som går inn på tørken og som kommer ut igjen. Dette er fordi når luften går igjennom kornet vil den bli påvirket av både fuktigheten i kornet og temperaturen til kornet. Slik at når det er høyere fuktighet i kornet enn det er i luften vil luften ta opp fuktighet fra kornet, men hvis det er motsatt at luften har høyere fuktighet enn kornet vil kornet ta opp fuktigheten fra luften. Da fukter vi kornet, ikke tørker det. Det samme vil skje med temperaturen på kornet og i luften. Derfor er relativ luftfuktighet et dårlig mål på å sammenlikne luften inn og ut av tørken. Siden temperaturen sjeldent er lik spesielt på varmluftstørker. Derfor må vi legge inn en formel for å regne om den relative luftfuktigheten til spesifikk luftfuktighet. Den spesifikke luftfuktigheten er et mål på hvor mye vann det er i en kilo luft og dette er likt uavhengig av temperaturen. Det vil si at på samme spesifikke luftfuktighet ved ti grader vil den relative fuktigheten være 20% mens på 40 grader vil den relative fuktigheten være på 3,3%. Dette vises veldig godt i figur 1, hvor vi ser disse temperaturene og fuktighetspunktene markert. Her har vi den relative fuktigheten som de bøyde



Figur 1 Mollier diagram. Egenprodusert

linjene inne i grafen, og vi ser godt at når temperaturen stiger synker den relative fuktigheten fort

For å regne om dette trenger vi en formel. Denne formelen har 2 deler. Først må vi regne ut vanndamptrykket ved den gitte temperaturen og den relative luftfuktigheten. Dette gjøres ved formel:  $P_v = RH/100 \times P_{sat}(T)$ . Dette er en omgjort versjon av Clausius – Clapeyron ligningen (Helseth, 2024). Her er  $P_v$  damptrykket,  $RH$  relativ luftfuktighet,  $P_{sat}$  er en betegnelse for hvordan Clausius – Clapeyron ligningen bruker faste verdier som er forsket frem for å regne ut metningstrykket til luften ved en gitt temperatur. Dette regnes på denne måten  $P_{sat}(T) = 6.112 \times \exp(17.67 \times t / t + 243.5)$  (Koutsoyiannis, 2012). Alle de gitte verdiene er forsket frem og funnet at disse stemmer ganske godt med virkeligheten. 6.112 er for å beregne metningstrykket i vanndampen. Mens 17.67 og 243.5 er verdier som gjør det mulig å beregne metningstrykket i forhold til temperaturen. For at denne formelen skal fungere må temperaturen være omgjort til kelvin istedenfor celsius som sensorene måler.

Når dette er regnet ut kan vi gå over til å bruke damptrykket til å regne ut den spesifikke luftfuktigheten. Det gjøres ved å fullføre Clausius – Clapeyron ligningen ved denne formelen:  $q = 0.622 \times P_v / P - P_v$  (Helseth, 2024). Her får vi spesifikk luftfuktighet, ved å bruke konstanten 0.622 som er forholdet mellom tørr luft og våt luft (Nugent & Russell, 2024),  $P$  er atmosfærisk trykk som vi bruker 1013.25 hPa som er vanlig atmosfærisk trykk på havnivå (Nugent, 2024).  $P_v$  er damptrykket vi kom frem til i formelen over. Da får vi spesifikk luftfuktighet i kg vann pr kg luft, som er en litt rar og tungvinn enhet. Vi deler den derfor til slutt på 1000 og får gram vann pr kg luft.

Når vi har gjort denne omregningen får vi et tall som kan sammenliknes uavhengig av temperaturen på luften. Dette er en stor fordel nå som vi skal prøve å bruke luftfuktigheten til å måle om kornet er tørt eller ikke. Dette vil gjøre at prinsippet tåler store temperaturforskjeller og allikevel kan se hvor lenge det fjernes vann fra kornet.

### 1.1.3 Tørketeknikk

For å tørke korn trenger vi tørr luft som hele tiden blir blåst igjennom kornet. Når luften er tørrere enn kornet vil luften ta til seg fuktigheten til kornet. Her trengs det ett punkt som slipper luften inn og et sted luften kan gå ut. Kornet som ligger imellom disse to punktene ligger i

den såkalte «tørkesonen» (Fladstad & Tengedal, 2002). Hvor stor og hvordan tørkesonen er utformet er det stor forskjell på fra de forskjellige tørkene. I denne oppgaven har vi laget en modell av en rullerende satstørke. Denne består av mange små tørkesoner som igjen blir satt sammen til en stor. Dette gjør tørkingen veldig effektiv (Fladstad & Tengedal, 2002).

Når vi skal tørke korn har vi vann i tre faser. Den første og enkleste fasen er fritt vann. Altså vannet som befinner seg rundt uten på kornet. Så har vi bundet vann. Dette er vannet som er inni kornet og gjør at kornet blir oppsvulmet og kjennes bløtt. Til slutt har vi hydratiseringsvann. Dette er det vannet som er en del av bestanddelen i selve kornet (Reykdal et al., 2018). Etter hvert som man tørker kornet blir det tyngre og tyngre å få ut vannet og det går saktere og saktere.

#### **1.1.4 Løsningen**

Dette kan man selvfølgelig få de som selger komplette tørkeanlegg til å lage. Problemet med dette er at ett komplett nytt tørkeanlegg med råbygget rundt og styring blir ekstremt dyrt i forhold til det man tjener på å tørke og lagre selv. Da kan man heller få litt dårligere betalt for kornet og levere direkte fra åkrene slik at tørking og lagring blir mølla sitt problem. Med et slikt enkelt automasjonssystem kan vi modernisere dagens relativt tungdrevne gårdsanlegg til å bli minst like moderne som de møllene Felleskjøpet drifter. I tillegg er det ingen begrensinger på funksjoner man kan legge til siden man koder selv og det eneste som egentlig setter begrensinger er den som koder. Samtidig, har man først lagt alt inn slik at det styres fra en digital hjerne kan man få inn eksterne kodere til å få det slik man vil hvis man ikke klarer dette selv.

Nå begynner vi å komme inn på kjernen i denne oppgaven. Det er å teste om vi i prinsippet klarer å måle om kornet er tørt i tørken slik at vi kan stoppe tørkeprosessen på riktig tidspunkt og gjøre at tørken kan tømme og fylle seg selv. Dette vil gi en kornbonde mye mer handlingsrom og muligheter til å drive ute på jordet i den hektiske innhøstingstiden. Samtidig får han stor kapasitet på et kanskje eldre og litt for lite tørkeanlegg. Dette vil kunne sammenliknes litt med at en melkebonde får en melkerobot. Det er fortsatt behov for å følge med og følge opp, men det gir bonden friheten til å kunne gjøre det når det passer inn i hans hverdag, og ikke når kua eller tørka må ha bonden på plass.

## 1.2 Problemstilling

Under tema gårdsautomasjon er det veldig mye vi kunne tatt for oss, og etter hvert som vi jobber med dette ser vi flere og flere muligheter. Men vi rekker ikke å se på alt i denne oppgaven så vi må snevre det inn til en liten bit. Vi skal ta for oss tørkeanlegg og hvordan vi kan styre dette automatisk.

Problemstillingen blir derfor:

Teste om vi i prinsippet klarer å se om kornet er tørt ved hjelp av lufttemperatur og – fuktighetssensorer på inntaksluft og eksosluft. Kan si om kornet er tørt eller ikke.

Med dette kommer det bygging av en modell varmluftstørke som lar oss teste prinsippet med ekte korn på en veldig liten skala som vi har god kontroll over.

## 1.3 Delmål og avgrensinger

Siden oppgaven blir så bred og stor prøver vi å avgrense og dele opp oppgaven litt.

### 1.3.1 Konstruksjon

Delmålet for konstruksjonen er å lage en funksjonell modell av en satskorntørke. Den skal kunne tørke ekte korn på en relativt like måte som en faktisk fullskala tørke.

### 1.3.2 Styring

Delmålet for styringen er å lage en funksjonell styring som leser sensorene og utfra de kan både styre tørken og logge data slik at vi kan behandle de i etterkant.

### 1.3.3 Testing

Målet med testingen er å se om det å måle forskjellen på fuktigheten i inntaks- og uttaksluften kan si om kornet er tørt eller ikke. Målet er ikke nødvendigvis å lage en ferdig løsning, men heller å se om det er noe man kan jobbe videre med og optimalisere slik at man på sikt kan få en ny type styring av korntørker.

## 2. Material og Metode

### 2.1 Elektronikk

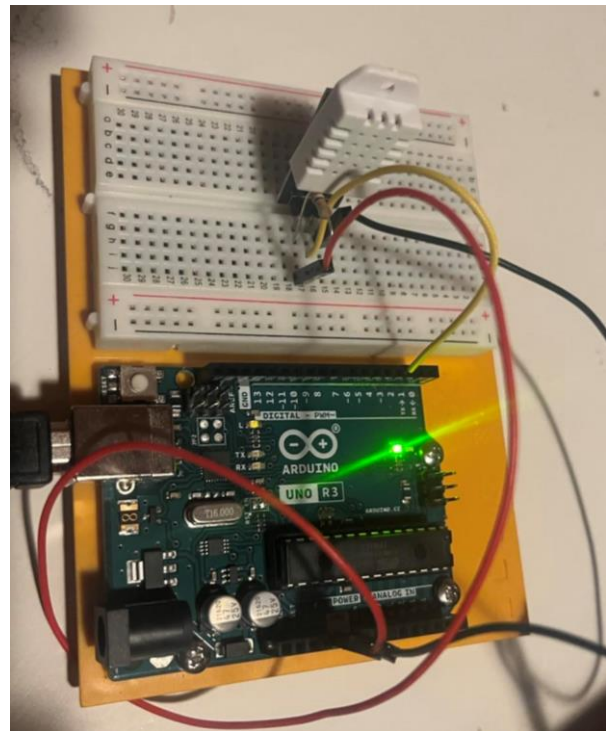
I denne oppgaven bruker vi en del forskjellige elektroniske komponenter. Det er dette som egentlig er den sentrale delen av oppgaven og da spesielt luftfuktighet og temperatur sensorene som er det vi skal teste om kan brukes til å fortelle en styringsenhet at kornet faktisk er tørt. Ellers vil det bli brukt en del forskjellige komponenter og en styring for at alt kan snakke sammen. Det som er kult med dette er at akkurat det samme kan i praksis brukes på ett ekte fullskala tørkeanlegg. Vi har også noen komponenter som bare brukes til å få modellen til å fungere. Disse kommer vi til i slutten av dette underkapittelet.

#### 2.1.1 Temperatur- og fuktighetssensor

For å kunne lese om kornet er tørt bruker vi en DHT 22 temperatur- og luftfuktighetssensor. Grunnen til at vi velger denne sensoren er at det er en billig sensor som har et måleområde godt innenfor det som den trenger for å kunne lese temperaturen og luftfuktighetene til tørkeluften.

Denne sensoren har et måleområde fra  $-40$  til  $80$  pluss grader celsius med en nøyaktighet på  $\pm 0,5$ . Luftfuktigheten leser den fra  $0$  til  $100\%$  med en nøyaktighet på  $2$  til  $5\%$ . Den sender ut nye verdier en gang hvert andre sekund, noe som er mer enn nok for å kunne lese når verdiene er innenfor de rette perimetrene.

Sensoren trenger strøm,  $3,3 - 5$  V, jording og en datakabel som kobles til mikrokontrolleren. I tillegg trenger den en såkalt «pull up resistor» for å holde signalet høyt på data kabelen. I dette tilfellet gjøres det ved å bruke en  $10$  k ohm motstand mellom plusskabelen og datakabelen (*In-Depth*, 2018). For å kunne koble opp sensoren i Arduino IDE må vi bruke et bibliotek som



Figur 2 Dht 22 sensor koblet opp. Egenprodusert

lar skriptet tolke signalene fra sensoren. I dette tilfellet er det DHT sensor library (*adafruit/DHT-sensor-library*, 2011/2024). På dette bildet ser vi hvordan sensoren blir koblet direkte til Arduinoen med et kontaktbrett. Koden for dette er lagt ved i «Vedlegg 1 koble opp en sensor»

Fra nå av vil vi referere til denne sensoren som DHT 22.

## 2.1.2 Arduino UNO R3

Arduino UNO R3 er «hjernen» til tørka vår. det er den som leser sensorene og utefra det viser oss data og bestemmer hva motorene skal gjøre. Arduino UNO R3 er et eksempel på det som kalles en mikrokontroller. Dette er som en liten PC som kan jobbe selvstendig og styre ting. Den er utstyrt med en ATmega328P prosessor (*Arduino Uno Rev3*, 2024). Dette er komponenten som bestemmer hva som skal skje utfra en kode den har fått opplastet.

Utover dette har Arduinoen 14 digitale pinner som fungerer binære, altså at den bare kan lese eller sende ut høyt eller lavt signal. Dette fungerer utmerket til de fleste situasjoner hvor det skal leses om en sensor slår ut eller ikke, for eksempel ved en fullmelder som sier at en silo er full, eller som vi bruker i denne oppgaven, at den sender ut et høyt signal når en motor skal starte og sender ikke ut signal når den ikke skal gå. Seks av disse pinnene har også mulighet til PWM, Pulse Width Modulation eller på norsk, puls brede modulasjon, som kort forklart er at man kan regulere bredden på de elektriske pulsene mens frekvensen holder seg konstant. Det gjør at man enkelt kan regulere kraften som går til en komponent og for eksempel endre hastigheten på en motor eller dimme en LED-lyspære (*Pulse Width Modulation - an overview / ScienceDirect Topics*, 2024).

