ICU

inclusive care unit

ICU – TEKNISK RAPPORT

Gjermund Persson Myrvang / gjermupm

Herman Sætre Gunnerød / hermasg

Miski Mahamud Ali / miskima

Ola Nordrum Isachsen / olanis

Vlera Kelmendi / vlerak

Innholdsfortegnelse

1. Introduksjon

Rapporten tar for seg den tekniske løsningen og skal gi innsikt i hvordan løsningen er implementert og hvordan den fungerer. I tillegg skal den gi dybde og forklaring til videoen og vise samspillet mellom de ulike komponentene. Rapporten vil også belyse utfordringer vi har møtt på underveis i prosessen og hvordan vi håndterte dem.

1.1 Om gruppen

Gruppen består av Gjermund Persson Myrvang, Herman Sætre Gunnerød, Miski Mahamud Ali, Ola Nordrum Isachsen, Vlera Kelmendi. Vi er alle førstegangsstudenter ved Instituttet for Informatikk ved UiO. Ingen av oss har tidligere teknisk og informatisk erfaring. Derfor er dette vårt første prosjekt hvor vi lager en teknisk løsning. Åpenbart har dette gjort det vanskeligere å lage en veldig avansert løsning. Allikevel har vi fra starten tenkt stort og sagt til oss selv at vi ikke skal la oss begrense av våre tekniske egenskaper.

2. Mål for prosjektet

Målet vårt for dette prosjektet er å bruke sensorbasert teknologi til å skape en artefakt sammen med brukere fra målgruppen. For å nå målet vårt skal vi gjennom hele prosjektet bruke filosofien om DMB, med spesielt fokus på de tre prinsippene medbestemmelse, samskapning og gjensidig læring.

2.1 Prototypens involvering

Prototypen er selve resultatet av prosjektet og er involvert til å svare på et behov til målgruppen. Behovet vi endte opp med var «forståelse». Vi har laget en prototype hvor konseptet er å ikke lage lyd og prototypen er tiltenkt å brukes i ulike spill-sammenhenger. Bakgrunnen for det er at spillet skal brukes som et verktøy for å skape nettopp forståelse og lærdom. Ved å bruke en lydsensor til å sette en støygrense vil det tvinge deltakerne i spillet til å gå ut fra komfortsonen og stemmebruk, til å fokusere mer på kroppsspråk og tegnspråk. Denne grensen kan ikke vår hørsel sanse, derfor blir sensoren en slags ‘dommer’. Dersom dommeren dømmer at noen har laget for høy lyd vil runden være tapt. Dette bidrar til et konkurranse-element noe vi håper kan være en positiv sideeffekt.

3. Presentasjon av video

Videoen består av tre deler. Første del viser et bruksscenario hvor to personer som er døve bruker prototypen sammen med to andre hørende. Gruppen deler seg inn slik at en døv og hørende er på lag. Klippet prøver å illustrere hvordan prototypen kan brukes.

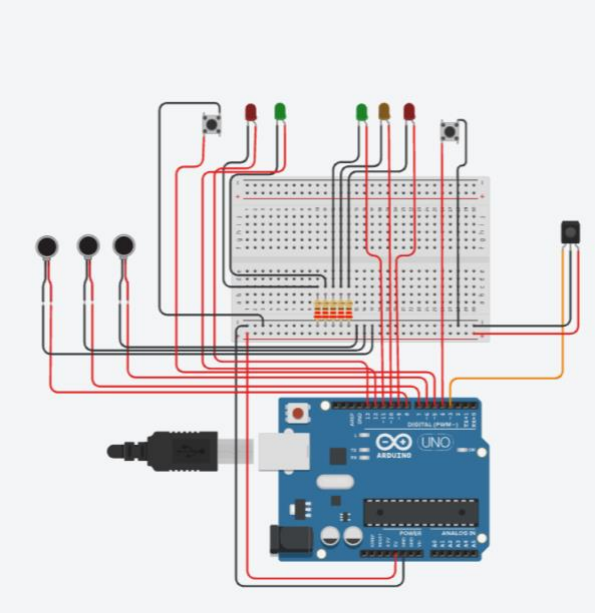
Man kan se at det første laget ikke lager noen lyd og dermed har klart runden. Etterpå er andre laget sin tur og vi ser at den hørende ikke klarer å unngå bruk av stemmen, noe som fører til at de taper runden.

