

Gymnasiearbete Handroid

Spårning och representation av fingerrörelser.

Gabriel Calota
Jonathan Damsgaard Falck
William Johansson

Lärosäte: ABB Gymnasiet

Klass: 190S

Handledare: Andreas Jillram, ABB Gymnasiet

Sammanfattning

Abstract

Innehåll

1	Inledning	1
1.1	Syfte	1
1.2	Bakgrund	1
1.3	Frågeställning	1
2	Teori	1
2.1	Elektromagnetism	1
2.1.1	Magnetfält	1
2.1.2	Induktion	1
2.2	Elektronik	2
2.2.1	Spolar	2
2.2.2	Mätning av spolar	2
2.2.3	Kondensatorer	2
2.2.4	Operationsförstärkare	2
2.2.5	Filter	2
2.2.6	Oscillatörer	2
2.2.7	Flexsensorer	3
2.2.8	Accelerometer	3
2.2.9	Arduino Nano 33 BLE	3
2.2.10	555-timer	3
2.3	Bluetooth Low Energy	3
2.3.1	Peripheral	3
2.3.2	Central	3
2.4	Rendering	3
2.4.1	Unity	3
3	Metod och material	4
3.1	Val av metod	4
3.2	Konstruktion	4
3.2.1	3D-hand	4
3.2.2	Spolar	4
3.2.3	Montering på hand	5
3.3	Kretsar	6
3.3.1	Filter	6
3.3.2	Oscillator	6
3.4	Rendering	7
3.4.1	Mikrokontroller	7
3.4.2	Renderingsprogram	7

4 Resultat	7
4.1 Oscillator	7
4.1.1 Spolar	7
4.1.2 Oscillatorkrets	7
5 Diskussion och Slutsats	8
5.1 Spårning	8
5.1.1 Oscillator	8
5.1.2 Filter	8
5.2 Visualisering	8
5.2.1 3D-utskriven hand	8
5.2.2 Digital hand	8
5.3 Planering	8
5.4 Slutsats	8
6 Avslutning	8
7 Källor	8
Bilagor	9
A Renderingskod	9
B Oscillatorkrets	9
C Filterkrets	9
D Kod Spolsnurrare	9
E Kod Arduino Peripheral	11
F Kod Rendering	12

1 Inledning

1.1 Syfte

Syftet med projektet är att undersöka hur människan finmotoriska rörelser kan detekteras, hur rörelserna såsom fingerrörelse kan användas som inmatning till olika system och utveckla ett prototyp som kan både detektera och återskapa fingerrörelser från en människohand. De finmotoriska rörelser kan återskapas antingen med en digital representation av rörelsen eller med en fysisk robothand.

1.2 Bakgrund

Det finns ett ökat intresse och en ökad efterfrågan på olika sätt för människan att interagera med datorer och robotar. Bland annat hur människokroppen kan användas för inmatning till datorprogram och till olika maskiner. Den här tekniken är som mest utvecklad inom virtual reality och augmented reality och det är inom dessa områden som tekniken i nuläget har störst användning. Tekniken är dock fortfarande relativt begränsad och använder mestadels grövre motorik som inmatning och förlorar den precision som kan ges av finmotoriska rörelser.

1.3 Frågeställning

2 Teori

För att få en bättre bild av hur en robothand kan skapas och styras med hjälp av olika sensorer undersöktes andra, liknande projekt. Projektet drog inspiration från en kandidatexamen från två KTH-studenter [1] med ett liknande syfte, och idéer för tummens funktion på den fysiska handen kom från ett projekt som hette *Etho Hand* [2] vars syfte var att utveckla en hand som kunde utföra komplexa rörelser.

