Datorbaserat mätsystem för bedömning av kroppsuppfattning hos patienter med ätstörningar

Paul Dadzie
Firas Majid

2016

LUNDS UNIVERSITET
Lunds Tekniska Högskola

Examensarbete i Elektrisk Mätteknik

Institutionen för Biomedicinsk Teknik
Handledare: Mikael Evander (LTH), Gunhild Kjölstad(BUP)
Examinator: Johan Nilsson (LTH)

Innehåll

Disposition

1. **Introduktion**
   Detta är avsett för att ge läsaren en insikt i vad denna rapport handlar om. Här specificeras också syfte och mål.

2. **Teori**
   Genomgång av den grundläggande teori som krävs för att förstå innehållet i rapporten.

3. **Hårdvara**
   I detta avsnitt berörs hur prototypen är konstruerad och vilka komponenter som valts.

4. **Mjukvara**
   Här förklaras val av design/struktur som programmet är uppbyggd av. Här visas också den visuella strukturen för hur programmet fungerar.

5. **Resultat**
   Det som kan mätas gällande hur bra målen är uppfyllda visas i detta avsnitt.

6. **Diskussion**
   Här diskuteras resultaten och hur de ska tolkas. Avsnittet innehåller även tankar och reflektioner angående fortsatt arbete.

7. **Referenser**
   Innehåller de referenser varifrån information har använts. Stilen IEEE används för hänvisning och upplägg av dessa.

1 Introduktion

1.1 Bakgrund

1.1.1 Vad är en ätstörning?

Till skillnad från tidigare generationer har vi idag god tillgång till mat vilket gör att föräldrar inte behöver låta sina barn gå hungriga. Tyvärr finns det ändå föräldrar som får se sina barn svälta, inte pga av matbrist, utan pga av en sjukdom, anorexia nervosa (”nervös aptitförlust”). Denna svält beror på att de inte har en aptit och innebär att den drabbade bemästrat sin hunger och äter inte fastän han/hon är hungrig.

Denna mystiska sjukdom, som också är den mest allvarliga formen av ätstörning, drabbar främst flickor och unga kvinnor[1].

1.1.2 Bedömning och klinisk utredning av en ätstörning

Det finns fyra punkter att ta hänsyn till vid en utredning och bedömning av en ätstörning hos ett barn:

1. Förändrad kroppsfunktion pga av ätstörningen
2. Familjefunktion
3. Individuell psykopatologi
4. Kroppsuppfattningsstörning

Den sista aspekten, kroppsuppfattningsstörning, är den som ligger som grund till detta arbete och kommer därför att förklaras närmre.

En överskattning av kroppsstorlek tillsammans med ett missnöje med kroppen är mycket vanligt vid anorexia nervosa, speciellt överskattas mage och lår. För att bestämma hur patienten ska behandlas görs olika tester och bedömningar. Bedömning av perceptuell uppfattning av kroppsstorlek (EGON[1]) och ett kroppsattitydtest

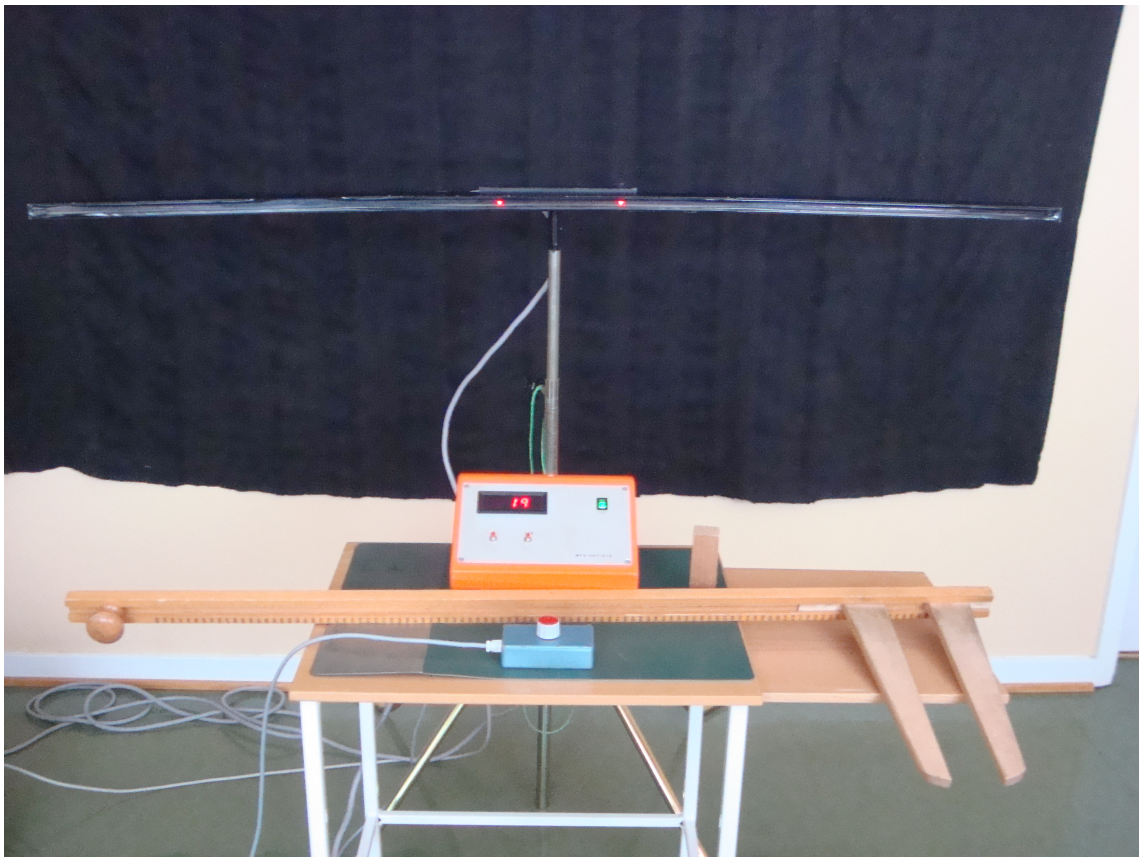
[1] Den första patienten hette Egon vilket systemet namngavs efter.

(BAT[2]) används för att få en uppfattning av graden av störning och för att avgöra vilka patienter som ska erbjudas kroppskännedomsbehandling av sjukgymnast. Detta arbete kommer att fokuseras på att förbättra bedömningen av den perceptuella uppfattningen av kroppsstorlek, EGON, som förklaras närmre i nästa avsnitt.

[2] Body Awareness Teherapy

1.2 EGON

1.2.1 Hur ser EGON ut idag och hur används den i en kroppsuppfattningsbedömning?



EGON med lysdiodrampen, kontrollpanelen, mätsticka & handkontroll

EGON idag är ett analogt system som tillverkades som ett examensarbete på LTH 1985. Systemet består av en kontrollenhet som sjukgymnasten använder, en handenhet för patienten samt en lysdiodsramp som är 159 cm. Handenheten består av en ratt som styr vilka två lysdioder som lyser på rampen och på kontrollenheten kan man, förutom att utläsa avståndet mellan de upplysta lysdioderna, nollställa rampen så att inga lysdioder lyser.

Bedömningen[2][4], som går ut på att jämföra patientens verkliga kroppsdimensioner med sina uppfattade kroppsdimensioner, går till enligt följande:

1. Det första måttet som tas är ett neutralt objekt (t.ex en kloss) som används som ett referensobjekt för normalisera resultatet[3].
2. Sjukgymnasten tar ett mått på objektet med mätstickan och visar det för patienten (se bild nedan).
3. Patienten ställer in måttet på lysdiodrampen mha. handenheten. Detta görs två gånger, det första har lysdioderna i miten som utgångsläge medan det andra har ytterkanterna[4].
4. Alla värden samt det uppmätta värdet (mha. mätstickan) förs in i ett protokoll.