Andre og tredje del av videoen går under det vi kaller «det tekniske hjørnet». Her er hensikten å forklare hvordan det teknologiske fungerer, og hvordan komponentene er satt sammen i brukergrensesnittet. Den første delen av tekniske hjørnet er et kort klipp som viser prototypens ulike funksjoner og skal illustrere at vi har en teknologisk løsning som fungerer. Siste delen består av mer detaljerte forklaringer og illustreringer av hvordan det tekniske faktisk fungerer.

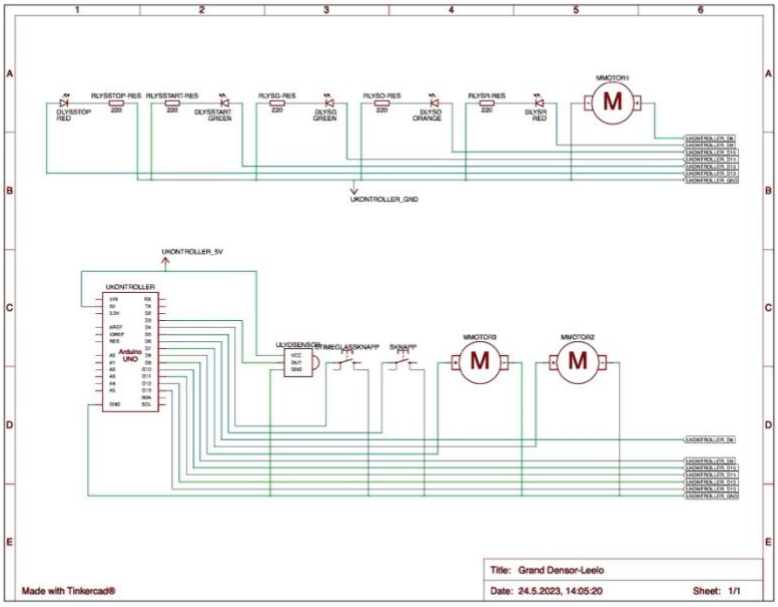
3.1 Link til video

https://vimeo.com/831199694?share=copy

4. Teknisk spesifikasjon



1 Kretstegning - Laget på https://www.tinkercad.com/ NB! I den implementerte løsningen får Arduino strøm fra 9V batteri



2 Kretsdiagram

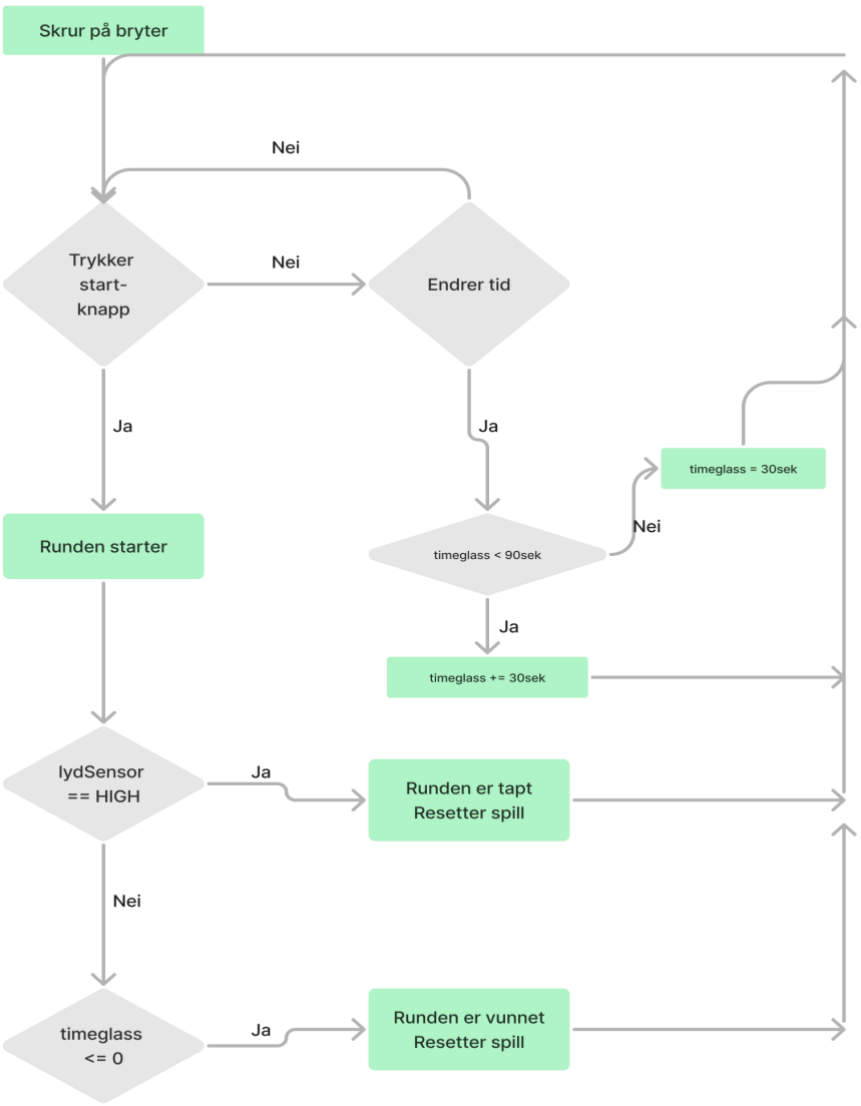
I den tekniske løsningen har fokuset vært på at det skal være oversiktlig og fleksibelt, slik at eventuelle endringer skal være enkelt å forholde seg til. Løsningen er satt sammen av lysdioder, knapper, lydsensor og vibrasjonsmotorer. For å beskytte lysdiodene mot overbelastning bruker vi 220ohm resistorer. De vil sørge for å gi strømmen nok motstand slik at diodene holder lenger. Knappene er initialisert som INPUT_PULLUP slik at man veldig enkelt kan sjekke når knappen er trykket ned og gjøre handlinger basert på det. Lydsensoren har en threshold man kontrollerer på utsiden av sensoren.