Arduinoen har også seks analoge pinner som lar oss sende ut og lese forskjellige analoge signaler. Et analogt signal er at en gitt volt betyr en verdi sensoren leser, og mikrokontrolleren må da tolke volten til en verdi, for eksempel temperatur. Den kan også sende ut forskjellige mengder volt som kan brukes på same måte som den digitale PWM til å endre hastigheten på motorer eller dimme lys.

Arduinoen trenger strøm fra en strømkilde. Når vi koder og tester ut kontrolleren og sensorene bruker vi USB-en på PC fordi den også brukes til å laste opp ny kode, men når vi bare tester tørka bruker vi en ekstern strømkilde. Dette kommer vi tilbake til lenger ned.



Fra nå av vil vi kalle Arduino Uno R3 bare for Arduinoen eller mikrokontrolleren.

### 2.1.3 LCD-skjerm

Vi bruker en enkel alphanumeric LCD-skjerm av typen S – STN (*Liquid Crystal Displays (LCD) with Arduino | Arduino Documentation, 2023*). Denne bruker vi for å vise sensor dataen fra hver enkel sensor ovenfor hverandre i sanntid. Dette gjør vi både for vår egen del under testing, for å bedre forstå hva som skjer i tørka, og for at folk som kommer og ser enklere skal forstå hva som skjer i praksis. Det er en relativt liten skjerm som kan ha 16 karakterer på to linjer, noe som gir litt begrensinger i hva vi kan vise. Skjermen er satt rett i koblingsbrettet og er koblet slik som vist på figur 16.

### 2.1.4 Relemodul

Vi bruker en relemodul i denne oppgaven for å gi større strømstyrke til motorene vi bruker, siden disse krever mer kraft enn hva Arduinoen sender ut. Ett rele kan enkelt forklares som en elektrisk bryter som skrur seg av og på ved hjelp av en styrestrøm. Når den styrestrøm er den som oftest på og sender igjennom den strømstyrken som er koblet til. Men hvis styrestrømmen er av sender den ikke gjennom den sterkere strømmen. Dette er det som er så genialt når vi setter en mikrokontroller sammen med det rette releet, så kan vi styre anlegg som krever store mengder strøm bare ved hjelp av den lille 5 V styrestrømmen som Arduinoen sender ut.

Den konkrete relemodulen vi har brukt i denne oppgaven er en Luxorparts relemodul med 4 utganger (*Luxorparts Relemodul for Arduino - Moduler, 2024*). Denne er laget for å bli styrt av Arduinoer og er utstyrt med pinner for å enkelt kunne koble seg imellom mikrokontrolleren og relemodulen. For å koble denne opp trenger den konstant 5V strøm og jording, pluss at den må ha styrestrøm til de releutgangene den skal styre. Til slutt må den ha den store strømkilden den skal bruke. Den takler opptil 30 V DC strøm med 10 Ampere og den kan ta opptil 250 AC strøm med 5 Ampere. Her er det viktig å ikke belaste den med mer enn 10 Ampere (*Luxorparts Relemodul for Arduino - Moduler, 2024*). Da kan man risikere at hele releet brenner opp.

## 2.1.5 Strømforsyning

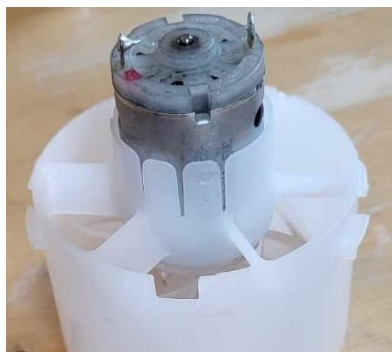
Som strømforsyning hadde vi en universal PC-lader liggende som vi tidligere hadde kjøpt på Elkjøp. Denne er 230v AC inn fra vanlig støpsel i hus. Ut har den en USB-port som gir ut DC 5V strøm som vi bruker til å styre Arduinoen. Hoveduttaket på laderen som er til å lade datamaskiner er DC strøm der du kan justere volten med et lite hjul mellom 12V og 22V. Vi syntes dette var praktisk siden vi da har muligheten til å justere hastigheten på motorene med strømstyrken. Uttaket bruker vi som strømforsyning til viftemotoren og kornskruen. Strømmen mellom strømforsyningen startes og stoppes av relect.

## 2.1.6 Elektromotor

Til skruen bruker vi en elektromotor for å rotere skruen slik at kornet blir dyttet oppover. Her prøvde vi først med en veldig liten 5V elektromotor uten giring. Det fungerte ikke siden den ikke hadde noe særlig dreiemoment. Derfor måtte vi finne en annen motor som var sterkere og med fordel roterer litt saktere.

Den motoren vi landet på var en «RS PRO, 12 V dc, 10 Ncm, Brushed DC Geared Motor, Output Speed 82 rpm»(*RS PRO Brushed Geared DC Geared Motor, 1.31 W, 12 V Dc, 10 Ncm, 66 Rpm, 4mm Shaft Diameter | RS PRO, 2024*). Dette er motor som kan brukes fra 6V til 24V og man kan styre hastigheten ved å øke eller senke volten. Den har også en nedgiring på 1:100, altså motoren må rotere 100 ganger for at akselen skal rotere 1 gang.

## 2.1.7 Hårføner



Figur 3 viftemotor fra hårføner.  
Egenprodusert

Når vi skulle finne en passende vifte til tørka så undersøkte vi først på internett etter kjølevifter til datamaskin, men vi fant snart ut at det var like greit at vi brukte en hårføner. Vi kjøpte derfor en hårføner på biltema med flere hastighetsinnstillinger. Vi håpet å kunne bruke denne i sin helhet og kun koble til styring, men vi merket fort at dette ble stort og klumpete så vi demonterte hårføneren til småbiter og brukte kun vifta. Siden den hadde en maksimal strømstyrke på 24 volt koblet vi den

opp mott et rele til styringen. Vifta hadde et rundt viftehus så vi 3D printet en hylse som vi tredde vifta i og som kobles rett på kanalen til innluft.

## 2.2 Digitale verktøy

For å lage alt vi skal i denne oppgaven bruker vi endel forskjellige digitale verktøy. Vi bruker Inventor og Cura for 3D tegning og modellering. Vi bruker tinkercad til å lage flytskjema. Vi bruker Arduino IDE for å skrive og laste opp kode og vi bruker ChatGPT til å hjelpe oss med koding og andre generelle spørsmål.

### 2.2.1 ChatGPT

Dette bruker vi i hovedsak for å hjelpe oss med kodingen. Grunnen til at vi bruker KI til koding er fordi den er mye flinkere til å komme raskt med forslag til kode, som vi da kan modifisere til det spesifikke formålet vi trenger. Dette sparer oss for ekstremt mye tid som vil bli brukt på å søke opp og finne den rette måten og skrive koden på. Isteden kan vi fortelle ChatGPT hva vi ønsker, og den lager et forslag til oss på 10 sekunder som vi så kan teste og forbedre til det spesifikke formålet vi trenger. Dette stiller fortsatt krav til å forstå hvordan kodingen fungerer og hvordan kode skal skrives. Man kan ikke gjøre dette effektivt uten en god grunnleggende forståelse for koding, men man trenger ikke å skrive hvert enkelt ord grammatisk rett for å få koden til å fungere. Dette er ofte problemet med å få en kode til å fungere, siden den må ha helt korrekt tegnsetting og grammatikk for å fungere.

Vi bruker også ChatGPT i noen grad til å komme med forslag på tekniske løsninger og ideer på hvordan ting best mulig skal kobles opp. Dette er også for å spare tid siden KI mye raskere klarer å slå opp i spesifikasjonene til komponentene og fortelle oss hvordan ting skal kobles for å fungere. For å få dette til å fungere kreves det her også en god grunnleggende forståelse av sensorene, mikrokontrollene og hvordan strøm fungerer i en krets.

Vi har brukt KI fra chatgpt.com. og vi har brukt gratisversjonen som på tiden vi skriver dette er versjon 3.5.

## 2.2.2 Inventor

Vi bruker Inventor til å tegne 3D-modellen av tørka før den blir 3D-printet. Tørka er såpass stor og har en del vinkler som gjør at den ikke lar seg 3D-printes i en stor del. Dette gjør at vi må lage mange små deltegninger som vi til slutt setter sammen i en sammenstillingsfil slik at vi kan se at ting passer om hverandre før vi printer de ut i praksis. Dette sparer oss for mye tid og bortkastede printer siden vi slipper å lage alle delene for så å tilpasse de slik at de passer ferdigprintet. Det gjør det også vesentlig lettere å tegne når vi bare kan måle opp det vi trenger til de forskjellige delene på den digitale modellen.

Versjonen vi har brukt av Inventor er «autodesk inventor professional 2024».

## 2.2.3 Cura

Cura er en såkalt «slicer» for 3D printere. Det er et program som omgjør en STL-fil som står for stereolitografi som er en måte å beskrive hvordan 3D objektet ser ut for programmet. Vi henter STL filen fra Inventor og gir den til Cura som lager «G code» som er en kode man gir til 3D-printeren, som sier hvordan den skal bevege alle sine akser for å lage det objektet vi har tegnet i Inventor. I Cura kan man også gjøre finjusteringer på hvordan printeren skal oppføre seg i forhold til hva den printer. Her er det typisk temperatur og hastighet man stiller på, og så bestemmer man om man trenger noe støtte på enkelte deler når den blir printet og hvordan den støtten skal oppføre seg.

## 2.2.4 Arduino IDE

Arduino IDE er en kodeplattform spesial laget for å være enkel og kommunisere med Arduino sine mikrokontrollere. Med kodeplattform menes et program som er laget for å skrive kode. Dette er et enkelt program med mange funksjoner som gjør det enklere å skjønne hva som skjer, hva som fungerer, og hva vi må jobbe mer med. Arduino IDE er basert på kodespråkene C og C++.

## 2.2.5 Tinkercad

For å tegne ledningsnett til modelltørka har vi brukt 3D-programmet Tinkercad. Der ligger det ferdige moduler av skjermer, sensorer, releer og hjerner som du kan koble sammen, tegne inn ledninger mellom, og velge farge på. Hvis du vil se at alt er koblet riktig kan du laste inn koden i programmet og testkjøre systemet der. Du får også et profesjonelt ledningsnett ut i riktig format som du kan bruke til å koble i virkeligheten.

## 2.2.6 PuTTY

PuTTY er en programvare som lar oss lese data som blir sendt over forskjellige porter på PC. Vi har brukt dette til å lese dataene Arduinoen leser fra DHT2 sensorene som da blir sendt til PC gjennom, i dette tilfellet, com port 3 som er en av USB-portene på PC-en. PuTTY legger så de dataene den leser inn i en tekstfil som vi da enkelt kan ta igjen og behandle i R.

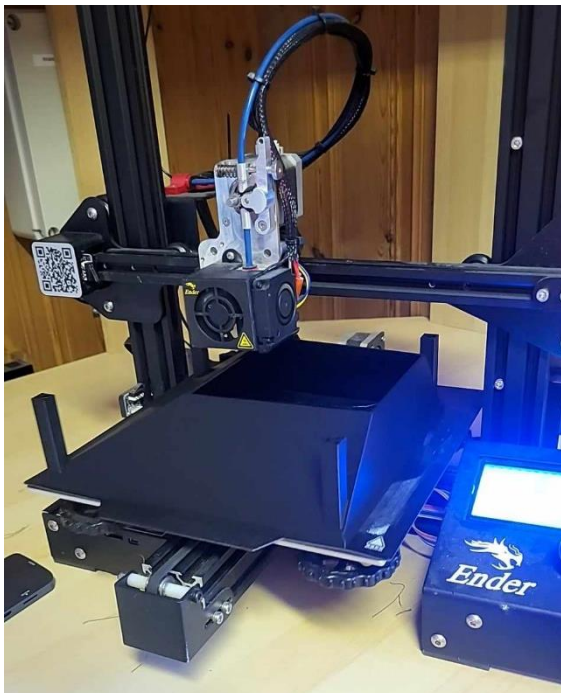
## 2.2.7 R

Vi har brukt R til å behandle og presentere dataene vi har logget fra tørken. Dette er ett program som lar oss behandle store datasett på en enkel måte med hjelp av koding. Her slipper vi å håndtere dataene direkte, men kan heller behandle hele datasettet på likt. Dette er veldig fordelaktig siden vi leser data hver andre sekund, noe som gjør at det blir utrolig mange datapunkter pr datasett.

## 2.3 Utstyr verktøy og materialer

### 2.3.1 3D-printer

Vi bruker en Ender 3 3D-printer. Dette er en god og stødig 3D-printer som er relativt billig. Den er enkel i bruk, men krever litt tid for å få satt opp og fininnstilt godt slik at du får fine og solide printer. Den krever også litt ettersyn og vedlikehold for at den skal fungere optimalt. Altså vi må ha et øye med den hele tiden slik at det ikke bare blir en svær haug med plastikk.



3D-printeren har et byggevolum på 235 mm bred og 235 mm lang og 250 mm høy.