2.1 Elektromagnetism

2.1.1 Magnetfält

Magnetfält är fält som rör sig i en riktning från nord- till sydpolen på en magnet eller runt en strömförande elektrisk ledare. Den magnetiska flödestätheten, ekvivalent med magnetfältets *styrka*, kring en ledare med ström, avtar med avstånd från ledaren, vilket är relevant för detta projekt. [3]

2.1.2 Induktion

Inducerad spänning är spänning som alstras när en ledare befinner sig i ett magnetfält, på grund av att de laddade partiklarna hamnar på motsatta sidor av ledaren. Det skapar en skillnad i laddning, vilket korrelerar med spänning. Enligt Lenz lag ger denna spänning upphov till en ström, i en sluten krets, och att strömmens riktning motverkar förändringen av det magnetiska flödet. Denna princip kan användas för att alstra ström ur ett magnetfält, exempelvis i en generator. Induktion fungerar också åt motsatt håll: ström och spänning i en krets ger upphov till magnetfält runt ledaren. [3]

2.2 Elektronik

2.2.1 Spolar

En spole är en ledare som lindats i varv på ett sådant sätt att magnetfältet som skapas när ström flödar igenom har en nord- och en sydpol. Spolar kan lindas runt ett material, en *kärna* som förstärker magnetfältet, eller bara med luft i mitten. På grund av att spolen inducerar spänning kommer den motverka strömmen i kretsen den ingår, enligt Lenz lag, vilket bland annat ger en förskjutning mellan spänning och ström, i en växelströmskrets. [3]

2.2.2 Mätning av spolar

För att dimensionera en krets kan det vara viktigt att veta vilka värden ens komponenter har. Till resistorer och kondensatorer är detta oftast enkelt, då de flesta moderna multimeterna kan mäta resistans och kapacitans. Verktyg för att mäta spolars induktans (som mäts i enheten 1 Henry) är sällsyntare, och därför finns olika metoder för att själv mäta spolar utan specialverktyg. För detta kan exempelvis en funktionsgenerator och ett oscilloskop användas, och genom att utnyttja spolens upp- och urladdningsförmåga kan man med hjälp av matematiska formler approximera spolens induktans. [4]

2.2.3 Kondensatorer

Kondensatorn är en elektrisk komponent som kan lagra elektrisk energi, genom att laddas upp och laddas ur med spänning, vilket ger upphov till en förskjutning mellan ström och spänning, i en växelströmskrets. [3]

2.2.4 Operationsförstärkare

2.2.5 Filter

2.2.6 Oscillatörer

En oscillator är en elektrisk krets eller en del av en elektrisk krets där växelström, ström som regelbundet byter riktning, skapas. På grund av spolars och kondensatorers upp- och urladdningsförmåga kan dessa användas, i olika kombinationer för att göra en krets där strömmen och spänningen svänger (blir svaga/starkare, och byter riktning) med en viss frekvens. Oscillatörer är användbara om man vill skapa växelström med en likströmskälla, vilket för det här projektet är mycket relevant för att kunna alstra ett varierande magnetfält i spolarna.

En sorts oscillator är en Wien-bryggeoscillator (*Wien Bridge Oscillator*) [5] där resistorer, kondensatorer och en operationsförstärkare används för att skapa en vågformad växelströmssignal, där frekvensen bestäms av resistorernas och kondensatorernas värden. En annan sorts oscillator är en 555-oscillator [6], som bygger på ett chip med en integrerad krets (*IC*) vid namn 555. 555-chippet kan ge fyrkantsformade utsignaler om de sätts i ett visst värde, och där frekvensen bestäms med hjälp av resistorer och kondensatorer. Det går också att filtrera signalen så att den liknar en sinusvåg, med hjälp av en spole och kondensatorer.

2.2.7 Flexsensorer

En flexsensor är en sensor som ändrar resistans proportionerligt med hur mycket den är böjd. Flexsensorer kan användas för att mätta hur mycket ett finger är böjd och de har använts i projekt och produkter som

2.2.8 Accelerometer

2.2.9 Arduino Nano 33 BLE

Arduino Nano 33 BLE är en mikrokontroller baserad på kretsen Nordic nRF52480 producerad av företaget Arduino. Arduino Nano 33 BLE är en fysiskt liten mikrokontroller. Nordic nRF52480 är bestyckad med en Cortex M4F processor och en NINA B306 BLE-modul som möjliggör kommunikation med Bluetooth Low Energy. Arduino Nano 33 BLE har 8 analoga ingångar och kan drivas av spänning mellan 4,5 och 21 volt. Annan viktig hårdvaruinformation om mikrokontrollern kan ses i tabell 1. [7]

Mikrokontrollern används för att läsa av mätvärden från sensorerna och för att skicka vidare mätvärdena till datorn där renderingen sker. Den programmeras i C++ med hjälp av platformIO, en tillbyggnad i utvecklingsmiljön Visual Studio Code som möjliggör programmering av mikrokontrollers i C++.