Threshold-grensen vi bruker i prototypen har vi testet og leket oss frem til gjennom mye prøving og feiling. Til slutt har vi vibrasjonsmotorene. De er veldig enkle å forholde seg til når de vibrerer så lenge man sender strøm gjennom dem.

Helt overordnet baserer komponentene seg i hovedsak på tre elementer: tid (millis), lydsensoren (threshold) og de to knappene. De fleste komponentene er derfor i hovedsak passive med mindre tidsvariabelen når spesifikke faser, thresholden gir utslag på støy eller at en av knappene blir trykket ned. Mer spesifikt vil komme frem gjennom koden.

4.1 Aktivitetsdiagram

Funksjonaliteten til Hysj kan forklares ved følgende kodeflyt:



3 Aktivitetsdigram som forklarer kodeflyten

4.2 Materialer

4 Ytre skallet til prototypen, laget av treverk med spillets navn i treklosser

5 MDF plate: her måler vi opp og skjærer ut et kvadrat som skal passe i boksen

Prototypen er i hovedsak laget av treverk. Valget av materiale begrunnes med brukerne sitt ønske om at løsningen skal være robust og se naturlig ut. Selve brukergrensesnittet er laget av en kvadratisk MDF plate. Med brukergrensesnittet menes altså der hvor komponentene brukerne interagerer med er festet.

Vi er fornøyde med materialene og syntes de gir en «gjennomført look». Når boksen er lukket ser den nøytral ut med unntak av en bryter som sitter på baksiden. Det svarer godt på brukerens ønske om et materiale som er robust, men som glir inn i omgivelsene. Treklossene som indikerer navnet gir i tillegg boksen et mer spill-relatert utseende som vi var ute etter.

4.2.1 Utfordringer med materialer

Vi har valgt relativt enkle materialer og de har ikke vært problematiske å jobbe med. Det eneste vi kan påpeke er at MDF platen var kanskje litt myk, så da vi sagde den ut ble den litt flisete og ujevn. Dette fikk vi derimot ordnet med litt sandpapir.

Man kunne påpekt at trevirke er et dårlig materiale-valg med tanke på hvis det skulle oppstått høy varme fra de tekniske komponentene. Imidlertid vil vi påstå at løsningen vår er såpass simpel og det oppstår ikke nok varmeproduksjon til at det vil bli et problem.