Figur 4 3Dprinter i bruk for og printe bunndel.  
Egenprodusert

### 2.3.2 Plasttype PLA

Vi bruker en plasttype som heter PLA-plast som står for polylactic acid. Vi bruker denne i hovedsak fordi det er en av de billigste og mest «lett»-printede plastkktypene som man får til 3D-printing. PLA er ikke veldig sterkt, men modellen vi har laget stiller heller ikke veldig store krav til at plastikken må tåle mye. Dette gjorde det veldig enkelt å lande på PLA som plasttypen vi bruker. En annen fordel med PLA er at den er laget av biomasse som maistivelse eller sukkerrør, og da ikke som tradisjonell plastikk som er laget av olje (PLA Plastic/Material – The Ultimate Guide, 2023).

### 2.3.3 Lim

I hovedsak designer vi delen til å bare kunne sitte løst sammen. Men noen av delene må limes fordi vi ikke klarer å 3D-printe de i en hel del. Dette er da spesifikt inn- og ut-luftveggene og skruen. Her blir delene printet ut hver for seg og må limes sammen. Skruen krever et godt sterkt lim slik at den ikke bare løsner der vi har limt sammen de to delene. Samtidig kan det ikke bli en svær klump med lim siden skjøten også er en del av skruen og kornet må klare å skli forbi der. Mens inn- og ut-luftveggene er litt annerledes. Her er det viktig at det tetter godt og at limet ikke sprekker opp hvis plasten vrir seg litt ved at varm luft går igjennom. Her kreves et lim med mer elastisitet og bedre «fuge»-egenskaper.

På skruen har vi valgt å gå for Loctite Power Epoxy Universal 5. Dette er et sterkt to-komponent lim som har fungert godt til å holde skruen sammen uten at man trenger mye lim. Når man klarer å blande den riktig, herder den også veldig fort som er en fordel siden skruen er litt vanskelig å få til å ligge helt rett.

På inn- og ut-luftveggene har vi valgt å gå for Tec7 som er en god fuge-, lime- og tettemasse. Denne er mer elastisk og tåler varme slik at den ikke sprekker om så plasten skulle vir seg litt. Samtidig er den ekstremt god på å tett rundt steder man ikke vil ha luftlekkasjer.

## 2.4 Metode

Vi har tre ganske forskjellige ting vi har drevet med som er det denne oppgaven munner ut i. Vi begynte med mye modellering, for så å utvikle styringen. Da altså både sette sammen komponent og utvikle kode for å få alt til å gjøre som vi ville, og til slutt teste prinsippet som oppgaven faktisk skal prøve å finne ut av.

### **2.4.1 Modellering**

Måten vi vil gå frem for å modellere og designe selve tørken på vil være at vi begynner med å tenke ut hvilke mål vi kan bygge utefra. Da er det 3D-printeren som setter begrensningene på hvor stort hvert element kan være.

Vi kommer til å begynne med selve tørke delen av tørken og bygge oss utefra den. Det vi mener med tørke delen er det luften skal komme inn og gå igjennom kornet. Her vil det være to parallelle vegger som slipper luft inn og fanges opp i et møne som går på tvers igjennom hele tørken. Når vi har bestemt oss for størrelsen på dette vil vi gå videre til bunnen slik at den får riktig vinkel og førene kan renne greit ut mot oppsamlingspunktet. Når vi har disse målene kan vi designe en skrue som passer in mot bunnen og til slutt lage en synkesone som passer over toppen og holder alt sammen.

Det er viktig at alt passer godt siden vi har tenkt at selve tørken skal være demonterbar hvis man trenger å fjerne noe som har tettet seg eller hvis man vil vise hvordan den fungerer inni.

### **2.4.2 Styring og koding**

Når vi går frem for å bygge styringen, altså koblingen av Arduinoen og kodingen, vil vi gå frem på samme måte som ved modelleringen. Vi vil begynne med å få sensorene til å fungere også gå videre med å legge på forskjellige funksjoner som vi trenger. Det vi ser for oss før vi begynner er at vi må kunne lese sensorene, starte og stoppe motorene til vifta og skruen. Vi må også kunne sende data over til en PC som kan logge de i ett tekstdokument slik at vi videre kan analysere dataene vi får i fra tørken.

### **2.4.3 Testing**

Det er først er vi skal begynne å se om vi klarer å finne noen sammenheng med forskjellene i lufttemperatur og fuktighet og om dette kan fortelle oss om kornet er tørt. Klarer vi å finne dette vil vi enkelt kunne stoppe tørkeviften og starte tømning når vi når en terskelverdi vi har satt for at kornet er tørt.

For å kunne teste hva verdiene må være på for at kornet er tørt må vi først lage fuktig korn. I tørken har vi funnet ut at forreddikfrø er de beste for oss å bruke som korn, siden dette er



ganske små, harde og helt runde frø. Dette gjør de godt egnet både til å flyte gjennom tørken og lar seg greit skru opp igjen av skruen vår. Dette har vi kommet frem til etter mye testing med mange forskjellige kornsorter. Vi har testet bygg, havre, hvete, oljefrø, mais og ris, men har landet på at forreddikfrø fungerer best til å vise hvordan tørken fungerer. Men når vi skal teste hvordan sensorene leser luften i forhold til fuktigheten på kornet har vi brukt hvete siden dette er enklere å få presise målinger med vanlig vannmåler til korn på.

Først når vi skal teste fyller vi spissbunnen med tørre frø slik at vi ikke får en utørka bløtt masse i bunnen. Vi må gjøre dette siden vi ikke har «rulleringsspjeld» i bunn av tørkedelen av tørken vår som det er på fullskalatørker. Dette spjeldet gjør at det ikke står bløtt korn i spissbunnen under tørking. Så fyller vi nok korn over i en vanntett pose som vi kan lokke igjen også fyller vi den med 2dl vann. Da får vi omtrent 35 % fuktighet på frøene, det er ikke så viktig at frøene alltid har lik fuktighet siden vi vil teste at prinsippet i tørken fungerer uavhengig av hvilken fuktighet kornet har. Så heller vi disse frøene oppi tørken slik at de fyller tørkesonen og godt opp i synkesonen slik at når fuktigheten blir borte og frøene blir mindre er det nok til at vi fortsatt får dekket tørkekanalene, slik at luften ikke slipper ut gjennom toppen.

Da er vi klare for å starte tørken, men for å fange dataene fra sensorene slik at vi kan analysere tørkeforløpet og prøve å finne de verdiene vi vil stoppe tørken på må vi koble styringen til en PC slik at vi logger alle dataene til en tekstfil vi senere kan analysere. Dette gjøres over en USB-kobling som sender data igjennom COM port 3 til PC. Samtidig kan vi se på sensordataene i real time på LCD-skjermen som er koblet på styringen. Dette gjør at vi enkelt kan se når tørkingen ikke foregår lenger og vi kan avslutte testen.

Når testen er avsluttet kan vi lege dataene inn i R og se hvor fuktigheten og temperaturen flater ut og blir likere og likere, da kan vi anta at kornet er tørt. Dette krever selvfølgelig at vi ikke har fuktig luft som vi bruker til å tørke, men i dette testscenariet er ikke det ett problem siden vi står inne i et tørt verksted på ca. 20 grader celsius.

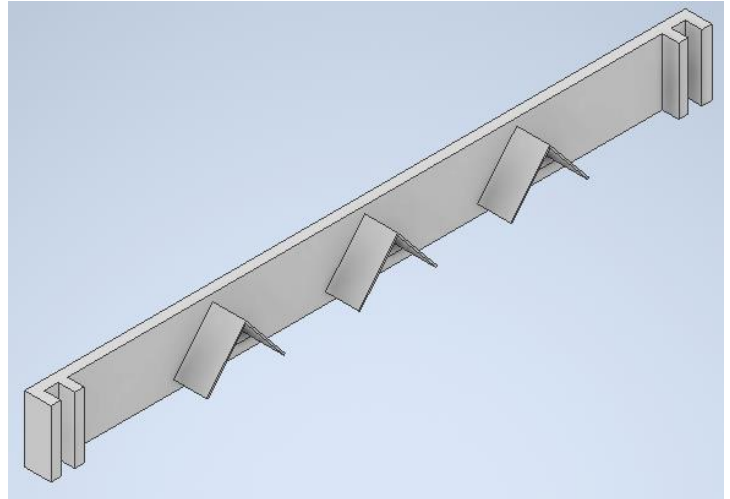
## 3. Resultat

### 3.1 3D modell

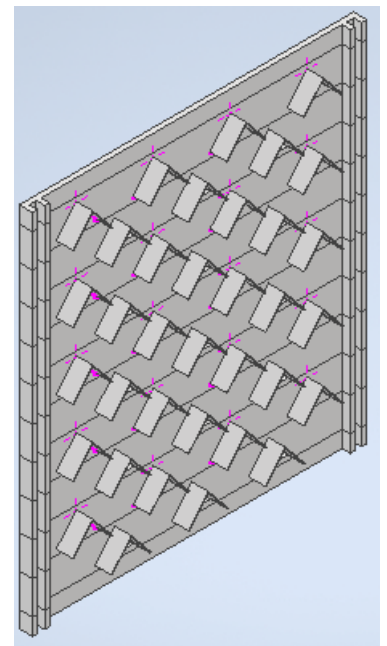
Det vi begynte med da vi skulle lage modellen var å bygge veggene hvor luftstrømmen skal komme ut og gå inn igjen. Dette er veggene i tørka som går parallelt med hverandre og som gjør at innluften kan gå inn i kornet fra ene siden, stige opp gjennom litt korn også gå ut igjen av den andre veggen etter den har tatt med seg fuktighet vekk fra kornet. Her måtte vi også bestemme oss for hvor stor tørken

skulle være etter som alt annet må bygges lik at de passer til disse veggene. Selve tegningen ble laget ved at vi først lagde seksjoner for inn- og utluft, også stablet vi disse oppå hverandre til vi fikk ønsket høyde på inn- og utluftveggene. På bildet av veggen ser vi også at det er trekantene der utluften skal gå ut. Dette er for å kunne legge på de takformene som skal gå tvers igjennom hele tørken og som gjør at luften slipper inn og ut av de rette kanalene uten at kornet får rent ut gjennom luftkanalen.

Så fant vi ut at istedenfor å 3D-printe de to andre veggene som selve tørken består av, så vil vi skjære de ut av pleksiglass. Dette er for å gjøre det enklere å se hva som er og skjer inne i tørken. Dessuten er det vanskelig å få store glatte og fine plater på 3D-printeren.

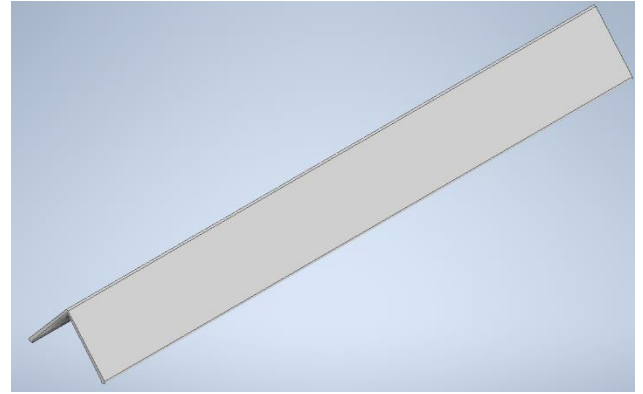


Figur 6 Innluft seksjon. Egenprodusert



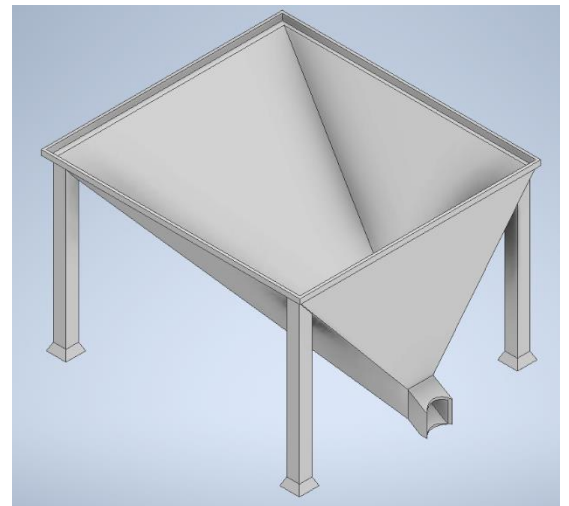
Figur 75 innluft hel vegg. Egenprodusert

På figur 8 ser vi takformene som går imellom inn- og utluft veggene i etasjene gjennom hele tørken.



Figur 6 takform til luftkanal. Egenprodusert

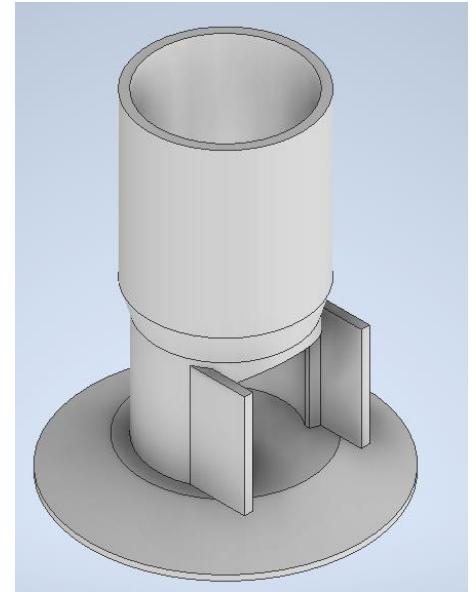
Når vi hadde funnet ut størrelsen på hele firkanten av selve tørken kunne vi begynne å lage spissbunnen. Her begynte vi virkelig å få bruk for sammenstillingsfilen hvor hele modellen blir satt sammen digitalt. Da var det veldig enkelt å tegne opp omrisset av tørkefirkanten og så lage en bunn som skrådde innover og til en side slik at kornet kan renne pent ned igjennom tørken og ut til en skrue som skal løfte den opp.