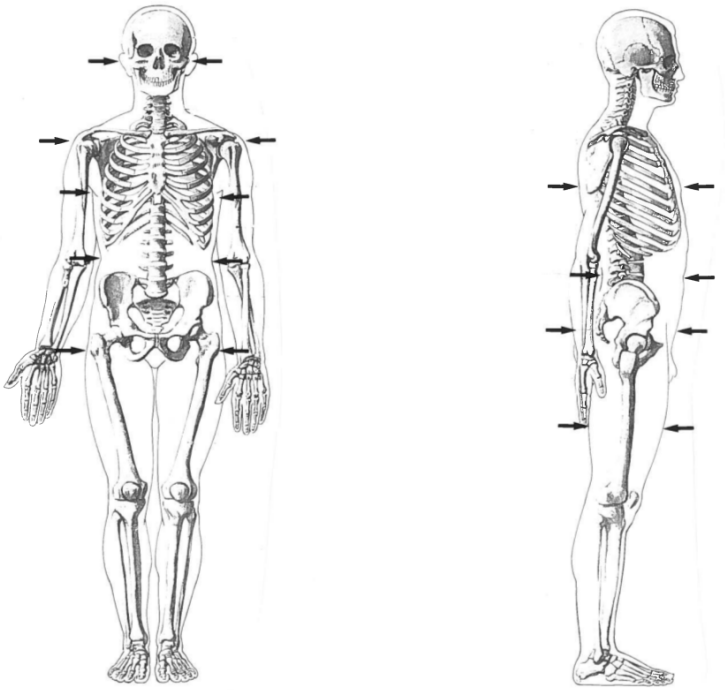
Mätsticka används för att illustrera kroppsmåttet som ska uppskattas för patienten

[3] Dvs. ta bort eventuella felkällor som som inte har med patientens kroppsuppfattning

[4] Man ställer alltså in måttet från noll på första försöket medan man på andra försöket utgår från det maximala måttet och minskar avståndet mellan de upplysta lysdioderna. Genomsnittet av dessa två skattningar räknas som skattningsvärdet.

Efter referensobjektet görs detta på de olika kroppsmåtten vars resultat också förs in i protokollet. Med hjälp av dessa kan sedan ett BPI[5] beräknas.

$$BPI = \frac{Skattning1 + Skattning2}{2} \mathbf{X} \frac{100}{Objektets\ Mått}$$



De olika kroppsmåtten som ska uppskattas. Bild från *Atlas Anatomii Czlowieka, 1975.*

[5] Body Perception Index. Ger en bild över hur mycket patientens kroppsuppfattning skiljer sig från verkligheten.

1.2.2 Tidigare förbättringsprojekt för EGON

Ett LabVIEW-baserat mätsystem [5] utvecklades som ett projektarbete i kursen Datorbaserade Mätsystem. Detta systemet ersatte den stora analoga kontrollpanelen med en dator, lysdiodrampen med en projektorbild och handenheten med en digital version (potentiometer med Arduino kopplat via USB till datorn).

När man startar en mätning går systemet igenom alla kroppsmått som ska mätas enligt protokollet vilka också fördes in i protokollet automatiskt. När man avslutat det sista måttet beräknas alla värden och man kan sedan exportera en rapport i xml.

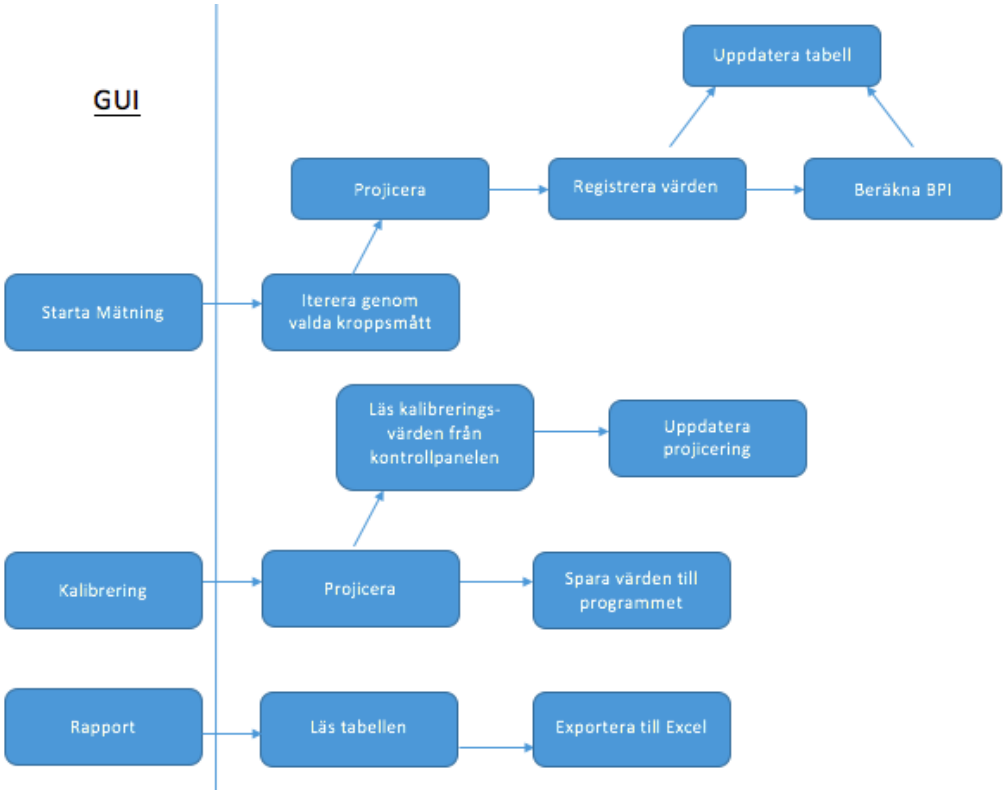
De största bristerna med systemet var främst användningen av en potentiometer samt begränsningarna i flexibiliteten. Användningen av en potentiometern medför att man, inför varje ny mätning, manuellt måste nollställa den genom att vrida tillbaka den. Dessutom var systemet uppbyggd så att alla mätningar var tvungen att köras efter varandra utan avbrott. Detta betyder att en felakting mätning inte kan köras om individuellt, utan samtiga måste göras om. Det var också svårt att få en bild över hur bra mätningarna verkar vara då alla resultat synliggjordes först när man var färdig.

Med detta sagt får projektet anses som lyckat då det visade, väldigt tydligt, hur ett digitaliserat system dramatiskt kunde öka kvaliteten av bedömningarna som används i den kliniska utredningen.

1.3 Syfte

Detta examensarbete kommer att undersöka och utveckla det digitaliserade systemet så att det kan ersätta det gamla analoga EGON. Förutom att digitaliseras och ha ökad precision så behöver systemet ha flexibilitet samt ha automatiska beräkningar (All information och alla värden kan exporteras till en rapport i Microsoft Excel. Detta kommer att ge försöksledaren möjligheten att lägga mer fokus på patienten.

1.4 Översiktlig lösning, CBIT



Översikt på systemets funktionalitet

Systemet, CBIT (Computer Based Image Treatment), som ska utvecklas kommer att bestå av två stycken handenheter (en till patienten och en till försöksledaren) där den ena kommer att användas för att flytta på markeringarna som används för skattningarna och den andra sparar skattningen och går vidare till nästa mätning. Dessa skall kommunicera med en dator som hanterar användargränssnitt, datainsamling,beräkningar och presentation(via extern skärm/projektor) samt loggning av mätningar. Den ena handenheten (försöksledarens) kommer att vara trådlös för att möjliggöra fri rörelse i rummet[6].

[6] Patienten står på en markerad punkt på golvet och behöver därmed inte en trådlös enhet. Detta försöktes att implementeras men resulterade i sämre noggrannhet i systemet, läs mer om detta i avsnitt 3.1 och 6.1.

Handenheterna kommer att baseras på arduinokort för deras enkelhet, låga pris och storlek. Istället för en potentiometer, som användes i förbättringsprojektet för EGON (se avsnitt 1.2.2), så kommer en pulsgivare (rotary encoders) att användas, dels för att smidigare nollställa den mellan mätningar[7] och dels för dess höga precision. Kommunikationen kommer att ske via USB och Bluetooth, och programmet kommer att skrivas i LabVIEW. Användargränssnittet och kopplingen mellan datainsamlingen och programmet kommer att hanteras på ett flexibelt sätt så det är möjligt att både köra ett förinställt mätprogram samt ta enskilda mätvärden.