4.3 Komponenter

Utsyr	Ant	Beskrivelse	Hvorfor
Utstyr fra Arduino startpakke:			
Arduino Uno	1	Microkontroller	Brukes til å kontrollere hvordan komponentene kommuniserer med hverandre.
Breadboard	1	Brett for å lage kretser	Lage kretser, koble sammen ulike komponenter
Knapp	2	Brytere som lukker en krets når trykket. Detekterer av/på signaler.	Registrerer trykk og endrer en variabel slik at spillet kan være i gang og ikke igang eller endre timeglass
Rød lysdiode	2	Type diode som lyser når strøm passerer gjennom	Symbolisere når programmet ikke er igang eller at det er 30 sekunder igjen
Grønn lysdiode	2	Type diode som lyser når strøm passer gjennom	Symbolisere når programmet er i gang eller at det er > 60 < 90 sekunder igjen
Gul lysdiode	1	Type diode som lyser når strøm passer gjennom	Symboliserer > 30 < 60 sekunder igjen
220ohm resistor	4	Motstand strømmen går gjennom i en krets	Ledpærene tåler ikke 5V strøm, derfor brukes resistor for å gi strømmen motstand i kretsen. 220ohm er brukervennlig mot lyspærene.
9V Batteri	1	Strømkilde til Arduino	Plugges inn i «VIN-pin» pin på arduino, gir strøm til kontrolleren
Ekstra utstyr			
Sound Sensor	1	Lydsensor som er en slags mikrofon som settes 'HIGH' dersom den fanger en lyd som overstiger innebygget 'threshold'. Denne kan justeres	Initialisere lys og vibrasjon når lyd har overskrivet threshold. Indikerer at runden er tapt
Vibration Motor	3	Vibrasjonsmotor som vil vibrere dersom den settes 'HIGH'	Output, vibrerer og symboliserer at runden er tapt
Female - to - male wire	20	Tillater enklere koblinger mellom ledninger	Blir mer oversiktlig og robuste koblinger
Strømbryter	1	2-polet strømbryter	Gir mulighet til å skru av/på hele kretsen. Man slipper å fjerne batteriet

4.4 Koden

Koden strekker seg over 300 linjer ettersom vi er opptatt av at det skal være ryddig og oversiktlig. Koden inneholder kommentarer som skal vise hva som skjer og hvorfor. Ytterliggere forklaringer vi mener er relevante å belyse utdypes i avsnittet om utfordringer samt løsninger.

Link til kode:

https://github.com/olaisa123/ICU

4.4.1 Forklaring av koden

Koden er programmert med en hendelsesbasert tankegang. Loopen går kontinuerlig og «venter» på at relevante hendelser skal skje. Disse hendelsene kan være at sensorverdien blir oppdatert og overstiger threshold-grensen, tidsvariabelen oppdateres og er <= 0 eller at en av knappene er registrert som trykket ned. Hver av disse hendelsene vil igjen sørge for at utvalgte kodeblokker kjøres. Eksempelvis vil kodeblokken som kaller metoden *rundeFullfort()* og resetter spillet skje dersom tidsvariabelen når 0.

4.4.2 Utfordringer ved koden + løsninger

Selv om den tekniske løsningen vår ikke består av spesielt mange eller kompliserte komponenter, har vi måttet jobbe endel med koden. Til å begynne med var koden uoversiktlig og derfor vanskelig å endre. Vi bestemte oss for å flytte mye av funksjonaliteten inn i metoder, utenfor loopen. På denne måten ble loopen lettere å lese.

Ettersom koden kun kjører en tråd har vi vært nødt til å unngå bruk av delay(). Til å begynne med benyttet vi delay() i deler av koden der sensoren skulle være aktiv. For å unngå dette måtte vi legge på endel linjer, blant annet i oppdaterTimeglass(), men dette var nødvendig for bedre funksjonalitet.

```
// ----------------- SPILL I GANG ------------------
if (erIgang) {
    //Lagrer tid gått fra forrige måling i diff
    //Oppdaterer differanse fra sist måling
    //Teller ned timeglass ved å trekke fra diff
    diff = millis() - startTidspunkt;
    startTidspunkt += diff;
    timeglass -= diff;
```

6 Kodeblokken som kjøres når spillet er igang

Når det kommer til oppdatering av timeglass variabelen måtte vi også unngå delay(). Her valgte vi å lage to variabler: diff og startTidspunkt.

- Starttidspunkt holder tidspunktet runden startet.
- diff holder differansen / tiden det har gått siden sist loop.

For hver loop vil timeglass telle ned. Dette gjøres ved å trekke fra tiden det har gått siden sist loop. (timeglass - = diff)

Differansen finner vi ved å ta millis() – startTidspunkt.