Figur 7 Spissbunn. Egenprodusert

Når selve spissbunnen var tegnet fant vi ut hvor vi får ut kornet som skal gå inn på skruen. Her også var det veldig nyttig med sammenstillingsfilen for å se at inntaket på skruen og uttaket på tørken kom riktig i forhold til hverandre. Det ga oss også muligheten til å lage noen vinger på bunnen til skruen slik at spissbunnen og skruen kan presses inn mot hverandre og stå stødig. Når vi da hadde tegnet bunnen til skruen kunne vi sette beina på spissbunnen slik at den sto i riktig høyde og skruen passet inn på spissbunnen.

På figur 10 ser vi bunnen på skruen. Grunnen til at denne er så kort er fordi vi fant ut at der var bedre å kjøpe ett glatt plastrør og bruke det som selve røret til skruen istedenfor for å 3D-printe dette. Vi prøvde først å 3D-printe det, men da ble det for høy friksjon inni røret til at den motoren vi hadde klarte å dra rundt skruen. Basen i bunn er stor og rund slik at skruen skal stå støtt. Dette er viktig siden den skal skru 50 cm oppover og ha motoren på toppen av det igjen. I tillegg får den god støtte av spissbunnen som står inne i de flappene som kommer ut ved inntaket.



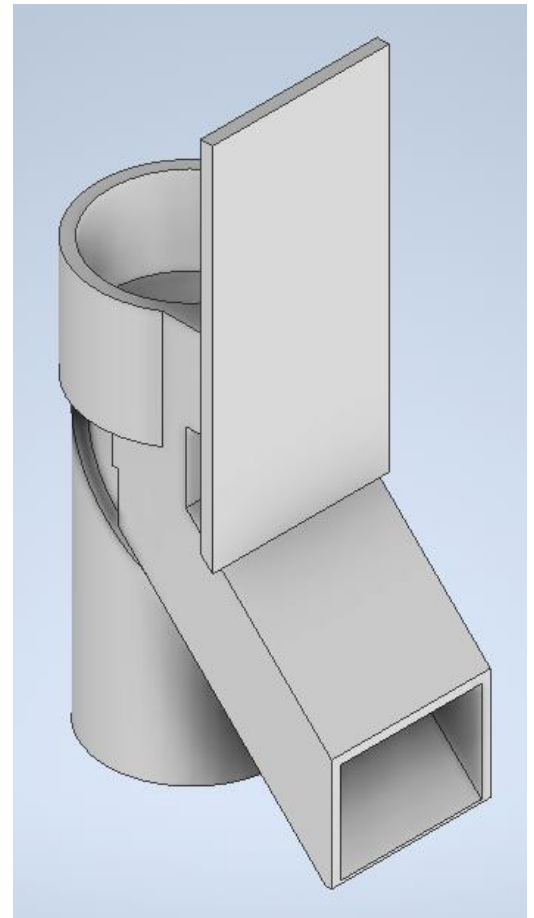
Figur 8 bunn til skruer. Egenprodusert

Vi har testet dette med å bruke skruer til å løfte kornet flere ganger. Den første gangen brukte vi en kort modell bare for å se om konseptet fungere. Det gjorde det. Vi klarte å løfte både ris- og maiskorn med den skruen. Så lagde vi en full lengde skruer med 3D-printa rør. Det fungerte sporadisk, men kilte seg relativt ofte og for å få den i gang igjen da måtte vi ta skruen ut av røret og helle ut alt som var inni. Spesielt med ris var dette et problem siden risen var så liten at den la seg mellom vingene på skruen og røret. Da røret i tillegg var litt ruglete etter 3D-printingen så ble friksjonen så stor at alt stoppet.

Vi gikk derfor over til å bruke ett hardplastrør og bare 3D-printe bunn- og toppstykket. Dette fungerte bedre, spesielt med maisen. Men da fant vi et nytt svakt punkt som også gjør mais uegnet til dette. Maiskornene er så store at de kiler seg i toppen på inntaket til skruen. Noen ganger kommer et maiskorn som legger seg akkurat i toppen på inntaket akkurat i på midten i det skruen skal gå inn i røret sitt. Da kiler alt seg og stopper.

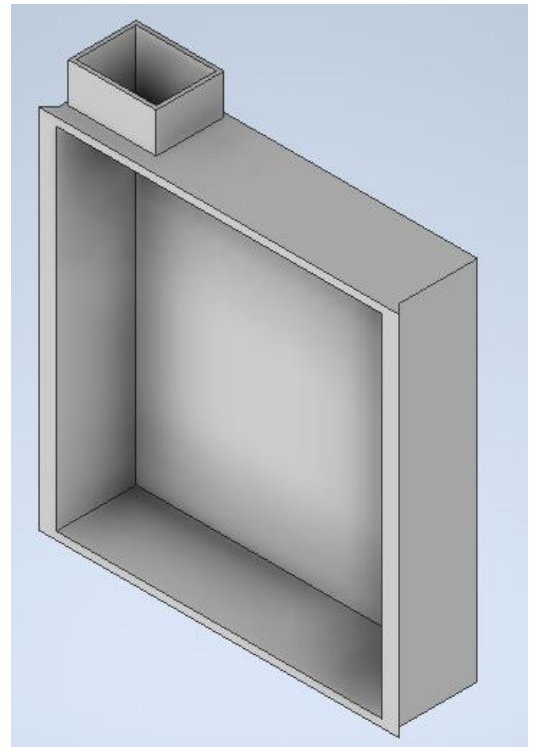
En stor del av problemet med at det kiler seg er at vi til nå har brukt en veldig liten elektromotor. Denne har to problemer. Det ene er at den ikke har spesielt stort moment og det andre er at den har ekstremt rask turtall. Vi kan skru ned hastigheten ved å redusere volten men det igjen reduserer dreiemomentet som igjen gjør at det raskere kiler seg. Løsningen på dette problemet var å bytte ut den lille 5V-motoren med en større motor som bruker mellom 6 – 24 volt og hadde stor utveksling slik at den gikk med en hastighet på sikre 120 omdreininger i minuttet med 24 V. Dette fungerte vesentlig bedre. Med denne motoren fikk vi et problem med at skruen knakk akkurat der åpningen sluttet og skruen går inn i røret. Dette ble løst med å bruke forredikk frøene som nevnt over, siden disse heller vek unna enn å knekke skruen slik som korn og mais gjorde.

På figur 11 ser vi toppen av skruen. Her kommer kornet opp og når det når toppen blir det ledet ut kanalen som vi ser på figuren. Motorfestet er her integrert og skruen er lagret opp i senter av motorfeste. Motoren sitter oppe i rundingen, og basen på motoren ligger mot den loddrette platen slik at når motoren blir belastet står den fortsatt stille og roterer skruen. Denne måten å feste motoren på gjør det veldig enkelt å ta det fra hverandre når skruen tetter seg eller vi vil tømme den.



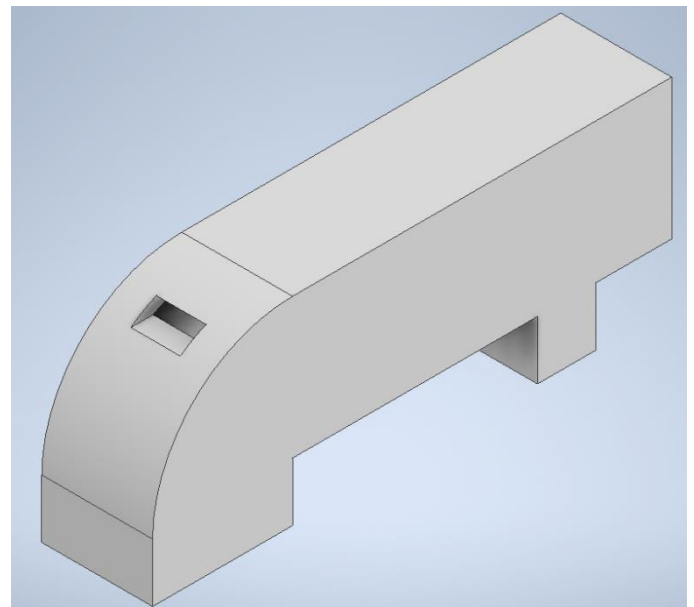
Figur 9 topp til skru. Egenprodusert

Så begynte vi med luftdelen av tørka der vi skal gjør de faktiske målingene av temperaturen og tørkeluften. På denne figuren ser vi hvordan kassen som er limt på utluftveggen av tørka ser ut. Luften vil komme inn og gå ut igjennom hullet i toppen. Grunnen til at dette ikke er printet som en del er fordi de store flatene er veldig vanskelige å printe over printeplaten. Disse platene blir limt til inn- og utluftveggen slik at luften fanges opp inne i det store hulrommet og går enkleste vei, som på utluften er opp i gjennomhullet i toppen. Et problem, spesielt på innluft-siden, er at luften blir så varm at den gjør at plasten bøyer og vrir seg slik at limet sprekker. Dette gjør at vi ikke kan bruke veldig sterke varmekilder og at vi er avhengig av å bruke lim som er sterkt og noe elastisk slik at det ikke bare sprekker.



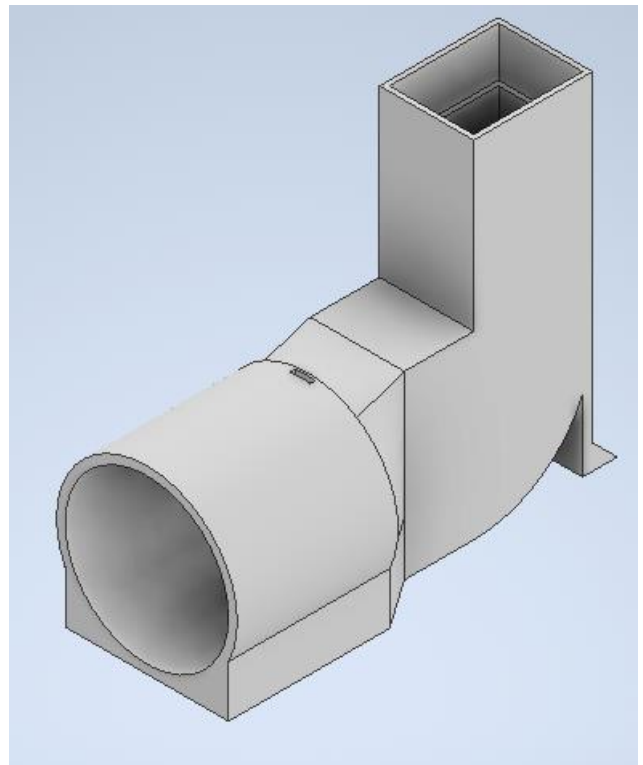
Figur 10 kasse til utluft. Egenprodusert

På figur 13 ser vi kanalen som sitter oppå utluft kanalen denne er for å fange opp utlufta og bende den 90 grader slik at all lufta må over sensoren som sitter i den åpne slissa. Det er her vi skal gjøre de faktiske målingene av utlufta som skal gi oss informasjon om hvor mye vann som blir transportert vekk og da igjen kunne si noe om når kornet er tørt nok. Sensoren vil bli satt inn slik at tilkoblingspinnene stikker ut av slissen og vi kan koble oss til dem. Den vil bli fuget fast med lim slik at den sitter fast og er luft tett. Denne sitter oppå tørka og dytter den fuktige utlufta vekk fra inntaket.



Figur 11 bend til utluft med feste til sensor. Egenprodusert

På figur 14 ser vi tegningen av hvordan luften skal suges inn og måles. Den store tuben ytterst er der vi skal sette inn hårføneviften, og for at den skal stå riktig vei måtte vi ha motoren innover. Derfor er det en liten slisse i toppen slik at vi kan få ut strømledningene til den. Etter at luften har blitt akselerert av viften går den bort mot svingen som går oppover. Ytterst i dette bende skal DHT 22-sensoren sitte og måle luft temperaturen og fuktigheten på innlufta, som er det vi skal basere oss på for å bestemme om kornet er tørt. Man ser ikke på denne tegningen hvor sensoren skal settes inn, men det er oppunder der man ser kanalen bøyer oppover. Denne sensoren skal også settes fast med limes fast slik at luft ikke lekker rundt den. Denne biten skal stå på bakken og være med å støtte opp konstruksjonen.