[7] En potentiometer måste fysiskt nollställas genom att ställa in ratten igen medan en pulsgivare är oändlig och kan nollställas genom mjukvaran

2 Teori

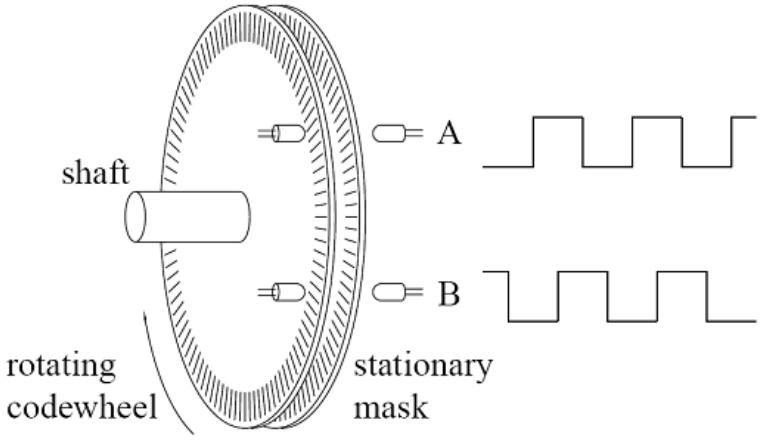
2.1 Optisk pulsgivare

En pulsgivare[8], är ett instrument som med hög precision läser av positionen hos en axel. Antalet punkter hos pulsgivaren anger dess upplösning/precision. En pulsgivare med 200 punkter kan ange axelns position med en precision med 360°/200, vilket motsvarar 1.8 grader.

Till skillnad från en potentiometer kan pulsgivaren vridas oändligt många varv.

2.1.1 Räkna pulser

Den optiska pulsgivaren använder sig av en roterande skiva med en lysdiod som ljuskälla på ena sidan och fotodiod på andra sidan som avger en puls för varje ”hål/spalt” som passerar.

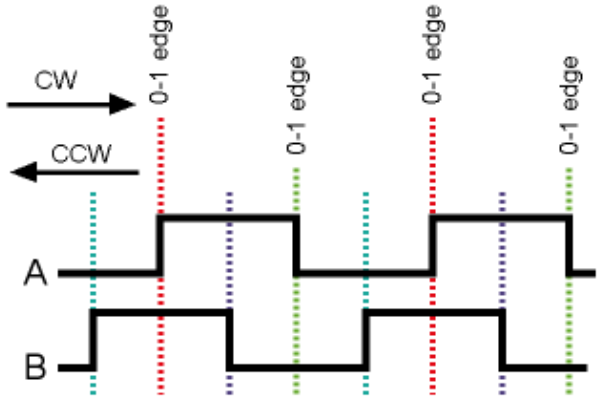


En pulsgivare med två uppsättningar lysdioder och fotodioder, Från [6].

[8] I detta projektet används en optisk pulsgivare. Det finns dock pulsgivare där andra metoder används, t.ex magnetiska, kapacitiva, mekaniska m.fl

2.1.2 Avgöra riktning

Genom att använda två uppsättningar lysdioder och fotodioder (med en förskjutning så att fotodioderna inte detekterar spalterna samtidigt) kan man avgöra i vilken riktning pulsgivaren rattas i. Kanalerna är förskjutna 90° vilket gör det möjligt att avgöra riktning. När en positiv flank (när signalen går från låg till hög) upptäcks på kanal A kollas kanal B, är den hög så rör vi oss i medurs (”CW”) och är den låg så rör vi oss moturs (”CCW”). Figuren och tabellen nedan visar hur riktningen avgörs beroende på nuvarande och föregående värden på kanalerna.



Pulser från två kanaler, Från [8]

.

A-förra värdet	A-nytt värde	B-förra värde	B-nytt värde	Riktning
0	0	0	0	Stilla
1	1	1	1	Stilla
0	1	1	1	Medurs
1	0	0	0	Medurs
0	0	0	1	Medurs
1	1	1	0	Medurs
0	0	1	0	Moturs
1	1	0	1	Moturs
0	1	0	0	Moturs
1	0	1	1	Moturs

Tolkning av riktning med hjälp värdena på kanal A & B

2.2 LabVIEW

LabVIEW är en utvecklingsmiljö för ett grafiskt programmeringsspråk som kallas Göch är utvecklat av National Instruments. Detta skiljer sig från ”vanliga” programmeringsspråk på främst två sätt: ikoner representerar funktioner istället för rader av kod och flödet av data är det som avgör hur programmet körs istället för instruktioner.

Programmen som skrivs i LabVIEW kallas för Virtuella Instrument (s.k. VIs) och består av en frontpanel och ett blockdiagram. Frontpanelen fungerar som ett användargränssnitt där block såsom knappar, numeriska indikatorer, kontroller osv kan byggas in. I blockdiagrammet läggs det till kod med hjälp av grafiska representationer av funktioner för att styra frontpanelobjekten.

2.2.1 Seriell kommunikation vs. LIFA/LINX

LIFA[9] och LINX är uppsättningar verktyg för LabVIEW som gör det möjligt att interagera med Arduino över t.ex USB. Dessa gör det möjligt för LabVIEW att läsa och skriva till Arduinon samt möjliggör användning av dess ingångar och utgångar.

Pga. att LIFA (som ursprungligen skulle användas) inte stöds längre låg valet mellan efterträdaren LINX och vanlig seriell kommunikation genom VISA[10](en standard för konfigurering och programmering av instrumentering innefattande bl.a seriell kommunikation och USB).

Då den vanliga seriella kommunikationen var mycket enklare att använda (LINX behöver externa bibliotek vilket komplicerar proceduren) så valdes den.

[9] LIFA: LabVIEW Interface For Arduino
[10] VISA: Virtual Instrument Software Architecture

2.2.2 State Machine vs. Master-Slave vs. Producer-Consumer

State Machine är en designstruktur som används i system där det finns tydliga tillstånd, varje tillstånd kan leda till andra tillstånd och kan avsluta processflödet. Då vår mjukvara, vid vissa tillfällen, behöver ha parallella processer så blir en State Machine ineffektiv och oanvändbar.

Master-Slave är en struktur där multipla loopar kan köras samtidigt och i olika hastigheter. En loop agerar som master och kontrollerar alla slav-loopar.

Producer-Consumer är baserad på Master-Slave strukturen där indelningen består av de som producerar data (producenter) och de som konsumerar data (konsumenter). Producenterna tar in data och avgör vilken konsument som ska exekveras och i vilken ordning. Databuffring används för kommunikation mellan looparna och är det som skiljer denna struktur från Master-Slave. Denna struktur används därför med fördel i applikationer där man tar in mycket data och behöver bearbeta den i ordning.

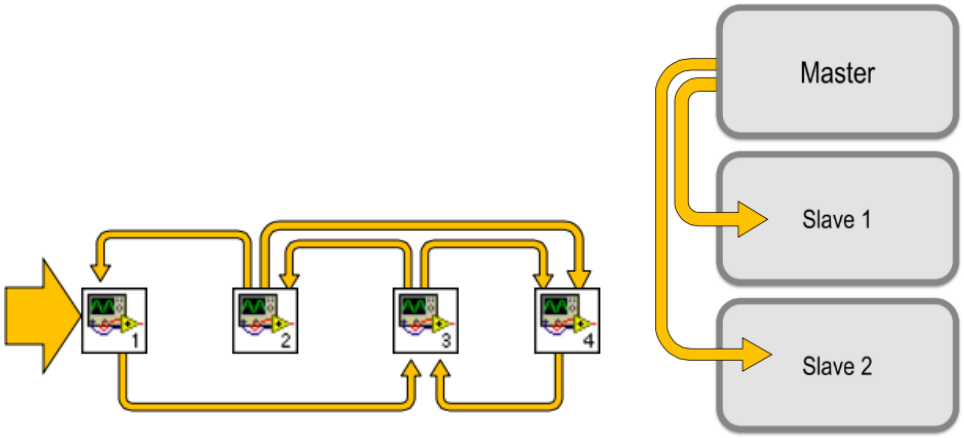
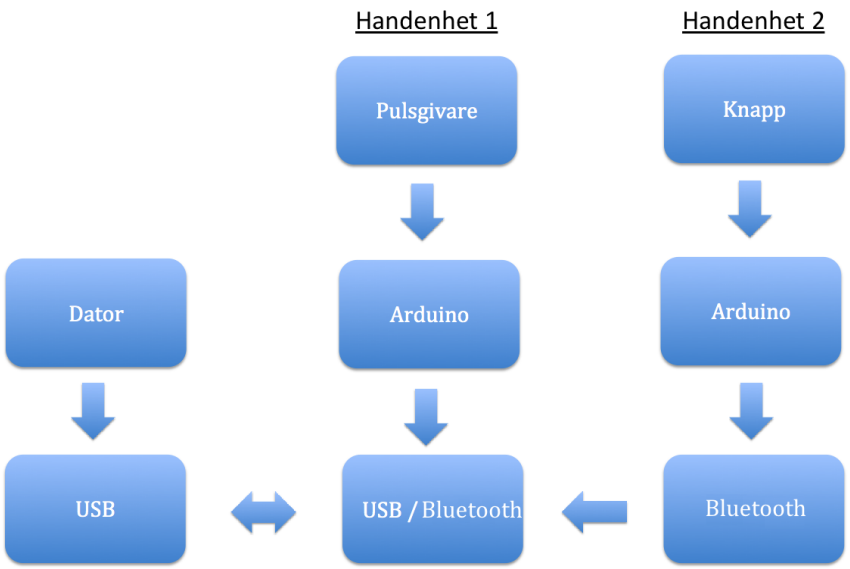