Tabell 1: Viktiga specifikationer Arduino Nano 33 BLE [7]

Mikrokontroller	nRF52840
Driftspänning	3,3V
Inspänning (min)	4,5V
Inspänning (max)	21V
Digitala ingångar/utgångar	14 st
Analoga ingångar	8 st
Längd	45 mm
Bredd	18 mm

2.2.10 555-timer

2.3 Bluetooth Low Energy

Bluetooth Low Energy liknar traditionell Bluetooth, men är mer energieffektiv.

2.3.1 Peripheral

2.3.2 Central

2.4 Rendering

2.4.1 Unity

Unity är en programvara som möjliggör skapandet av real-time 3d-miljöer för spel, film etc. <https://unity.com/our-company>.

3 Metod och material

3.1 Val av metod

Metoden som valdes var att spåra fingrarna med hjälp av spolar på finertopparna som sänder ut ett magnetfält som genom att läsas av skulle kunna ge finrarnas rotation och position. Valet av metod stod mellan tre huvudkandidater. Avläsning av magnetfält från spolar på fingertopparna, flexsensorer för att ta fram fingrarnas rotation, eller flertalet IMU-sensorer som genom att mäta röreslerna på olika delar av handen kan ge data på hur hela handen rör på sig. Metoden med accelerometrar valdes bort tidigt eftersom det var en väldigt avancerad och dyr metod som skulle kräva minst 7 stycken sensorer för att fungera och minst 16 stycken sensorer för att kunna spåra hela handen. Då metoden krävde mycket för att komma igång bedömdes den icke-länplig för projektet. Metoden med flexsenorer var den metod som bedömdes som lättast att genomföra. Metoden valdes bort på grund av att flexsensorer ansågs för begränsande då flexsensorerna endast fanns i specifika storlekar som gjorde att vissa positioner och rotationer skulle vara svåra att mäta. Flex-sensorerna som sensorer var även väldigt dyra och metoden bedömdes vara aningen enkel och hade redan gjorts många gånger tidigare. Dessutom kan en flexsensor endast mäta rörelse i en riktning vilket innebar att precis mätning av alla fingrars rörelser skulle kräva ett stort antal sensorer. Slutligen valdes magnetfält eftersom det verkade som en lovande metod. Magnetfält används till största delen för att spåra grövre rörelser men bedömdes ha potential att mäta även finare röresler. Metoden var även den som var minst beprövad vilket innebar att gruppen skulle behöva ägna sig åt mera testning och utforskning.

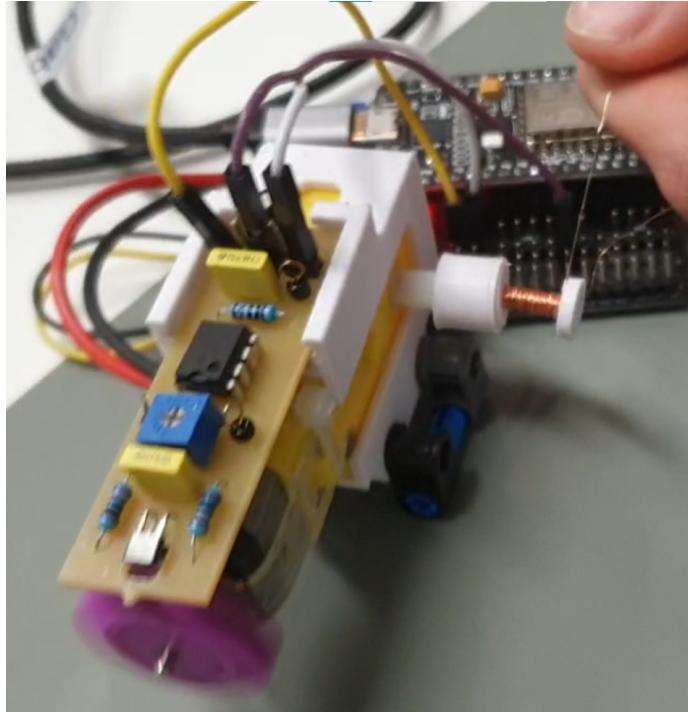
3.2 Konstruktion

3.2.1 3D-hand

En 3D-printad fysisk hand designades som skulle imitera den rörelse som spårats med sensorsystemet. Designen bestod av fingersegment som kunde skrivas ut med en 3D-skrivare och 7 servon som med hjälp av trådar skulle böja fingrarna.