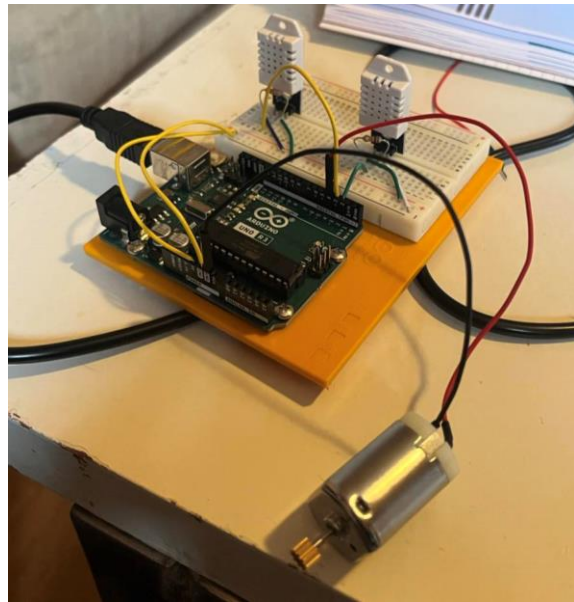


*Figur 12 bend til innluft med feste til vifte og sensor.  
Egenprodusert*

## 3.2 Styling og kode

Når tørken var ferdig designet, printet og vi hadde sett at den fungerte slik som vi tenkte kunne vi begynne med selve styringen.

Vi startet med å få begge DHT 22-sensorene til å sende ut data og utover det starte en motor. I starten har vi bare koblet på motoren slik at vi ser at den går rundt når de rette kriteriene er møtt. Det første vi gjorde av konkret bygging og koding var å koble alt sammen i kontaktbrettet vi har sammen med Arduinoen. Dette er for å se at det fungerer slik vi har tenkt det skal fungere både i virkelige scenarier, men også på modellen vi bygger.



På bilde ser hvordan dette er koblet. Her har vi to DHT 22-sensorer koblet opp slik at vi kan lese temperatur og luftfuktighet i to

*Figur 13 arduino med to sensorer og motor. Egenprodusert*

forskjellige scenarier. Det er dette vi skal utnytte for at automasjonen vet når kornet er tørt. I det oppsettet her er koden slik at når sensor 1 har lik eller høyere temperatur og luftfuktighet sammenlignet med sensor 2 så starter motoren og motoren fortsetter å gå uavhengig av hvilke sensorverdier den får etter den har fått kriteriene oppnådd en gang.

Vi gjør dette for å se at alt fungerer, men vi ser også at sensorene klarer å lese like verdier når de står sammen i et identisk miljø.

Etter dette satte vi sensorene inn i modellen og koblet de opp slik at de kommuniserer med Arduinoen. Og igjen sjekket vi at vi fikk ut målinger fra sensorene. For å gjøre dette enklere satte vi inn en LCD-skjerm for å vise sensordataene i sanntid, dette gjør det enklere for oss å se at alt fungerer og enklere for andre og se hva som skjer og hvordan det fungerer.

Når vi hadde koblet sensorene og skjermen gikk vi over til det vi må ha for å få modellen til å simulere tørkingen, vifte og skrue. Først gjorde vi klar strømforsyningen og lagde oss en overgang slik at vi kunne skru dette sammen inn i releet og at vi fikk strøm til begge releene vi skal bruke. Jordingen som går utenfor ble bare splittet slik at vi fikk to jordingskabler som kobles til strømforsyningen med kabelsko.

Når releet hadde fått hovedstrømmen sin, koblet vi til styrestrømmen fra Arduinoen slik at vi kunne styre om viften og skruen får strøm fra Arduinoen, og til slutt koblet vi sammen strømledningene til viften og skruen innpå hver sin releutgang. Releet ble da koblet slik at når det fikk styrestrøm sendte den hovedstrømmen videre til motorene.



Når vi hadde kommet hit ble det skrevet en kode for å aktivere sensorene og det ene releet slik at vi kunne teste at alt fungerte da bare byttet vi om ledningene på styrestrømmen for å kunne se at releene til begge motorene virker slik vi har tenkt.

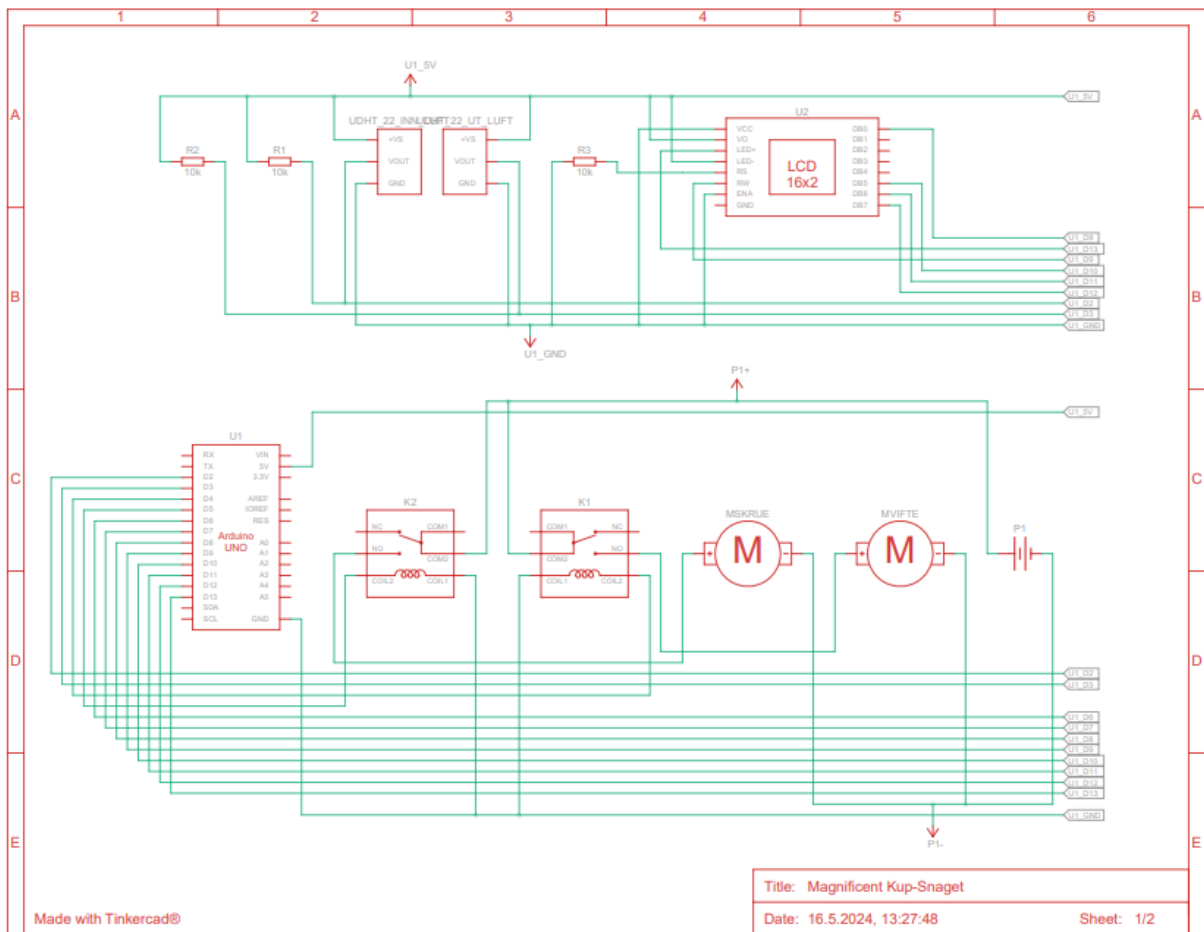
Mens vi drev med dette fant vi ut at vi ville ha trykkebrytere slik at vi kan manuelt skru av og på begge releene. Dette gjør det mye enklere for oss å ha alt oppkoblet hele tiden og at vi slipper å dra ut og sette i ledninger for å styre enkle ting. Da vi fikk koblet opp dette var vi klare til testing av prinsippet.

Når vi hadde begynt å teste litt fant vi fort ut at vi måtte regne om relativ luftfuktighet til spesifikk luft fuktighet, siden dette gir oss et bedre sammenlikningsgrunnlag som ikke blir påvirket i samme grad av temperaturen på luften. Dette gjorde vi med formelen for spesifikk luftfuktighet nevnt i innledningen

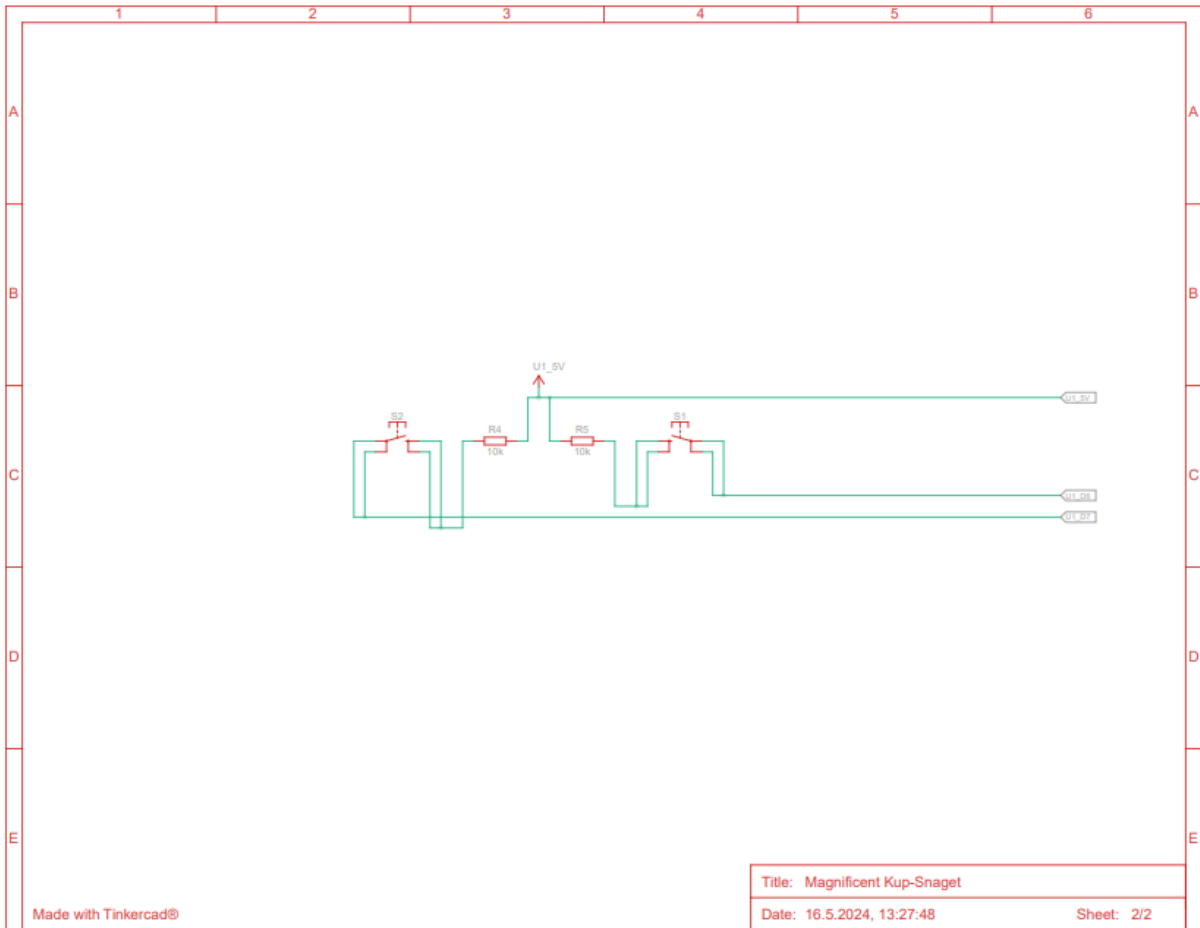
### **3.2.1 Ledningsnett**

Ledningsnettet vi har tegnet med Tinkercad viser på en strukturert måte hvordan vi har koblet de forskjellige komponentene. Når vi har koblet på modellen i virkeligheten har vi brukt et koblingsbrett slik at det er lett å koble om hvis det blir koblet feil, og det gjør det også lettere

å få testet forskjellige koblingsmetoder.