Illustration av State-Machine (till vänster) och Master-Slave (till höger)

3 Hårdvara

3.1 Översikt



Översiktsbild över hårdvaran i prototyperna

Hårdvaran består av två handenheter, där kommunikationen mellan datorn och handenheten med pulsgivaren (som används av patienten) är kontinuerlig så att mätpunkterna på skärmen/projektorbilden flyttar sig medan man vrider på pulsgivaren. Handenheten med knappen (som används av försöksledaren) används för att spara skattningen och gå vidare till nästa mätning, denna har enbart kommunikation med den första handenheten. Då försöksledaren behöver kunna röra sig fritt (patienten står alltid på en markerad punkt på golvet) i rummet så är dennes handenhet trådlös med hjälp av Bluetooth. Dvs kommunikationen mellan handenheterna går via Bluetooth medan kommunikationen mellan dator och handenheten med pulsgivaren (patientens) går via USB.

Kretschema för båda handenheterna hittas i Bilaga A och programmeringskoden för enheterna i Bilaga B.

3.2 Optisk pulsgivare: YUMO COM-10932[11]

Pulsgivaren som används i detta projektet har 200 pulser (360°/200) vilket ger en upplösning på 1.8 grader. Den har dessutom fler än en uppsättning ljusdiod/fotodiod vilket ger oss möjligheten att avgöra riktning, dvs vi kan öka/minska avståndet mellan markeringarna på projektorbilden.

Pulsgivaren drivs på 5-12 VDC vilket gör det möjligt att använda samma spänningskälla som Arduinon.

3.3 Arduino Micro[12][13]

Arduino är en open-source plattform som används för elektronikprojekt. Det består av ett mikrokontroller kort som kan köpas färdigbyggt och använder ett eget programmeringsspråk som har en syntax som påminner om C[11].

Arduino Micro är ett väldigt litet mikrokontrollerkort[12] med en ATmega32u4. Den har 20 digitala ingångar/utgångar med interna pull-up resistorer[13] och en 16MHz kristalloscillator. Den har också en inbyggd USB-kommunikation vilket både eliminerar behovet av en sekundär processor och driver hela kortet.

Två Arduino mikrokontrollerkort kommer att användas i handenheterna för att hålla reda på pulsgivarens riktning, räkna pulser, hålla reda på knapptryckningar och skicka informationen via Bluetooth och USB.

[11] Programmeringsspråket är en som är en implementering av Wiring, ett open-source programmeringsramverk för mikrocontrollers.

[12] 18 x 48 mm

[13] Pull-up resistorer används för att ingången alltid ska vara definierad, dvs. alltid en etta eller nolla.

3.4 Bluetooth-modul: BlueSMIRF Silver[14]

BlueSMIRF Silver är en Bluetooth-modul för att ersätta seriell kommunikation via kablar. Denna versionen använder en modul med lite mindre räckvidd i jämförelse med BlueSMIRF Gold. För detta projektets syfte är räckvidden tillräcklig. Dessutom har den en seriell strömhastighet på 2400 - 115 200bps[14] vilket innebär att det inte kommer att märkas av några fördröjningar.

Modulen drivs på 3.3 - 6V vilket gör det enkelt att dela spänningskälla med både mikrokontrollerkorten och pulsgivaren. Modulens strömförbrukning ligger runt 30mA (Hur detta påverkar batterilivslängden diskuteras i avsnitt 6.2.3).

Bluesmirf Silver

[14] bps: bits per sekund

4 Mjukvara

4.1 Översikt

Mjukvaran innefattar ett program på datorn skrivet i G, programmeringsspråket i LabVIEW, och är designad efter en Producer-Consumer struktur (se avsnitt 2.2.2).

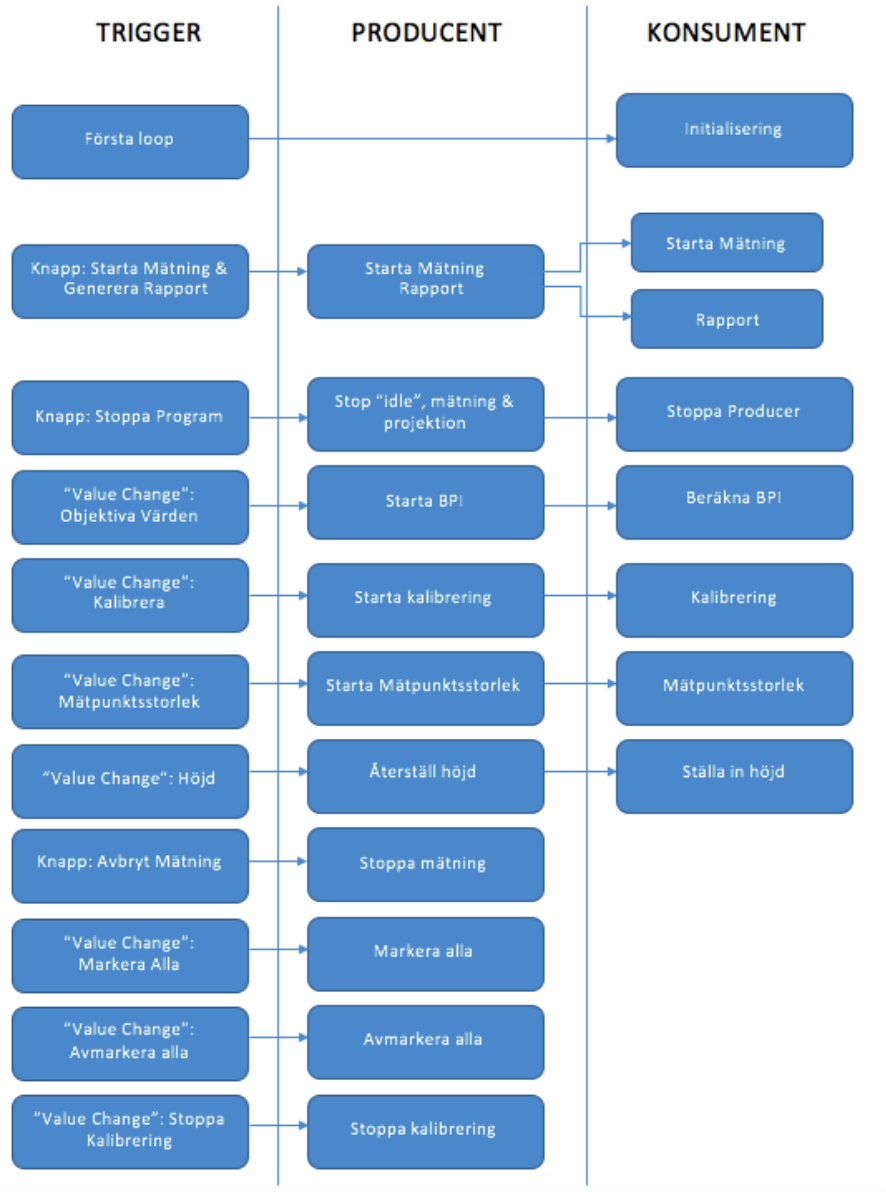
Programmet hanterar allt ifrån kalibrering av mätpunkterna på skärmen till själva skattningarna med värden från pulsgivaren till generering av en rapport. Användargränssnittet (närmare förklaring i användargränsnitt, avsnitt 4.5) består av tre aktiva flikar där olika funktioner och kontroller finns tillgängliga för användaren. Det finns en fjärde flik, men denna används för att tolka vanliga felmeddelanden som kan uppstå, dvs. den är inte aktiv i programmet.

Programmet använder sig av två skärmar (alternativt en skärm och en projektor) där den primära skärmen används för kontrollpanelen, det är här programmet styrs. På den sekundära skärmen syns mätpunkterna som patienterna använder för att skatta de olika mätningsvärdena.

Många av buffringsmöjligheterna som ingår i en Producer-Consumer struktur är aktivt borttagna då många av kontrollerna/knapparna inaktiveras när man är i ett specifikt tillstånd. Detta valdes för att undvika att köa upp events av misstag, t.ex att råka dubbelklicka på Starta Mätning"hade, efter avslutat mätning, startat ännu en.