3.2.2 Spolar

Spolar användes för att skapa och läsa av magnetfältet som används för att läsa av fingrets position och rotation. En spole består av en järnkärna, 0,1 mm koppartråd samt 3D-printade hattar. För att få lämplig storlek och funktion från spolarna valdes ett varvtal på 1000 varv. Spolarna konstruerades genom att limma fast hattar i ändarna på järnkärnan. Sedan lindades spolarna med koppartråd. Lindningen gjordes genom att fästa den olindade spolen i en motor som genom att snurra samtidigt som koppartråd var fäst i den olindade spolen gjorde att järnkärnan lindades med koppartråden. Motorn var kopplad till en Arduino mikrokontroller som med hjälp av en varvmätare kunde läsa av hur många varv koppartråd som hade lindats runt spolen. När mikrokontrollern räknat att motorn roterat 1000 varv stängdes motorn automatiskt av. Se figur 1. Kod för spolsnurraren finns i bilaga D



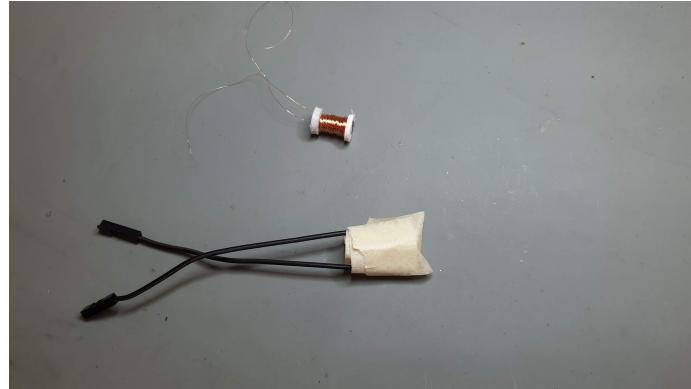
Figur 1: Bild från när en spole lindades med hjälp av Spolsnurraren.

De färdiga spolarna mättes sedan med oscilloskop och funktionsgenerator för att säkerställa dess induktans, för att användas i oscillatorkretsen. Den första spolen som färdigställdes jämfördes också med spolar som används i utbildnings syfte inom fysik, för att säkerställa att de hade samma funktion. Detta genom att använda två fysik-spolar för att skicka och ta emot signaler, där en spole var kopplad i serie med en funktionsgenerator som genererade en sinusformad signal, och där den andra var kopplad till ett oscilloskop för att läsas av. En av fysik-spolarna byttes sedan ut mot den egensnurrade spolen, för att se om signalen kunde skickas och tas emot av dessa.

För att få bättre kontakt med andra kablar och komponenter valdes ett antal spolar ut för att lödas fast i kablar av större tjocklek. Jämförelsen mellan en spole med fastlödd kabel och en spole utan fastlödd kabel kan ses i figur 2.

3.2.3 Montering på hand

För att montera spolar på handen monterades de tre mottagarspolarna på ett kretskort som agerade spolhubb. De tre spolarna monterades i varsin riktning så att de kan ta upp fingerrörelser i tre dimensioner. Kretskortet modellerades i ultiboard och frästes sedan ut på skolan kretskortsfräs. På kretskortet monterades spolarna genom att limmas fast i kretskorten och kopparträden löddes fast på kortets undersida. För att kunna skicka vidare signalen som spolarna plockar upp löddes även en stiftlist fast på kretskortet.



Figur 2: En spole med endast hatt, kärna och lindning (överst) och en spole med kablar fastlödda (underst).

3.3 Kretsar

3.3.1 Filter

Filtret ska filtrera frekvenser som inte är inom ett visst frekvensområde och ett aktivt bandpassfilter valdes som filtertyp. Filterkretsen designades först med hjälp av formler för att bestämma värden på kondensatorer och resistorer. Kretsen testades sedan i simulering för att säkerställa att kretsens fungerar som förväntat, dock användes en ideal operationsförstärkare under simuleringen. När kretsen sedan simulerades med icke-ideal operationsförstärkare kunde kretsen inte längre filtrera signalen. Det visades sig att formlerna för kondensatorer och resistorer antog att operationsförstärkare var ideal för att kretsen skulle fungera. Kretsen designades sedan om med hjälp av ett filter designverktyg av Texas Instruments och fungerade i simulering med en icke-ideal operationsförstärkare.