Figur 14 ledningsnett til sensoroppsett. Egenprodusert



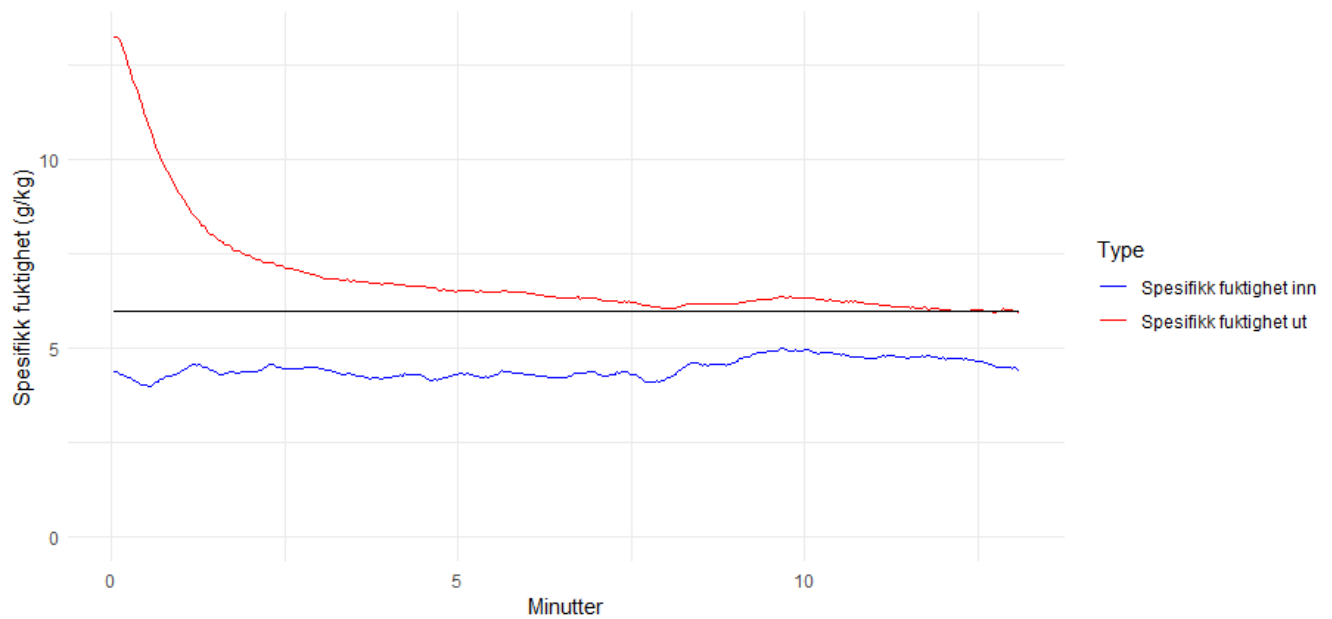
Figur 15 ledningsnett til sensoroppsett. Egenprodusert

### 3.3 Testing



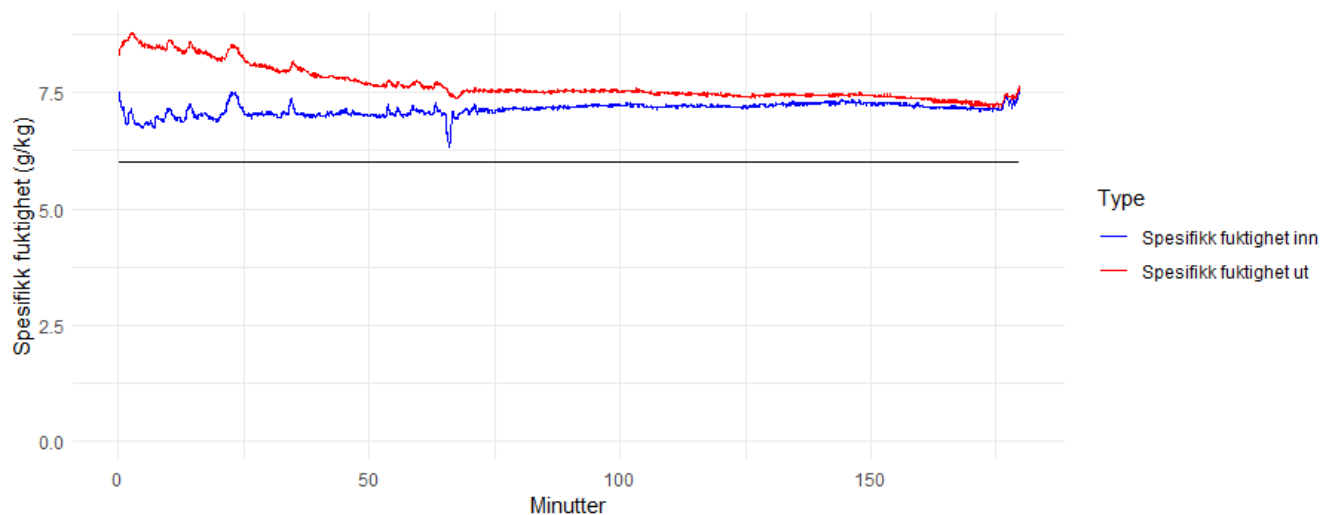
*Figur 16 ferdig tørke med styring. Egenprodusert*

Vi har kjørt tørken mange ganger og tilpasset koden litt etter hvert for å få den til å lese dataene mer presist. Den siste store tilpassingen vi gjorde var å regne om fra relativ luftfuktighet til spesifikk luftfuktighet slik at vi kan se hvor mange gram vann det er i hver kilo luft som går ut gjennom tørken. Med dette har vi laget noen grafer over tørkeforløpene.



Figur 17 testresultater tørke nr 6. Egenprodusert

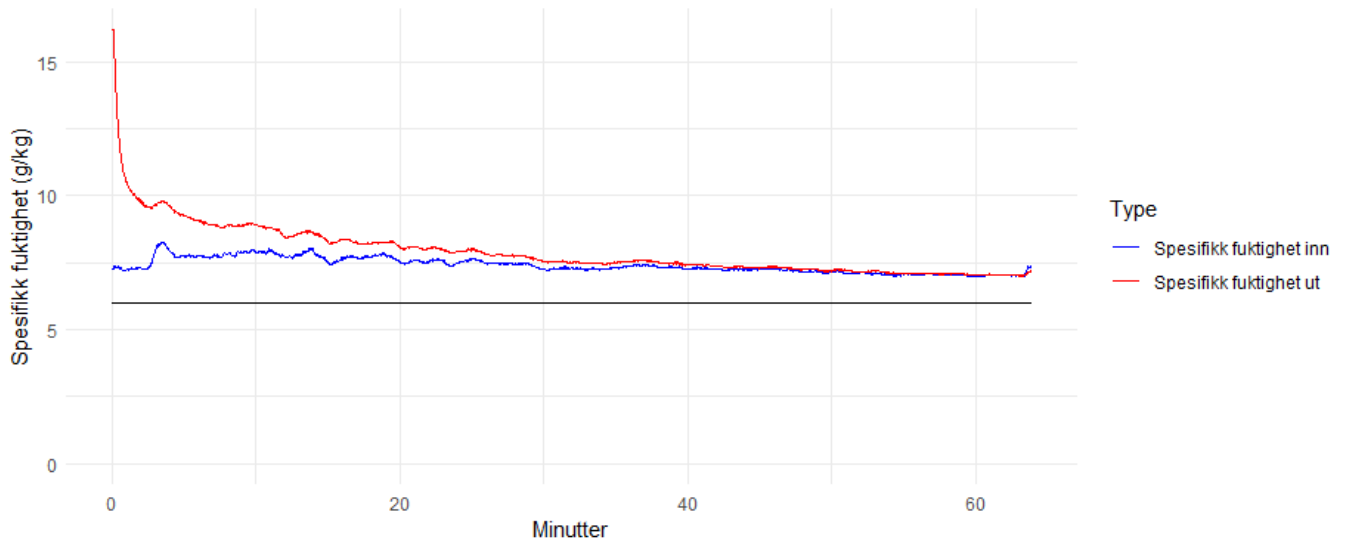
Dette var tørke nummer 6 med hvete. Den blå linjen er den spesifikke luftfuktigheten på innluften og den røde er den spesifikke luftfuktigheten på utluften. Her tenkte vi at tørken skulle være tørr og stoppe når den spesifikke luftfuktigheten på utluften var under 6 g/kg. Når vi stoppet og målte vanninnholdet var dette på 13% målt med vannmåler som er godt innfor.



Figur 18 testresultater tørke nr 7. Egenprodusert

Dette er tørke nummer 7 med hvete. Denne tørken ble kjørt noen dager etter tørke 6. Da hadde fuktigheten i rommet økt og dermed kom aldri innluften ned på det samme som den hadde hatt på tørke 6. Dette gjorde at vi aldri nådde stopplinjen på 6 g/kg og tørken bare fortsatte å gå. Når vi stoppet den manuelt og målte denne med vannmåler var fuktigheten på 6% vann målt

med vannmåler. Altså altfor tørt. Men som vi ser på grafen så tørket kornet hele tiden, men siden innluften var for fuktig nådde vi aldri grensen som vi hadde satt.



Figur 19 testresultater tørke nr 8. Egenprodusert

Her ser vi tørke nummer 8 med hvete. Denne ble kjørt samme dag som nummer 7 og her ser vi det samme. I denne tørken var ikke kornet like vått så det gikk fortere, men vi nådde heller ikke her grensen vår på 6 g/kg fordi innluften var for våt. Når vi stoppet denne og tok prøve av den var fuktigheten på kornet 9% målt med vannmåler.

## 4. Diskusjon

Nå har vi kommet til den mest interessante delen av oppgaven. Her skal vi se tilbake på hva vi har gjort og hvordan vi eventuelt kunne gjort dette annerledes. Vi skal også se litt hvordan dette kan tas videre i fullskala og om det kan være aktuelt og gjøre noe liknende i andre operasjoner i landbruket.

### 4.1 Modellering

Vi begynner med modelleringen siden det var dette vi startet med. Selve metoden ved modelleringen kunne vi ikke gjort på noen annen måte. Og det gikk egentlig lettere enn forventet å få alt til å passe så godt at vi ikke trengte å lime delene sammen. Dette er i all hovedsak Inventor som har gjort mulig siden det lar oss stille tegningene sammen inne i programmet før vi printet det ut. Hadde vi måttet printe alt ut og så gjøre tilpasninger for så å printe en ny versjon ville dette gjort jobben hvert fall dobbelt så tidkrevende. Hvis vi skulle gjort noe tilsvarende på nytt ville det nok vært fordelaktig å tegne en grovt målsatt skisse på papir som man kunne følge når man gjorde selve 3D-tegningen. Det gikk greit i dette tilfellet siden vi hadde en god forståelse av hvordan produktet skulle se ut når det var ferdig da vi har sett flere i fullskala i virkeligheten. Derimot skulle vi laget noe som ikke finnes fra før vil nok en skisse vært veldig nyttig.

På selve modellen er det noen endringer som ville gjort den mer egnet for testing slik vi har brukt den. Spissbunnen er et problem siden dette siktet ikke blir tørket, men kornet bare ligger der og venter på å bli rullert opp. Slik er det ikke på virkelige tørker siden disse har rulleringspjeld under tørkesonen. Dette derimot ble utrolig vanskelig og tidkrevende for oss å konstruere til modellen vår. Derfor valgte vi å se bort i fra dette. Siden vi ikke har dette rulleringspjeldet må vi fylle tørt korn i spissbunnen slik at vi er sikre på at den ikke tetter seg når vi skal tømme den igjen. Dette gjør det vanskelig å være sikker på at alt kornet vi tar vannmålinger av er det som har vært fuktet og tørket, og ikke er det som har ligget i spissbunnen som buffer. Skulle vi gjort forbedringer som vil gjort dette enklere ville vi droppet spissbunnen og skruen, og heller hatt en bunn som vi kunne slippe ned en liten mengde korn fra tørken på, uten at vi måtte stoppe og tømme alt. Her ville vi antageligvis ha laget en flat bunn med to skyvespjeld slik at vi kunne sluppet ned ca 3 cm av det i hele tørkesonen og dermed ta ut en prøve mens tørken går.

## 4.2 Kobling og koding

Koblingen og kodingen har vært en lang og dynamisk prosess både før og sammen med testingen. Her har vi brukt mye tid på å forme koden og lagt til eklektiske komponenter slik at vi enkelt både kan styre og se hva som skjer i selve tørkeprosessen. Vi hadde to ting vi la på etter hvert som gjorde det mye enklere å gjennomføre testingen.

### 4.2.1 Brytere

Etter at vi hadde kjørt den første testen fant vi ut at vi hadde stort behov for en mulighet til å stoppe viften uten at vi dro ut strømmen til tørken og dermed stoppet loggingen av tørken. Samtidig fant vi ut at vi ville legge på en knapp slik at vi kunne stoppe og starte skruen som rullerte kornet når vi ville. Vi satte på en trykkebryter til hver av disse funksjonene. Fordelen med disse bryterne er at de står i nøytralposisjon hele tiden. Slik at de bare endrer når de blir trykket inn. Dette gjør at vi enkelt kan styre motorene samtidig fra verdiene på sensorene og allikevel bruke bryteren helt uavhengig. Dette var spesielt fordelaktig når tørken stoppet etter at luften kom innenfor de parameteren der den skulle stoppe, og vi da kunne dobbeltsjekke dette ved å starte viften igjen.

### 4.2.2 LCD skjerm

Når vi hadde kjørt to tester fant vi ut at det var veldig vanskelig å følge med på utviklingen i tørkeforløpet i tekstfilen alle datapunktene ble logget til. Derfor la vi til en skjerm som viste dataene fra begge sensorene ved siden av hverandre i sanntid. Dette gjorde det mye enklere for oss å følge med på tørkeforløpet mens den holdt på å tørke. Noe som igjen både ga oss bedre innsikt i hva som skjedde i selve tørken, og gjorde det mye enklere å forklare andre som kom og så på hva vi drev med hva som skjedde. Vi begynte med å vise temperaturen og den relative luftfuktigheten, men på den ferdige koden viser vi bare spesifikk luftfuktighet på inn- og utluften, slik at vi enkelt kan se at kornet fortsatt tørker og hvor stor forskjellen er, som igjen sier oss noe om hvor nærme vi er at kornet er tørt.



### 4.2.3 Koding og KI

Vi har brukt mye tid i denne oppgaven på koding, og spesielt tilpassing av koding. Vi har i denne forbindelse brukt en del KI som sparringspartner for å få koden dit vi trengte den. Vi har gitt KI hva vi vil oppnå, fått en kode så har vi sjekket og testet denne. Da enten redigert manuelt i koden eller sendt den tilbake til KI med hvilke modifikasjoner vi vil gjøre.

Eksempel på hvordan vi direkte har brukt KI ser du i «Vedlegg 3 eksempel på bruken av ChatGPT til koding». Denne metoden har vi brukt om igjen og om igjen for å forme koden slik vi vil ha den. Men vi begynte først med å lage den koden som vi brukte til å se at vi fikk koblet opp en sensor og kunne lese av denne selv. Denne koden er vedlagt i «Vedlegg 1 koble opp en sensor».