4.2 Flödesschema - CBIT

I flödesschemat nedan kan alla producent-konsument förhållanden ses. Beskrivningar för alla producenter och konsumenter hittas i avsnitt 4.3 och 4.4.



CBIT - Flödesschema

4.3 Producenter

Producenternas uppgift är att kolla, samla in och skicka vidare data till rätt konsument. Alla producenter i vår mjukvara listas och förklaras nedan:

- **Timeout**
  Timeout är ett standardevent som används när inget annat event används (eller har triggats av användaren). I detta projekt så gör timeout-eventet ingenting aktivt utan väntar på kommandon från användaren.

- **Start Mätning**
  Detta event är huvudfunktionen i programmet och startar en BPI mätning. Den styrs av knappen ”Starta Mätning” och skickar ett kommando genom kön till konsumenten med samma namn.

- **Generera Rapport**
  Report triggas av ”Generera Rapport”-knappen och startar konsumenten ”Rapport”.

- **Starta BPI**
  Detta event triggas av att ett värde ändras i ”Objectiva Värden” - kolumnen. Eventet i sin tur startar konsumenten ”Beräkna BPI”.

- **Stoppa Program**
  Detta event stänger ner alla pågående events utan att de får slutföra sina uppgifter och stänger därefter programmet. Det är främst pågående mätningar och kalibreringsfunktionen som påverkas av detta event.

- **Stoppa Mätning**
  Detta eventet avbryter en pågående mätning på samma sätt som ”Stoppa Program”. Det som skiljer de åt är att ”Stoppa Mätning” bara bryter ”Mätning” - VI:ns iteration.

- **Markera Alla**
  Är ett litet event som möjliggör för användaren att markera alla kroppsdelsmätningar (checkboxarna till vänster om namnen). Kroppsdelsindikatorerna är en samling boolean konstanter.

- **Avmarkera Alla**
  Utför motsatsen till ”Markera Alla” och avmarkerar alla checkboxar.

- **Starta Kalibrering**
  Detta event startar konsumenten ”Kalibrering”.

- **Stoppa Kalibrering**
  Detta event startar konsumenten med samma namn.

- **Starta Mätpunktsstorlek**
  Startar konsumenten Mätpunktsstorlek".

- **Återställ Höjd**
  Detta event triggas av att värdet på patientens höjd i fliken ”Patient Information” ändras. Eventet i sin tur startar konsumenten ”Ställa in höjd”.

4.4 Konsumenter

Konsumenterna, som exekveras med rätt data vid rätt tillfälle[15], listas och förklaras nedan:

- **Starta Mätnin**
  Detta är huvudfunktionen i programmet som triggas av producenten med samma namn. Här projiceras två prickar på projektorn/skärmen, som i sin tur styrs av pulsgivaren. När knappen på den andra handenheten trycks så sparas avståndet mellan prickarna (i centimeter).

- **Rapport**
  Skapar ett exceldokument med patientinformation, skattningsvärden och beräkningar.

- **Initialisering**
  Triggas inte av en producent utan körs en gång när programmet startas. Använder sig av en initieringsfil (.ini) där inställningar som skärmbredd och längd finns sparat sedan tidigare. Detta för att slippa fylla i detta varje gång man startar programmet på samma dator.

- **Stoppa Producer**
  Detta är en enkel konsument som hanterar felkoder för att snabbare stänga programmet om något gått fel programmatiskt.

- **Beräkna BPI**
  Triggas av producenten med samma namn. Varje gång något värde i ”Objectiva Värden” ändras så triggas producenten och därefter konsumenten. Detta innebär att hela matrisen med skattningsvärden och objektiva värden itereras för att räkna ut alla BPI-värden.

[15] Ordning av exekvering och data som skickas in kontrolleras av de producerande looparna

- **Kalibrering**
  Startar manuell kalibrering för maxavståndet (159 cm) mellan staplarna som används i projiceringen. Därefter beräknas rätt skalning i programmet.

- **Mätpunktsstorlek**
  Här väljs storlek på de ställbara prickarna som används för mätningarna i fliken ”Kalibrering”.

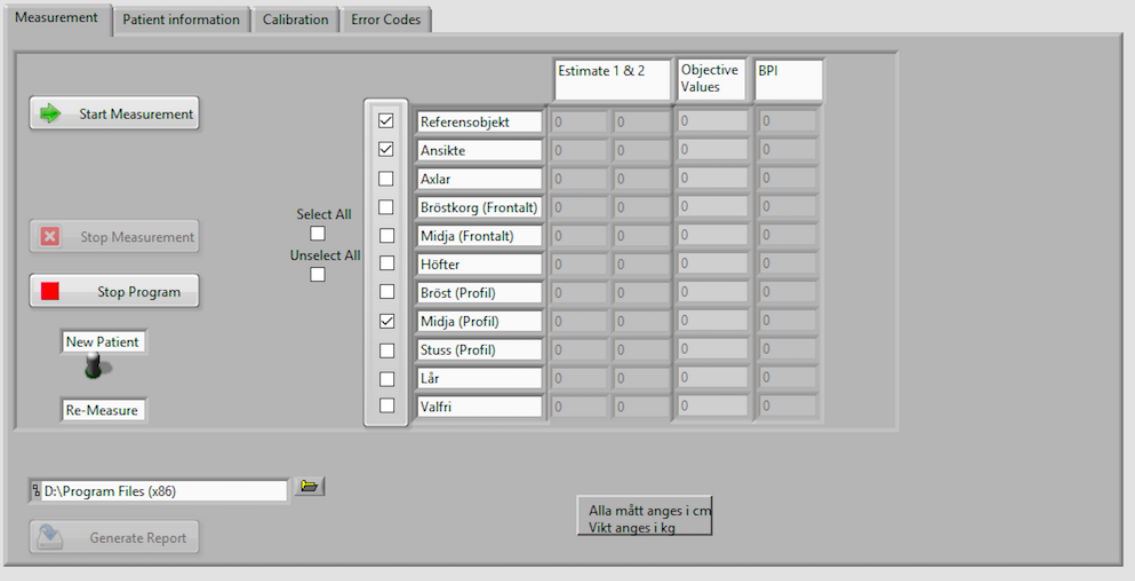
- **Ställa in höjd**
  Styrs av producenten ”Återställ Höjd’. Läser av det nya värdet och ställer in framtida mätningar/kalibrationer efter detta.

4.5 Användargränssnitt

Användargränssnittet är uppdelat i fyra flikar: *Measurement*, *Patient Information*, *Calibration*, och *Error Codes*. Av dessa används tre aktivt medan den fjärde, *Error Codes*, används för att tolka några av de fel som programmet kan stöta på.

Varje flik kommer att visas och förklaras mer ingående i detta avsnitt.

4.5.1 FLIK 1 - MEASUREMENT



Flik 1 - Measurement

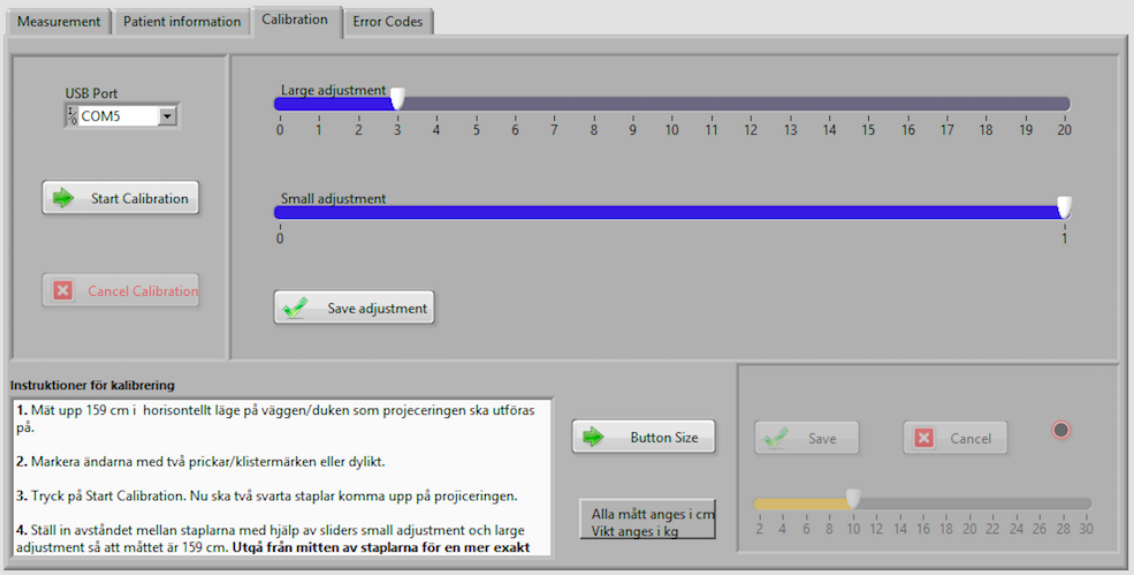
Denna flik är den som kommer att användas mest då det är här själva mätningarna sker. När en mätning ska börja väljs först vilka mätningar som ska utföras genom att använda bockningsrutorna vid mätningsnamnen. Dessutom väljer man mellan två lägen, *New Patient* och *Re-Measure*, där den första nollställer alla värden i matrisen medan den andra bara ändrar de värdena som är valda.