3.3.2 Oscillator

För de olika fingrarna bestämdes att oscillatorerna skulle ha frekvenser från 100 kHz, med ungefär 15 kHz mellanrum för att ge utrymme för filtrernas bandbredd. Fem oscillatorer behövdes, ett för varje finger. Ursprungligen planerades att använda oscillatorer som bygger på wienbryggor (Wien-bryggesoscillator). Till detta skulle behövas resistorer, kondensatorer och operationsförstärkare. Resistorer och kondensatorer dimensionerades och kretsen simulerades i program för att kontrollera att allt stämde. Efter förslag om att integrera spolarna i kretsen som en resonanskälla byttes oscillatorerna mot 555-oscillatorer. Här dimensionerades istället två olika kretsar i en: en med resistorer och kondensatorer, och en med kondensatorer och spolen.

Två oscillatorer byggdes i verkligheten: en med frekvensen 100 kHz och en med frekvensen 117 kHz. I 117 kilohertz-kretsen användes ett 556-chip, som har samma funktioner som två stycken 555:or; en på varje sida av komponenten, och som drar mindre effekt. Se bilaga för kretsschema.

3.4 Rendering

3.4.1 Mikrokontroller

Mikrokontrolleten Arduino Nano 33 BLE används för att kommunicera med renderingsprogrammet. Mikrokontrolleten väntar på renderingsprogrammet att ansluta och läser sedan av värdena från de olika sensorerna och publicerar dessa till en Bluetooth characteristics. För att åstadkomma kommunikation används biblioteket ArduinoBLE.

3.4.2 Renderingsprogram

Ursprungligen skulle ett CSharp program skapas som kunde kommunicera med mikrokontrolleten som sedan också skulle rendera fingerrörelserna. Istället för att skapa ett eget renderingsprogram beslutades det att spelmotorn Unity skulle användas. Kommunikationen mellan mikrokontrolleten och renderingsprogrammet sköttes därav istället av biblioteket BleWinrtDll som är går att direkt integrera med Unity och en färdig konstruerad modell från Ultraleaps Unity Plugin användes för att representera fingerrörelse. Unity programmet skulle anslutas till mikrokontrolleten som sedan skulle översätta värdena från sensorerna till rotationer i fingrarnas ledar. Programmet testades genom att skicka två specifika värden från mikrokontrolleten till Unity-programmet som roterade ett fingers ledar till två olika rotationer beroende på vilket värde som skickats.

4 Resultat

4.1 Oscillator

4.1.1 Spolar

De egentillverkade spolarna hade samma funktion som andra fabrikstillverkade spolar, enligt den jämförelse som gjordes. Signalen var något förändrad vad gäller amplitud, men hela signalen kunde tas emot. Störningar från omgivningen var dock något större, enligt resultaten som kunde ses i oscilloskop.

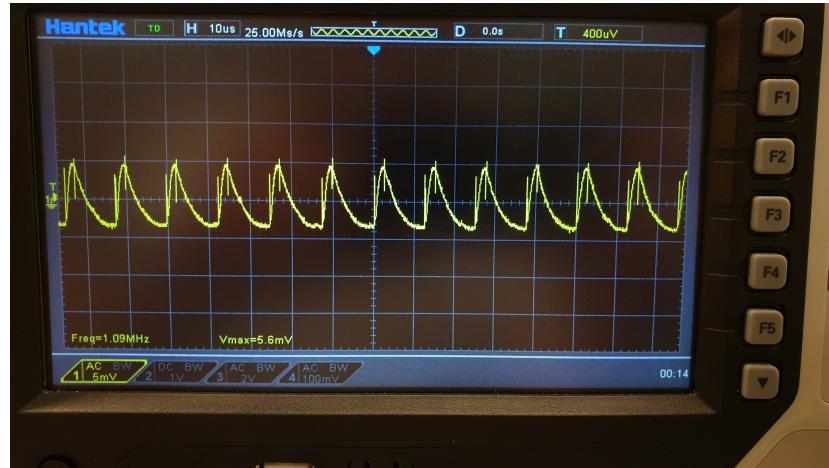
Första mätningen av de första tre spolarnas induktans gav en genomsnittlig induktans på 3,3 mH vilket användes vid dimensionering av oscillatorer. Vid en andra omgång mätningar mättes fem andra spolar, där den genomsnittliga induktansen var runt 7,0 mH, vilket ledde till att oscillatorkretsarna dimensioneerades om.