Til å begynne med opplevde vi at lydsensorer slo ut med engang når vi trykket start, fordi den oppfattet lyden av knappetrykket. I tillegg ble det innimellom en tilsynelatende uendelig loop mellom lydsensoren og vibrasjonsmotorene. Lydsensoren oppfattet en lyd og vibrasjonsmotorene startet å vibrere, for videre at lydsensoren oppfattet vibrasjonen. For å løse dette problemet lagde vi variabelen lydLaget. Dersom en lyd blir oppfattet av sensoren lagres millis() i lydLaget, og en kodeblokk sørger for at sensoren ignorer alle lyder de neste 3 sekundene. Dersom brukeren trykke på start/stop knappen vil også millis() bli lagret i lydLaget, på den måten ignoreres lyder som kommer av knappetrykk etc i starten av en runde.

```
//------------- START/STOP ---------------
if (startStoppKnappStatus == LOW) {
    startTidspunkt = millis(); //Lagrer startTidspunkt
    lydLaget = millis();       //LydLaget settes til millis
    delay(50); //Debunce

    erIgang = !erIgang;       //oppdaterer boolsk verdi

        if(!erIgang){       //Dersom runden avsluttes resettes timeglass
        timeglass = 30000;
        oppdaterTimeglass();
      }
    }
```

7 Kodeblokken som kjøres når en start/stopp knappen trykkes

Lydsensor:

```
if (millis() - lydLaget > 3000) {
  if (lydSensorVerdi == HIGH && erIgang) {
    lydLaget = millis(); //Lagrer tiden
```

8 Sørger for at det har gått 3sekunder siden forrige gang sensoren ga utslag på en lyd eller start/stopp knappen ble trykket . Vil også kun kjøre dersom spillet er i gang. Oppdater således variabelen lydLaget med millis()

Metoden oppdaterTimeglass() sørger for at de tre lysdiodene som indikerer hvor lenge det er igjen av runden oppfører seg i henhold til tiden. I denne metoden er det tre kodeblokker som har hensikt å skape en blinkende effekt. Effekten er ment til å gi brukerne indikasjon på at et tidsintervall straks er ferdig. Eksempel på dette er hvis runden startet på 90 sekunder vil det rød lyse blinke de siste sekundene før tiden er 60 sekunder. Vi har en *constant int blinkInterval* som er initialisert til 500. Variabelen *unsigned long forrigeBlink* lagrer kontinuerlig millis()ved blink i den relevante tidsintervallet slik at vi kan forsikre oss at det går minumum blinkInterval(500ms) mellom hvert blink. Begrunnelsen for denne måten er som tidligere nevnt at vi unngår å bruke delay().

```
void oppdaterTimeglass() {

  // 0 < sekunder igjen < 10 (RØD Blinking)
  //Unngår bruk av delay for å holde LydSensor aktiv
  if (timeglass <= 10000 && timeglass > 0){
    if(millis() - forrigeBlink >= blinkInterval){

      forrigeBlink = millis();

      //Bytter om på LOW/HIGH
      if(blinkStatus == LOW){
        blinkStatus = HIGH;

      } else{
        blinkStatus = LOW;
      }

      digitalWrite(lysR, blinkStatus);
    }
  }
```

9 Kodeblokken som illustrer ett av tre tidsintervall hvor det skal oppstå en blinkeeffekt. Dette tilfelle de ti siste sekundene

5. Oppsummering/konklusjon

Det helt sentrale med prototypen er lydsensoren. Oppgaven for prosjektet var å bruke «sensorer, til å sanse det vi mennesker ikke kan sanse: det vi ikke kan se eller høre, eller fenomener som er for raske eller langsomme til at vi får det med oss.». Måten vi har praktisert dette i vårt prosjekt er nettopp med lydsensoren. Lydsensorens threshold setter en grense som vi har brukt i vår prototype som en dommer. Denne grensen gir prototypen dette spill- og utfordringsmomentet vi har vært ute etter. For å konkludere så fungerer løsningen og har funksjonaliteten vi var ute etter. Imidlertid vil vi bemerke at prototypen fortsatt kun er en prototype. Lydsensoren er ikke veldig avansert, og det er veldig vanskelig å finne en grense som er fleksibel og passende til ulike scenarioer.

5.1 Veien videre

Dersom man skulle utviklet denne ideen videre er det mange ting man kunne forbedret. Fokuset vårt ville vært og implementert en bedre/smartere lydsensor som har funksjonaliteten til å måle desibel og skille mellom ulike lyder (analoge signaler). Videre ville vi skaffet bedre vibrasjonsmotorer, slik at vibrasjonene forplanter seg bedre i bordet og virkelig får boksen til å vibrere. Andre relevante oppgraderinger ville vært å ha et oppladbart batteri, bedre tilpassede ledninger, RGB led og et LCD timeglass. Generelt sett ville vi sørget for å bruke mer robuste komponenter og materialer.