Efter att ha valt vad för sorts mätning samt vilka mätningar som ska exekveras startar man mätningen genom att trycka på knappen *Start Measurement*. Då aktiveras handenheterna och patienten kan se och justera mätpunkterna (i realtid) som används för skattningen på projektorbilden, se bild nedan. Matrisen i fliken uppdateras allteftersom mätningen går vidare vilket gör det möjligt att följa det vad som händer direkt.

Projiceringen på den sekundära skärmen

När en mätning är avslutad kan man generera en rapport på valfri plats genom *Generate Report*.

4.5.2 FLIK 2 - CALIBRATION

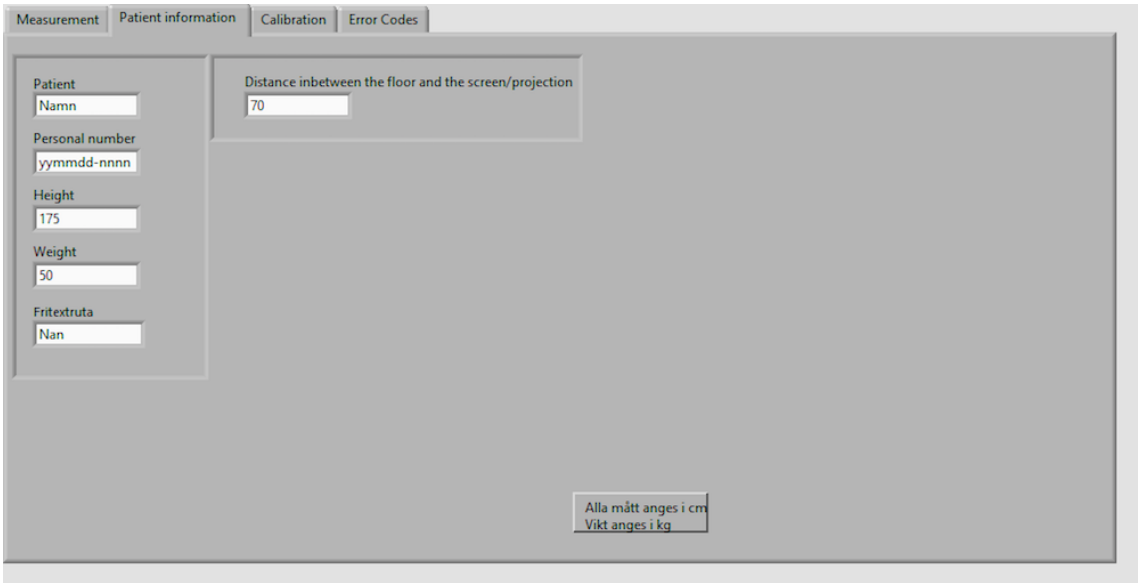


Flik 2 - Calibration

Kalibreringen används för att få rätt skalning i systemet för att räkna ut skattningsvärdena. När man startar kalibreringen så dyker det upp en bild på projektorbilden (likt figuren i 4.5.1 men utan cirklarna) med två staplar där avståndet ska vara 159 cm. Med hjälp av reglagen för grov- och finsjustering flyttas staplarna.

Här man också ändra storleken på de två mätpunkterna (cirklarna i projektionen, se figuren i 4.5.1) som används för när patienterna ska göra sina skattningar.

4.5.3 FLIK 3 - Patient Information



Flik 3 - Patient Information

Patientens namn, personnummer, längd och vikt skrivs in här och kommer med i rapporten när den genereras.

4.5.4 FLIK 4 - Error Codes

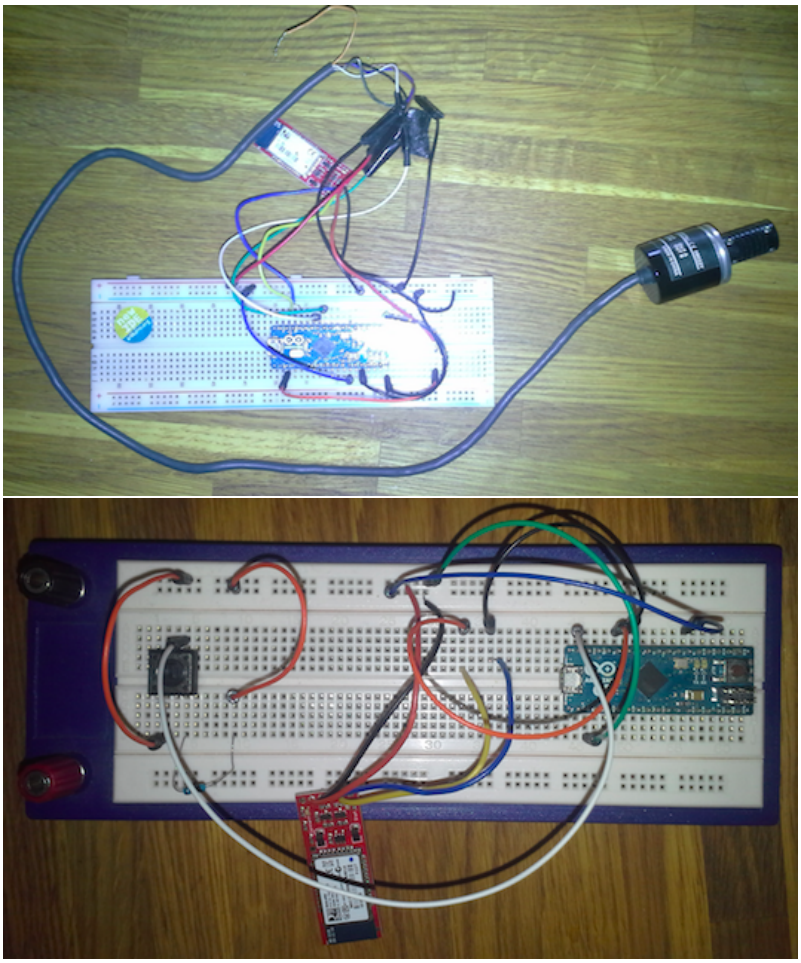
Denna flik innehåller information angående de vanligaste felmeddelanden som kan dyka upp och ger förslag på hur man kan lösa dem.

5 Resultat

5.1 Handenheter

Då inga patientkörningar genomförts diskuteras detta ej.

Det fysiska resultatet av lösningen på produkten är två stycken handenheter, en för patienten (trådbunden) och en för försöksledaren (trådlös). Läs mer om dessa i avsnitt 1.4.



Patientes handenhet (överst) och försöksledarens handenhet (underst)

5.2 Begränsningar i Bluetoothkommunikationen

Den ursprungliga tanken med projektet var att ha båda handenheterna trådlösa med hjälp av Bluetooth-moduler. När detta implementerades visade sig vi gick miste om en del mätpunkter vilket gjorde hela systemet oanvändbart. Troligtvis berodde detta på att Arduinon inte är effektiv på att hantera multipla parallella Bluetooth-anslutningar på en och samma enhet, vilket medförde att vissa mätpunkter från pulsgivaren förkastades.

Lösningen blev istället att bara ha en handenhet trådlös medan den andra fick anslutas med USB-kabel.

5.3 Användargränssnitt

Användargränssnittet är försöksledarens kontrollpanel (för detaljer om hur användargränssnittet, se avsnitt 4.5). Härifrån kan försöksledaren göra följande:

- Kalibrera systemet för bättre precision
- Ändra patientinformation
- Starta, avbryta och avsluta mätningar
- Bestämma vilka mätningar som skall göras (eller göras om)
- Generera rapporter i Excel.

Användargränssnittet är bara synligt på huvudskärmen vilket innebär att det bara är försöksledaren som ser det. På den sekundära skärmen (eller projektorbilden) visas antingen en svart bild (när mätningen ej är igång) eller mätpunkterna med tillhörande namn så att patienten och försöksledaren vet vilken mätning som är igång (se den andra figuren i avsnitt 4.5.1).