Under testning gick vissa spolar sönder, och lödningen släppte på två av fyra lödda spolar under laborationer och tester.

4.1.2 Oscillatorkrets

I simuleringar gav oscillatorkretsarna signaler med amplituder (maxvärdet) från 4 V till 5 V. Kretsen som dimensionerades för 100 kHz, som använde ett 555-chip, gav signaler som endast skilje sig från omgivningens störningar med någon millivolt, och vars frekvens rimligtvis kunde antas vara ungefär 10 gånger större än 100 kilohertz. Då spolen som ingick i kretsen gick sönder kunde fler tester inte utföras.

Kretsen som skulle ge 117 kHz, med 556-chippet, gav en signal med amplitud på cirka 5,5 mV, och frekvens som varierade mellan 115 kHz och 130 kHz. Signalen var inte helt sinusformad men oscillerade mellan positiv och negativ spänning i sågtandsliknande vågor. Se figur 3.



Figur 3: Oscillatorbild av signal från 117 kHz-oscillatorkretsen. Nere till vänster kan man se spänningens maxvärde, och hur stor frekvens signalen har (med störningar).

5 Diskussion och Slutsats

5.1 Spårning

5.1.1 Oscillator

5.1.2 Filter

5.2 Visualisering

5.2.1 3D-utskriven hand

5.2.2 Digital hand

5.3 Planering

5.4 Slutsats

6 Avslutning

7 Källor

- [1] M. Kazi och M. Bill, *Robotic Hand Controlled by Glove Using Wireless Communication*, 2020.

- [2] C. Konnaris, C. Gavriel, A. A. Thomik och A. A. Faisal, "EthoHand: A dexterous robotic hand with ball-joint thumb enables complex in-hand object manipulation", i *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2016, s. 1154–1159. DOI: 10.1109/BIOROB.2016.7523787.
- [3] A. Kullander Sjöberg, S. Nilsson, L. Boström och B. Ekstig, *Fysik för gymnasieskolan 1 och 2 Digital*. Natur & Kultur, 2019.
- [4] R. Dekker, *A Simple Method to Measure Unknown Inductors*, <https://www.dos4ever.com/inductor/inductor.html>, Hämtad 2022-05-10.
- [5] Electronics Tutorials, *The Wien Bridge Oscillator*, https://www.electronics-tutorials.ws/oscillator/wien_bridge.html, Hämtad 2022-05-10.
- [6] Learning about Electronics, *How to Build a Sine Wave Generator with a 555 Timer Chip*, <http://www.learningaboutelectronics.com/Articles/Sine-wave-generator-circuit-with-a-555-timer.php>, Hämtad 2022-05-10.
- [7] *Arduino® Nano 33 BLE*, ABX00030, Rev. 1, Arduino, april 2021.

Bilaga A

Renderingskod

Bilaga B

Oscillatorkrets

images/555-117kHz (Bild läggs in senare) bild på själva breadborden kanske?