Den komplette koden vi ente med er den vi har brukt på de 3 siste tørkeforløpene som er det som er vist i resultat delen. For å få den slik måtte vi til slutt endre en del på koden for å få den slik at den fungerte optimalt. Spesielt til å sette opp LCD-skjermen er KI dårlig på. KI klarer ikke holde seg innenfor hvor mange tegn som skjermen klarer å vise. Dette gjør at vi manuelt måtte endre hvilke tegn og hvor disse tegnene skulle stå slik at alt vi vil se fikk plass på skjermen.

Koden er selve hjerte i tørken. Det er den som logger og lager datapunktene samtidig som den sender strøm slik at de motorene som skal gå går. I tillegg ser den hele tiden etter om parameterne for å gjøre endringer blir oppnådd. Hvis man har komponenter og kan å kode er det bare fantasien som setter begrensninger på hva man kan klare å automatisere, spesielt innenfor ting som enten skal være på eller av. Her er KI et utrolig godt hjelpemiddel siden den lar folk som ikke er så flinke til koding automatisere ting den personen forstår. Dette gjør at de som vet hva som trengs får midlene og mulighetene til å kunne utvikle og lage automasjon selv.

### 4.2.4 Beregninger

Vi gjør en omregning i denne oppgaven og det er å regne om temperatur og relativ luftfuktighet til spesifikk luftfuktighet slik at vi får en verdi som er sammenlignbar uavhengig av temperaturen. Måten vi gjøre dette på er med måten som er beskrevet i innledningen i kapitlet målinger. Denne formelen bruker en del konstanter som gjør det mulig å regne om den relative fuktigheten til spesifikk fuktighet. Vi bruker også vanlig atmosfærisk trykk ved 0

meter over havet som konstant. Det er ikke sikkert at alle disse konstantene gir et helt riktig og nøyaktig resultat, men siden vi bruker de samme konstantene til begge beregningene blir de rimelig like selv om det kanskje ikke blir helt riktig. Siden det viktigste er at vi sammenlikner verdiene, er det viktigste at de er beregnet og målt på samme måte slik at vi enkelt og med rimelig sikkerhet har riktig differanse. Til videre utvikling og testing vil denne beregningsmåten være noe man kan se på for å øke nøyaktigheten og presisjonen må målingene slik at de bedre treffer på hvor tørt kornet er.

## 4.3 Testing

Testingen har fungert bare og gitt oss det vi trenger for å se om prinsippet kan fungere i fullskala og i praksis. Men det er flere ting her som er verdt å se nærmere på.

### 4.3.1 Kornets fuktighet

Først kan vi se litt på måten vi har kunnet teste. Fordi for å ha noe å teste må vi har rått korn. Dette er ikke så lett å finne i perioden januar til april hvor vi har bygget og gjort testene som er omtalt i denne oppgave. Derfor var vi nødt til å kunstig fukte kornet. Dette ble gjort ved å helle vann i en pose med kornet i og riste det rundt til vannet var absorbert i kornet. Dette vil selvfølgelig ikke gi lik «fuktighet» som ved naturlig rått korn. Dette kornet er rått helt inn til kjernen, mens det kornet vi har laget til har vært mest rått utenpå og mindre innover mot kjernen. Dette vil antageligvis påvirke tørkeprosessen og målingene noe, men vi tror vi vil kunne se like godt at luften blir tørrere i kornet med naturlig rått korn.

Vår forventning er at grafen vil bli noe mer jevne som i figur 19. ikke like hakkete som i figur 20 og 21. Dette er fordi fuktigheten er noe hardere bundet, og dermed vil den bli sluppet ut jevnere. Det kan godt tenkes at dette vil gjøre det enklere å måle når kornet er tørt utfra luftfuktigheten, siden det går saktere og luften tar mer jevnt opp fuktigheten, i motsetning til det kunstig fuktete kornet som slipper fuktigheten rundt skallet veldig fort og dermed blir tørt rimelig fort. Det kan også tenkes at det blir vanskelig å se når kornet er å riktig fuktighet helt igjennom siden når det innerste i kjernen på kornet skal tørkes slippes dette ganske sakte ut igjennom kornet og det kan bli vanskeligere å klare å se forskjellen i fuktigheten på luften.

Her vil det helt klart være fordel med en varmluftstørke der luften er ekstremt tørr, hvor luften tørker veldig effektivt og det skal frigjøres lite fuktighet for å gjøre utslag. Her kreves det mer testing for å kunne si akkurat hvordan dette vil fungere i praksis.

### **4.3.2 Sensorer**

Sensorene er en mulig feilkilde. DHT 22-sensorene er ikke av de aller billigste og enkleste på markedet, men de er heller ikke av de dyreste kalibrerte og veldig nøyaktige sensorene som man kan få tak i. DHT 22-sensoren er en masseprodusert og relativt enkel sensor å bruke og den passer til de fleste oppgaver siden den har et veldig bredt spekter den opererer i. Til vårt formål har den fungert greit. Vi har fått målinger som har sett fornuftige ut hele veien, og vi har ikke vært avhengig av at sensorene er ekstremt nøyaktige. Dette er fordi vi sammenligner verdiene inn og ut. Sensorene var like før vi begynte så om de måler litt feil, så måler de i alle fall likt feil slik at forholdet imellom dem er likt. Dette gjør at disse sensorene virker greit til vårt formål når vi tester dette prinsippet på ett tidlig stadie. Når man etter hvert tar dette videre og ser hvordan de håndterer en virkelig tørke med større luftmengde og masse støv er det mulig man burde se seg om etter mer presise og kanskje spesielt mer robuste sensorer som tåler å stå i ett røft miljø over en lengre periode. Samtidig er disse sensorene såpass billige at man kan godt bytte de flere ganger i sesongen uten at det ville være noen utgift å snakke om. Dette må vurderes fortløpende etter hvert som videre testing pågår, men i denne sammenhengen har disse sensorene fungert utmerket.

## 4.4 Tidsbruk

Hele denne oppgaven har blitt ganske stor i tidsbruken vi har brukt på å konstruere, koble og teste det vi trenger før vi kunne begynne å virkelig skrive oppgaven.

### 4.4.1 Tegning og 3D-printing

Vi begynte med tegningen i desember. I startet gikk det mye tid til 3D-printing og tegning. Dette er en relativt tidkrevende prosess siden det tar en del tid for 3D-printeren å printe de forskjellige tegningene. Man kommer heller ikke noe videre før man får printet den delen man har tegnet ut og sett hvordan den blir. Selvfølgelig kan man bare tegne på og stille det sammen i 3D-programmet, men ofte er det til stor hjelp å se hvordan de forskjellige komponentene er i virkelig verden før man går videre. Ofte ser man ikke før man har den fysiske delen hvordan man best kan tegne den neste. På grunn av dette er selve modelleringen og konstruksjonen en relativt tidkrevende prosess som vi brukte omtrent fra desember til tidlig april med kontinuerlig jobbing hver uke for å fullføre.

### 4.4.2 Styring og koding

Vi begynte med styringen og kodingen mot slutten av konstruksjonsperioden altså omtrent fra mars. Styringen gikk vesentlig fortere enn konstruksjonen. Siden her trengte vi ikke vente flere timer eller dager på at en komponent skulle bli ferdig, men vi isteden kunne jobbe kontinuerlig med utviklingen. Vi kunne teste og modifisere samtidig. Dette ble vi ferdig med i slutten av april.

### 4.4.3 Testing

Testingen var det som gikk desidert raskest. Vi begynte med testingen i midten av april og drev med dette i flere dager utover i mai. Testingen gikk fort siden når alt var ferdig laget kunne vi kjøre flere tørkeforløp på en dag. Dette gjorde at vi kunne samle inn all dataene vi trengte på få dager.

## 4.5 Veien videre

Til slutt kommer vi frem til hva vi tenke veien er videre. Er det noe potensiale for dette prinsippet? Er det mulig å utvikle det videre til å bli en robust måte å måle fuktighet i kornet på i ett korntørke anlegg? Vi mener at det er det, og at utefra det vi har sett at prinsippet viser godt potensiale til å kunne fortelle ett styringssystem at kornet er tørt og tørken er klar for å kjøles eller kjøres om.

For å forme prinsippet og iden ut til et ferdig produkt som en bonde kan dra god nytte av vil det kreve mere testing. Til nå har vi brukt en fast verdi til å som sier når kornet er «tørt». Dette fungerer bare hvis vi har en jevn spesifikk luftfuktighet på inntaksluften. Dette kan man endre på ved å teste seg frem til en modell som lager en flytende grense som sier at kornet er tørt når utluften kommer innenfor det rette parametere i forhold til inluften. Dette har vi ikke begynt å teste i denne oppgaven, men vil være ett naturlig sted å fortsette når vi tar dette over i fullskala testing på et ekte kornanlegg.

Det er relativt enkelt å montere dette systemet i ett ekte fullskala tørkeanlegg. Egentlig kan det puttes inn helt likt som det vi har brukt i denne oppgaven. Som en start trenger man heller ikke å koble seg på det eksisterende styringssystemet til tørken. Men heller bruke dette systemet utelukkende som et overvåkende system på inn og ut luften på tørke for å se om det har de samme trekkene som vi ser i den lille modellen. For å se om det stemmer da må man måle kornet i tørken på vanlig måte mens man logger hele tørkeforløpet slik vi har gjort i denne oppgaven. Etter man har logget en del tørke forløp vil man begynne å se sammenhenger og utfra dette vil man kunne lage en modell som flytter grensen for tørt korn utfra fuktigheten i inntaksluften.

## 5. Konklusjon

For å konkludere må vi se tilbake på problemstillingen i oppgaven. Den var todelt, siden ene delen av oppgaven var å teste prinsippet om vi kan se på luftfuktigheten om kornet er tørt eller ikke. Den andre delen var å lage en miniatyrtørke for å teste dette prinsippet i.

Vi begynner med den andre delen, altså konstruksjonen av tørken vi skal teste prinsippet i. Vi har tegnet og 3D-printet alle delene fra bunnen av og utfra egne erfaringer med tørkeanlegg. Så har vi bygget en styring som passer til å styre denne tørken og logge de dataene vi trenger til å teste prinsippet vi skal teste i denne oppgaven. Tørken fungerer utmerket og tørker korn meget effektivt. Den har noen nedsider spesielt med at den ikke har rulleringspjeld, noe som gjør det litt vanskelig å få gode målinger ut av spissbunnen. Utover dette fungerer tørken veldig godt til sitt formål.

Så til den andre og viktigste delen av oppgaven. Fungerer det prinsippet om å si om kornet er tørt eller ikke utfra å sammenligne fuktigheten på inn og ut luften. Vi ser godt at fuktigheten forsvinner fra kornet, og blir med luften ut. Vi har også klart å sette en grense for å stoppe tørken når kornet er tørt, men denne har bare vært fast og ikke kompensert for at fuktigheten på innluften endrer seg. Vi har ikke enda klart å lage en modell som regulerer hvor grensen skal gå for hvilken fuktighet utluften skal ha når kornet er tørt. Men dette synes vi ser ut som et ganske overkommelig problem etter hvert som vi får testet mer. Derfor tror vi at prinsippet vil fungere, og etter en del mer testing, og modifisering vil vi klare å lage en formel som lar oss fortelle en styringsenhet når kornet er tørt utfra forholdet mellom inn- og utluften.

## 6. Litteraturliste

*Adafruit/DHT-sensor-library*. (2024). [C++]. Adafruit Industries. <https://github.com/adafruit/DHT-sensor-library> (Opprinnelig verk utgitt 2011)

*Arduino Uno Rev3*. (2024). Arduino Official Store. <https://store.arduino.cc/products/arduino-uno-rev3>

Fladstad, O., & Tengesdal, G. (2002). *Tørking og lagring av korn: Teknisk fagskole, linje for naturbruk : fordjupningsområde plantedyrking*. GAN forl. [https://urn.nb.no/URN:NBN:no-nb\\_digibok\\_2016020408204](https://urn.nb.no/URN:NBN:no-nb_digibok_2016020408204)

Helseth, L. E. (2024). Damptrykk. I *Store norske leksikon*. <https://snl.no/damptrykk>

*In-Depth: Interfacing DHT11 and DHT22 Sensors with Arduino*. (2018, juli 5). Last Minute Engineers. <https://lastminuteengineers.com/dht11-dht22-arduino-tutorial/>

Jokiniemi, H. T., & Ahokas, J. M. (2014). Drying process optimisation in a mixed-flow batch grain dryer. *Biosystems Engineering*, 121, 209–220. <https://doi.org/10.1016/j.biosystemseng.2014.01.002>

Koutsoyiannis, D. (2012). Clausius-Clapeyron equation and saturation vapour pressure: Simple theory reconciled with practice. *European Journal of Physics*, 33, 295–305. <https://doi.org/10.1088/0143-0807/33/2/295>

*Liquid Crystal Displays (LCD) with Arduino | Arduino Documentation*. (2023, februar 7). <https://docs.arduino.cc/learn/electronics/lcd-displays/>

Ltd, R. P. (2024). *Buy a Raspberry Pi 5*. Raspberry Pi. <https://www.raspberrypi.com/products/raspberry-pi-5/>

*Luxorparts Relemodul for Arduino—Moduler*. (2024). <https://www.kjell.com/no/produkter/elektro-og-verktoy/elektronikk/utviklerkit/arduino/moduler/luxorparts-relemodul-for-arduino-4x-p87033>

Nugent, A. (2024). *Chapter 1: Atmospheric Basics*. <https://pressbooks-dev.oer.hawaii.edu/atmo/chapter/chapter-1/>

Nugent, A., & Russell, S. (2024). *Chapter 4: Water Vapor*. 9.