6 Diskussion

6.1 Slutsats

Arbetet ses som lyckat då systemet, bestående av två handenheter och ett användargränssnitt, uppfyller de krav som ställdes i början av processen (se avsnitt 1.4).

Det nya systemet har moderniserats och är helt digitalt vilket resulterat i att man nu bl.a använder sig av en extern skärm eller en projektor istället för en ljusramp som begränsar portabilitet. Dessutom medför digitaliseringen att alla beräkningar nu sker automatiskt och omedelbart samt att en rapport (i Excel) kan genereras från systemet, detta ger försöksledaren mer tid till att fokusera sig på patienten.

Patientens handenhet som styr mätpunkterna har fått mycket bättre precision i och med bytet från en vanlig potentiometer (som användes i förbättringsprojektet, se avsnitt 1.2.2) till en pulsgivare och därmed behöver inte handenheten nollställas manuellt genom att vrida ratten tillbaka efter varje mätning. Försöksledarens handenhet är trådlös och möjligör fri rörelse i rummet.

Systemet är nu mycket billigare att tillverka och kommer att medföra snabbare och billigare reparationer.

6.2 Rekommendationer för vidareutveckling

6.2.1 Konvertering till annat programmeringsspråk (Python)

För att systemet ska kunna användas på andra sjukhus runt om i världen krävs just nu en LabVIEW-licens vilken är väldigt dyr. Genom att skriva om programmet till ett programmeringsspråk som inte kräver licens behöver inte denna kostnad tas hänsyn till, vilket medför att vem som helst kan använda det utan extra kostnad.

Ett enkelt språk med bra möjligheter till att bygga upp ett användargränsnitt likt det som byggts i detta projektet, är Python.

6.2.2 Trådlös kommunikation mellan dator och handenhet

Ett naturligt steg för vidareutveckling vore att göra hela systemet trådlöst (t.ex via Bluetooth), dvs. att göra patientens handenhet trådlös också. Detta skulle innebära att ytterliggare Bluetooth-moduler skulle behöva integreras med pulsgivarenheten för att möjliggöra trådlös kommunikation med datorn.

Att tänka på är problemet som uppstår i exekveringstiderna vilket för oss medförde att en del mätdata gick bort och därmed försämrades noggrannheten i systemet.

6.2.3 Strömförbrukning

Mätningar på strömförbrukningen av handenheten med pulsgivaren behöver göras för att veta hur lång batterilivslängden är samt. för att veta hur den kan förbättras (om den anses vara otillräcklig). En grov uppskattning visar att de tre komponenterna har en strömförbrukning som följer:

- Arduino Micro - ca 30mA[16]
- Pulsgivaren - ca.30mA[11]
- Bluesmirf Silver - ca. 30mA[14]

För ett standard från Kjell & Company (2700mAh) fås därför en batterilivslängd på ca 30h. Uppladdningsbara batterier bör användas för att hålla kostnader nere och strömförbrukningen behöver utredas för att försöka sänka den.

6.3 Alternativa lösningar

En alternativ lösning till vår kalibrering hade varit att automatisera detta genom att, vid start av programmet, läsa av den sekundära skärmens (eller projektorbildens) upplösning och jämföra det med den fysiska storleken. Dock så kan detta medföra en viss osäkerhet då det kan inträffa felaktiga felläsningar, därför vore en kombination av manuell och automatisk kalibrering det bästa. Dvs. en automatisk kalibrering vid start av programmet med möjlighet att kalibrera om manuellt.

[16] Arduinons strömförbrukning är svår att bedöma utan mäta. Efter att ha läst igenom andra projekt kan det antas vara under 30mA.

Referenser

[1] Universitetssjukhuset i Lund, ”Ätstörningar hos barn och barn”, https://www.skane.se/Public/psykiatri_skane/bup/atstorning/dok/Behandlingsmanual%20AN%20o%20BN%202.pdf

[2] Wallin, Kronovall & Majewski, ”Body Awareness Therapy in Teenage Anorexia Nervosa: Outcome after 2 years”, Department of Child and Youth Psychiatry, University Hospital of Lund, Oct.1999

[3] Peter Slade, ”A review of body-image studies in Anorexia Nervosa and Bulimia Nervosa”, Liverpool University Medical School, 1985

[4] Peter Slade, ”A review of body-image studies in Anorexia Nervosa and Bulimia Nervosa”, Liverpool University Medical School, 1985

[5] Svensson & Wedding, ”Development of a Labview-based system for estimation of perceived body-image in patients with eating disorders”, LTH - Elektrisk Mätteknik, Dec. 2013

[6] Northwestern University mechatronics design wiki, ”Encoder with a spinning codewheel”, [Elektronisk bild]; http://hades.mech.northwestern.edu/index.php/Rotary_Encoder.

[7] Northwestern University mechatronics design wiki, ”Waveforms of the A & B channels of an encoder”, [Elektronisk bild]; http://playground.arduino.cc/Main/RotaryEncoders.

[8] National Instruments Tutorials, ”Tutorial: State Machines”, http://www.ni.com/tutorial/7595/en/.

[9] National Instruments White Papers, ”Application Design Patterns: Master/Slave”, http://www.ni.com/white-paper/3022/en/.

[10] National Instruments White Papers, "Application Design Patterns: Producer/Consumer", http://www.ni.com/white-paper/3023/en/.

[11] Sparkfun, "Datasheet, YUMO COM-10932", http://cdn.sparkfun.com/datasheets/Robotics/E6A2%20encoder.pdf

[12] Arduino, "Schematic, Arduino Micro", https://www.arduino.cc/en/uploads/Main/arduino-micro-schematic.pdf

[13] Atmel, "Datasheet, ATmega32U4", http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf

[14] Sparkfun, "Datasheet, BlueSMIRF Silver", http://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/Bluetooth-RN-42-DS.pdf

A Bilaga A - Programkod för handenheterna

A.1 Handenhet med pulsgivare

```
#include <Encoder.h>
#include <SoftwareSerial.h>

Encoder myEnc(5, 6);
const int buttonPin = 4;

int buttonState;             // the current reading from the input pin
int lastButtonState = LOW;   // the previous reading from the input pin

long lastDebounceTime = 0;  // the last time the output pin was toggled
long debounceDelay = 50;

char inData[10];       //to read data packet
int index;
int inInt = 0;
boolean started = false;
boolean ended = false;

int bluetoothTx = 9;  // TX-O pin of Bluesmirf silver, Arduino D9
int bluetoothRx = 10;  // RX-I pin of Bluesmirf silver, Arduino D10

SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup() {
  delay(5000);

  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
  pinMode(bluetoothTx, INPUT);
  pinMode(bluetoothRx, OUTPUT);
  bluetooth.begin(115200);
  bluetooth.print("$$$");
  delay(100); // Wait for CMD to come back
```

```
  bluetooth.println("SM,1");
  delay(100);
  //bluetooth.println("C,00066676B302");
  bluetooth.println("C,00066676A0CA");
  delay(100);
  bluetooth.println("---");
  delay(100);
  bluetooth.print("$$$");
  delay(100);
  bluetooth.println("U,9600,N");  // Temporarily Change the baudrate to
                                 //  9600 (until a reboot is done)
  delay(100); // Wait for Ack
  bluetooth.begin(9600);
}

long oldPosition  = 10;
int reset = 0;
int newButtonState = 0;
int oldButtonState = 0;
int encMax = 1590;           // Max och min valute for projecting
int encMin = 10;            // Min  1,01887 <=> 1555

void loop() {
  long newPosition = myEnc.read();
  if (newPosition != oldPosition) {
    if((encMin - 1) < newPosition && newPosition < (encMax + 1)){
      oldPosition = newPosition;
      Serial.println(newPosition);
    } else if(newPosition > encMax) {
      myEnc.write(encMax);
    } else {
      myEnc.write(encMin);
    }
  }

 if (Serial.available() > 0) {
    reset = Serial.read();
    switch (reset) {
```

```
    case 49:
     myEnc.write(0);
     break;
    case 50:
     myEnc.write(encMin);
     Serial.println(newPosition);
     break;
    case 51:
     myEnc.write(encMax);
     Serial.println(newPosition);
     break;
    default:
     break;
    }
  }

while(bluetooth.available() > 0) {
  char aChar = bluetooth.read();
  if(aChar == '<') {
    started = true;
    index = 0;
    inData[index] = '\0';
  } else if(aChar == '>') {
    ended = true;
  } else if(started) {
    inData[index] = aChar;
    index++;
    inData[index] = '\0';
  }
}

if(started && ended) {
  inInt = atoi(inData);    // Convert the string to an integer
  if (inInt == 1) {
    Serial.println("-1");
  }
  started = false;         // Get ready for the next time
  ended = false;
```

```
    index = 0;
    inData[index] = '\0';
  }
}
```

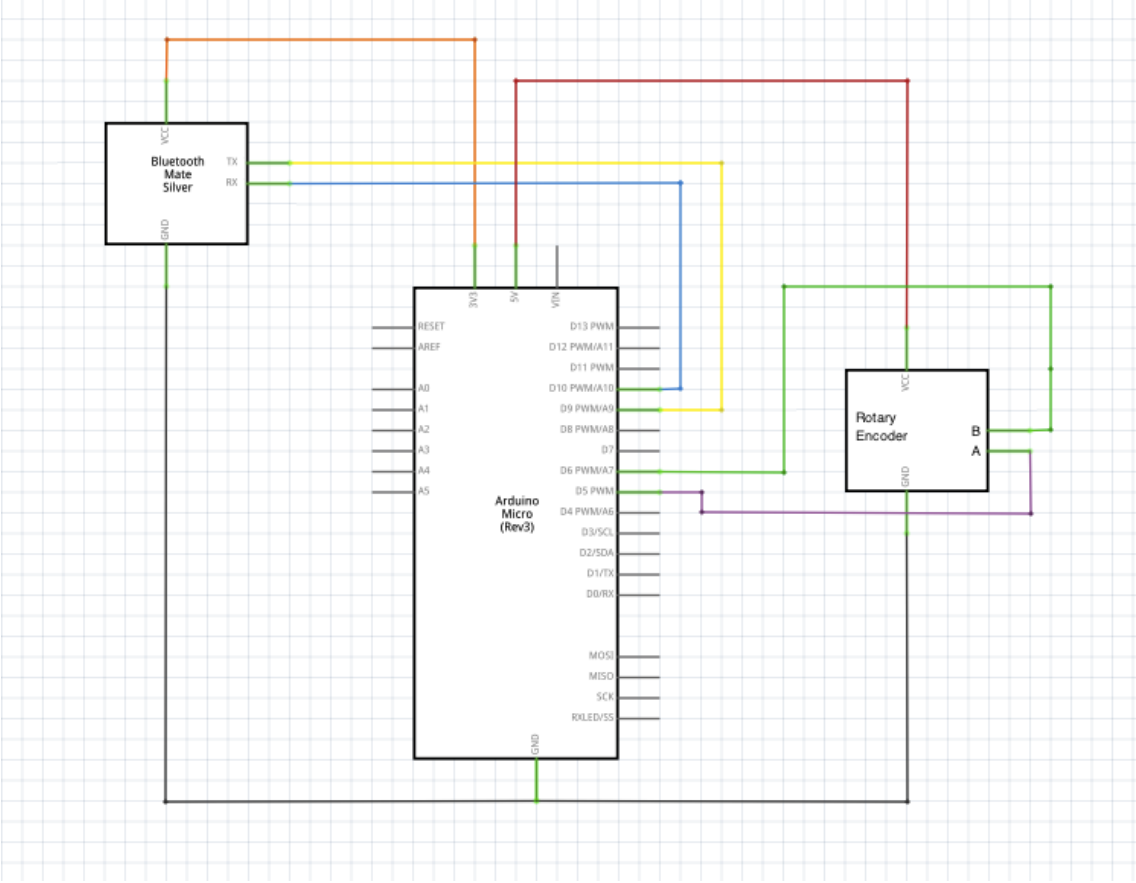
A.2 Handenhet med knapp

```
#include <SoftwareSerial.h>

int buttonState = 0;
int lastButtonState = HIGH;
int buttonPin = 12;

long lastDebounceTime = 0;
long debounceDelay = 50;

String transform;

int bluetoothTx = 7;  // TX-O pin of Bluesmirf silver, Arduino D7
int bluetoothRx = 8;  // RX-I pin of Bluesmirf silver, Arduino D8

SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup() {
  delay(2000);

  pinMode(buttonPin, INPUT);
  pinMode(bluetoothTx, INPUT);
  pinMode(bluetoothRx, OUTPUT);
  bluetooth.begin(115200);
  bluetooth.print("$$$");
  delay(100); // Wait for CMD to come back
  bluetooth.println("SM,0");
  delay(100);
  bluetooth.println("---");
  delay(100);
  bluetooth.print("$$$");
  delay(100);
  bluetooth.println("U,9600,N");  // Temporarily Change the baudrate to
                                  //  9600 (until a reboot is done)
  delay(100); // Wait for Ack
  bluetooth.begin(9600);
  Serial.begin(9600);
```

```
}

void loop() {
  int reading = digitalRead(buttonPin);

  if (reading != lastButtonState) {
    lastDebounceTime = millis();
  }

  if ((millis() - lastDebounceTime) > debounceDelay) {
    if (reading != buttonState) {
      buttonState = reading;
      if (buttonState == LOW) {
        transform = String("<1>");
        bluetooth.print(transform);
        Serial.println(-1);

      }
    }
  }
  lastButtonState = reading;
}
```

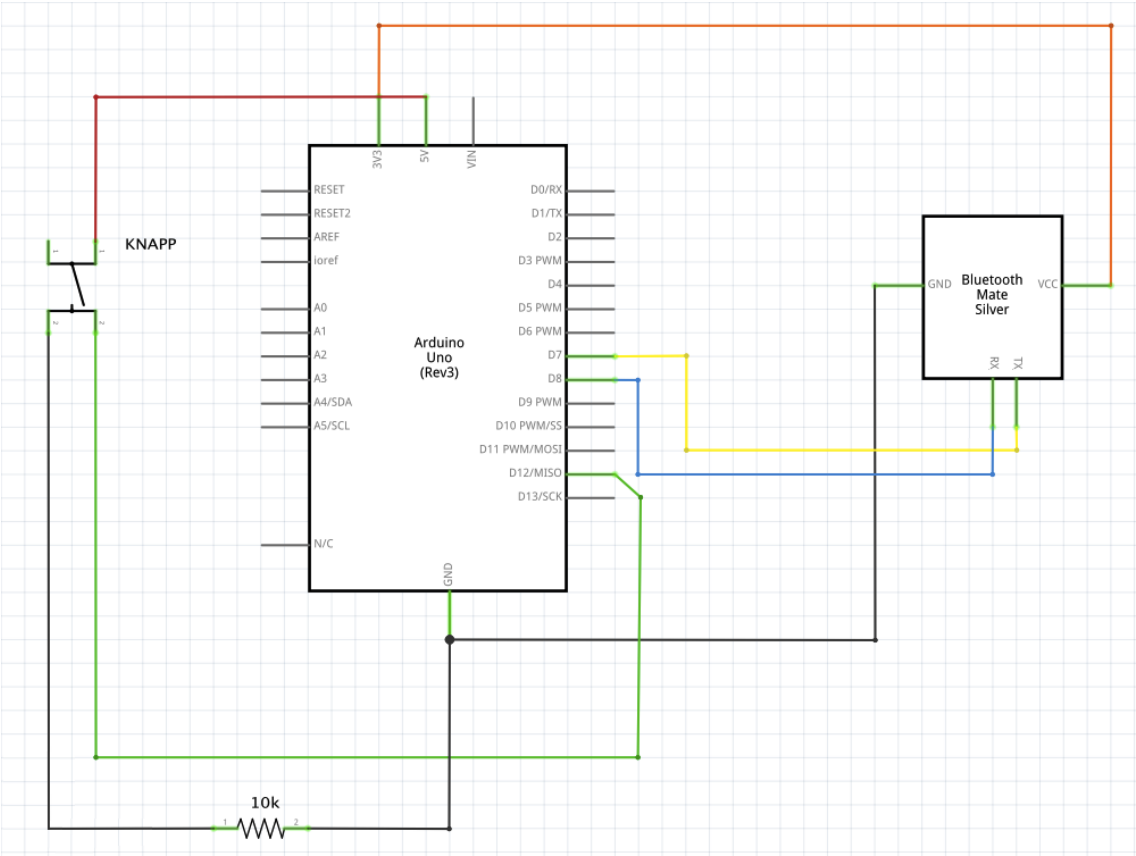
B Bilaga B: Kretsschema för handenheterna

B.1 Handenhet med pulsgivare



Kretsschema för handenhet med pulsgivare

B.2 Handenhet med knapp



Kretsschema för handenhet med knapp