*PLA Plastic/Material – The Ultimate Guide*. (2023, januar 22). All3DP. <https://all3dp.com/2/what-is-pla-plastic-material-properties/>

*Pulse Width Modulation—An overview | ScienceDirect Topics*. (2024). <https://www.sciencedirect.com/topics/engineering/pulse-width-modulation>

Reykdal, O., Authors, H., & T, D. (2018). *Drying and Storage of Harvested Grain—A Review of Methods*. <https://doi.org/10.13140/RG.2.2.14530.89286>

*RS PRO Brushed Geared DC Geared Motor, 1.31 W, 12 V dc, 10 Ncm, 66 rpm, 4mm Shaft Diameter | RS PRO*. (2024). RS Components International. <https://int.rsdelivers.com/product/rs-pro/rs-pro-brushed-geared-dc-geared-motor-131-w-12-v/4130622>

Samuelsen, E. (2023). Relativ fuktighet. I *Store norske leksikon*. [https://snl.no/relativ\\_fuktighet](https://snl.no/relativ_fuktighet)

## 7. Vedlegg

Vedlegg 1 Koble opp en sensor	Side 47
Vedlegg 2 Komplette kode for tørken	Side 48 - 53
Vedlegg 3 Eksempel på bruken av chatgpt til koding	Side 54 - 55



## Vedlegg 1 koble opp sensor

```
#include <DHT.h>

#define DHT_PIN 5

DHT dht(DHT_PIN, DHT22);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  delay(2000);

  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();
  if (isnan(temp) || isnan(fukt)) {
    Serial.println("Feil ved lesing av sensor");
    return;
  }
  Serial.print("Temperatur: ");
  Serial.print(temp);
  Serial.print(" °C, Luftfuktighet: ");
  Serial.print(fukt);
  Serial.println(" %");
}
```

## Vedlegg 2 Komplette kode for tørken

```
#include <DHT.h>

#include <LiquidCrystal.h>

#define DHTPIN1 2 // Definerer pinne for første DHT22 sensor
#define DHTPIN2 3 // Definerer pinne for andre DHT22 sensor
#define DHTTYPE DHT22 // Definerer typen av DHT sensor (DHT22)

DHT dht1(DHTPIN1, DHTTYPE); // Oppretter et DHT-objekt for første sensor
DHT dht2(DHTPIN2, DHTTYPE); // Oppretter et DHT-objekt for andre sensor

#define RELAY_PIN1 4 // Definerer pinne for relé 1
#define RELAY_PIN2 5 // Definerer pinne for relé 2
#define BUTTON_PIN1 6 // Definerer pinne for trykknapp 1
#define BUTTON_PIN2 7 // Definerer pinne for trykknapp 2

LiquidCrystal lcd(13, 8, 9, 10, 11, 12); // Starte opp LCD-skjerm

bool relayState1 = false; // Variabel for relé 1 sin tilstand
bool relayState2 = false; // Variabel for relé 2 sin tilstand

unsigned long lastTime = 0; // Variabel for å lagre siste tid sendt over serieporten

int consecutiveUnderLimitCount = 0; // Teller for antall påfølgende målinger under grensen
const int consecutiveLimit = 10; // Grense for antall påfølgende målinger under grensen

void setup() {
  Serial.begin(9600); // Starter seriekommunikasjon med baud rate 9600

  pinMode(RELAY_PIN1, OUTPUT); // Setter pinne for relé 1 som utgang
```

```

pinMode(RELAY_PIN2, OUTPUT); // Setter pinne for relé 2 som utgang

pinMode(BUTTON_PIN1, INPUT_PULLUP); // Setter pinne for trykknapp 1 som inngang med pull-up
motstand

pinMode(BUTTON_PIN2, INPUT_PULLUP); // Setter pinne for trykknapp 2 som inngang med pull-up
motstand

dht1.begin(); // Starter første DHT22 sensor
dht2.begin(); // Starter andre DHT22 sensor

lcd.begin(16, 2); // Starte LCD-skjermen med 16 kolonner og 2 rader
lcd.clear(); // Rydder LCD-skjermen
lcd.print("T1: --, H1: --"); // Skriver teksten på første linje
lcd.setCursor(0, 1); // Flytter markøren til starten av andre linje
lcd.print("T2: --, H2: --"); // Skriver teksten på andre linje
}

// Funksjon for å beregne spesifikk luftfuktighet basert på temperatur (t) og relativ luftfuktighet (h)
float calculateSpecificHumidityinn(float t1, float h1) {
    float tempInKelvin = t1 + 273.15; // Gjør om temperatur fra Celsius til Kelvin
    float saturationVaporPressure = 6.112 * exp((17.67 * t1) / (t1 + 243.5)); // Metningstrykket for
vanndamp
    float actualVaporPressure = saturationVaporPressure * (h1 / 100); // Aktuelt vanndamptrykk
    float specificHumidity = 0.622 * (actualVaporPressure / (1013.25 - actualVaporPressure)); //
Beregning av spesifikk luftfuktighet
    return specificHumidity;
}

// Funksjon for å beregne spesifikk luftfuktighet basert på temperatur (t) og relativ luftfuktighet (h)
float calculateSpecificHumidityout(float t2, float h2) {
    float tempInKelvin = t2 + 273.15; // Gjør om temperatur fra Celsius til Kelvin
    float saturationVaporPressure = 6.112 * exp((17.67 * t2) / (t2 + 243.5)); // Metningstrykket for
vanndamp
    float actualVaporPressure = saturationVaporPressure * (h2 / 100); // Aktuelt vanndamptrykk

```

```

float specificHumidity = 0.622 * (actualVaporPressure / (1013.25 - actualVaporPressure)); //
Beregning av spesifikk luftfuktighet

return specificHumidity;
}

```

```

void loop() {
// Leser temperatur fra første sensor
float t1 = dht1.readTemperature();
// Leser luftfuktighet fra første sensor
float h1 = dht1.readHumidity();

// Leser temperatur fra andre sensor
float t2 = dht2.readTemperature();
// Leser luftfuktighet fra andre sensor
float h2 = dht2.readHumidity();

// Beregn spesifikk luftfuktighet for inneluften
float specificHumidityIn = calculateSpecificHumidityinn(t1, h1);
// Beregn spesifikk luftfuktighet for uteluften
float specificHumidityOut = calculateSpecificHumidityout(t2, h2);

// Oppdaterer LCD-skjermen med temperatur og luftfuktighet fra begge sensorer
lcd.setCursor(3, 0); // Flytter markøren til kolonne 3 på første linje
lcd.print(t1); // Skriver temperatur fra første sensor
lcd.print("T2:"); // Skriver " T2 "
lcd.print(t2); // Skriver luftfuktighet fra første sensor
lcd.setCursor(0, 1); // Flytter markøren til kolonne 3 på andre linje
lcd.print(specificHumidityIn,5); // Skriver temperatur fra andre sensor
lcd.print(":"); // Skriver " :"
lcd.print(specificHumidityOut,5); // Skriver luftfuktighet fra andre sensor
}

```

```

// Skriver dataene til serieporten (Serial Monitor)
unsigned long currentTime = millis(); // Henter nåværende tid i millisekunder
if (currentTime - lastTime >= 2000) { // Sender data hvert sekund
    Serial.print(currentTime / 2000); // Deler på 1000 for å få tid i sekunder
    Serial.print(" , ");
    Serial.print(t1); // Sender temperatur fra første sensor
    Serial.print(" , ");
    Serial.print(h1); // Sender luftfuktighet fra første sensor
    Serial.print(" , ");
    Serial.print(specificHumidityIn, 5); // Sender spesifikk luftfuktighet for inneluften med fem
desimaler
    Serial.print(" , ");
    Serial.print(t2); // Sender temperatur fra andre sensor
    Serial.print(" , ");
    Serial.print(h2); // Sender luftfuktighet fra andre sensor
    Serial.print(" , ");
    Serial.println(specificHumidityOut, 5); // Sender spesifikk luftfuktighet for uteluften med fem
desimaler
    lastTime = currentTime; // Oppdaterer siste tid sendt
}

// Leser tilstanden til trykknapp 1
bool buttonState1 = digitalRead(BUTTON_PIN1);

// Leser tilstanden til trykknapp 2
bool buttonState2 = digitalRead(BUTTON_PIN2);

// Hvis trykknapp 1 er trykket og relé 1 er av, slå det på
if (buttonState1 == LOW && !relayState1) {
    relayState1 = true;
    digitalWrite(RELAY_PIN1, relayState1);
    delay(200); // Debounce delay
}

```

```

}
// Hvis trykknapp 1 er trykket og relé 1 er på, slå det av
else if (buttonState1 == LOW && relayState1) {
    relayState1 = false;
    digitalWrite(RELAY_PIN1, relayState1);
    delay(200); // Debounce delay
}

// Hvis trykknapp 2 er trykket og relé 2 er av, slå det på
if (buttonState2 == LOW && !relayState2) {
    relayState2 = true;
    digitalWrite(RELAY_PIN2, relayState2);
    delay(200); // Debounce delay
}

// Hvis trykknapp 2 er trykket og relé 2 er på, slå det av
else if (buttonState2 == LOW && relayState2) {
    relayState2 = false;
    digitalWrite(RELAY_PIN2, relayState2);
    delay(200); // Debounce delay
}

// Sjekker om spesifikk luftfuktighet for uteluften er under 0.006
if (specificHumidityOut < 0.0074) {
    // Øker telleren for påfølgende målinger under grensen
    consecutiveUnderLimitCount;

    // Hvis antall påfølgende målinger under grensen er større eller lik grensen
    if (consecutiveUnderLimitCount >= consecutiveLimit) {
        // Slår av relé 2
        relayState2 = false;
        digitalWrite(RELAY_PIN2, relayState2);
    }
}

```

```
delay(200); // Debounce delay

// Nullstiller telleren
consecutiveUnderLimitCount = 0;
}
} else {
// Nullstiller telleren hvis spesifikk luftfuktighet for uteluften er over eller lik grensen
consecutiveUnderLimitCount = 0;
}
}
```

## Vedlegg 3 Eksempel på bruken av chatgpt til koding

### Spørsmål:

trenger et skript der jeg starter et rele på digital pin 6 og 7 og som leser dht 22 sensorer på pin 4 og 5

### Svar fra ChatGPT:

Her er et Arduino-skript som starter to reléer på digitale pinne 6 og 7, og leser data fra DHT22 sensorer som er koblet til pinne 4 og 5:

```
cpp
#include <DHT.h>

#define DHTPIN_1 4 // Definerer pinne for første DHT22 sensor
#define DHTPIN_2 5 // Definerer pinne for andre DHT22 sensor
#define DHTTYPE DHT22 // Definerer typen av DHT sensor (DHT22)

DHT dht_1(DHTPIN_1, DHTTYPE); // Oppretter et DHT-objekt for første sensor
DHT dht_2(DHTPIN_2, DHTTYPE); // Oppretter et DHT-objekt for andre sensor

#define RELAY_PIN_1 6 // Definerer pinne for første relé
#define RELAY_PIN_2 7 // Definerer pinne for andre relé

void setup() {
  pinMode(RELAY_PIN_1, OUTPUT); // Setter pinne for første relé som utgang
  pinMode(RELAY_PIN_2, OUTPUT); // Setter pinne for andre relé som utgang
  dht_1.begin(); // Starter første DHT22 sensor
  dht_2.begin(); // Starter andre DHT22 sensor
}

void loop() {
  // Leser fuktighet og temperatur fra første sensor
```



```

float h_1 = dht_1.readHumidity();
float t_1 = dht_1.readTemperature();

// Leser fuktighet og temperatur fra andre sensor
float h_2 = dht_2.readHumidity();
float t_2 = dht_2.readTemperature();

// Starter reléene basert på temperatur fra sensor 1
if (t_1 > 25.0) {
    digitalWrite(RELAY_PIN_1, HIGH); // Starter relé 1 hvis temperatur er over 25°C
} else {
    digitalWrite(RELAY_PIN_1, LOW); // Slår av relé 1 hvis temperatur er under 25°C
}

// Starter reléene basert på temperatur fra sensor 2
if (t_2 > 25.0) {
    digitalWrite(RELAY_PIN_2, HIGH); // Starter relé 2 hvis temperatur er over 25°C
} else {
    digitalWrite(RELAY_PIN_2, LOW); // Slår av relé 2 hvis temperatur er under 25°C
}

// Venter i 2 sekunder før neste avlesning
delay(2000);
}

```

I dette skriptet brukes DHT biblioteket for å lese data fra DHT22 sensorer som er koblet til pinne 4 og 5. Reléene er koblet til pinne 6 og 7. Skriptet leser fuktighet og temperatur fra begge sensorene og starter reléene basert på temperaturavlesningene fra hver sensor. Hvis temperaturen fra en sensor overstiger 25°C, vil det tilhørende reléet bli slått på, ellers vil det være av.