Bilaga C

Filterkrets

Bilaga D

Kod Spolsnurrare

```
#include <Arduino.h>

#define hastighetssensor 12 //D6
#define motorPinDir 0 //D2
#define motorPinSpeed 5 //D1

unsigned long n_pulses = 0;
```

```

unsigned long time1 = 0;
unsigned long time2 = 0;
int n_turns = 0;
bool one = false;

ICACHE_RAM_ATTR void sensorCallback() {
    n_pulses++;
}

void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200);

    attachInterrupt(digitalPinToInterrupt(hastighetssensor),
        → sensorCallback, RISING);

}

void loop() {

    if (n_pulses % 96 == 0 && one) {
        n_turns++;
        one = false;
        Serial.print(n_turns);
        Serial.print(" Since last: 0.");
        Serial.print((millis() - time1));
        Serial.print(" ms ETA: ");
        Serial.print(((millis() - time1) *
            → (1000-n_turns))/(1000*60));
        Serial.print(" min ");
        Serial.print(((millis() - time1) *
            → (1000-n_turns))/(1000)%60);
        Serial.println(" s");
        time1 = millis();
    }
    if (n_pulses % 96 != 0) {
        one = true;
    }

    if (n_turns >= 1000) {

        digitalWrite(motorPinDir, 1);
        analogWrite(motorPinSpeed, 0);
    }
    else {
        digitalWrite(motorPinDir, 1);
        analogWrite(motorPinSpeed, 512);
    }
}

```

```
// put your main code here, to run repeatedly:  
}
```

Bilaga E

Kod Arduino Peripheral

```
#include <Arduino.h>  
#include <ArduinoBLE.h>  
#include <functions.h>  
  
const char *deviceServiceUuid = "190H";  
const char *deviceServiceCharacteristicUuid = "190F";  
  
String value;  
  
String val = "0100,0100,0100";  
unsigned long prevTime;  
int advertising;  
  
BLEService handService(deviceServiceUuid);  
BLEStringCharacteristic  
→ fingerCharacteristic(deviceServiceCharacteristicUuid, BLERead  
→ | BLEWrite | BLENotify, 14);  
  
void setup()  
{  
    Serial.begin(9600);  
    while (!Serial)  
        ;  
  
    if (!BLE.begin())  
    {  
        Serial.println("- Starting BLE module failed!");  
        while (1)  
            ;  
    }  
  
    BLE.setLocalName("Arduino Handroid");  
    BLE.setAdvertisedService(handService);  
    handService.addCharacteristic(fingerCharacteristic);  
    BLE.addService(handService);  
    fingerCharacteristic.writeValue(val);  
    BLE.advertise();  
  
    Serial.println("Nano 33 BLE (Peripheral Device)");  
    Serial.println(" ");
```

```

}

void loop()
{
    BLEDevice central = BLE.central();
    Serial.println("- Discovering central device...");

    if (central)
    {
        Serial.println("* Connected to central device!");
        Serial.print("* Device MAC address: ");
        Serial.println(central.address());
        Serial.println(" ");

        prevTime = millis();
        while (central.connected())
        {
            writeValues();
        }

        Serial.println("* Disconnected to central device!");
    }
}

void writeValues()
{
    fingerCharacteristic.writeValue(val);
    if (millis() - prevTime > 5000)
    {
        if (val == "0100,0100,0100")
        {
            val = "1000,1000,1000";
        }
        else
        {
            val = "0100,0100,0100";
        }
        prevTime = millis();
    }
}

```

Bilaga F

Kod Rendering

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;

```

```

public class handController : MonoBehaviour
{
    public Demo bluetoothScript;
    float speed = 0.1f;
    int[] orientation;

    public GameObject index_a;
    public GameObject index_b;
    public GameObject index_c;

    void Start()
    {
        GameObject bluetoothGameObject = GameObject.Find("Demo");
        bluetoothScript =
            → bluetoothGameObject.GetComponent<Demo>();
    }

    void Update()
    {
        orientation = Array.ConvertAll<string,
            → int>(bluetoothScript.subscribeText.text.Split(','), int.Parse);

        // Test to verify that Arduino Nano 33 BLE can control
        → Unity hand model
        if (orientation[0] == 100)
        {
            Quaternion toRotation = Quaternion.Euler(0, 0, 0);
            index_a.transform.localRotation =
                → Quaternion.Lerp(index_a.transform.localRotation,
                → toRotation, Time.time * speed);
            index_b.transform.localRotation =
                → Quaternion.Lerp(index_b.transform.localRotation,
                → toRotation, Time.time * speed);
            index_c.transform.localRotation =
                → Quaternion.Lerp(index_c.transform.localRotation,
                → toRotation, Time.time * speed);
        }
        if (orientation[0] == 1000)
        {
            Quaternion toRotation = Quaternion.Euler(0, 0, -85);
            index_a.transform.localRotation =
                → Quaternion.Lerp(index_a.transform.localRotation,
                → toRotation, Time.time * speed);
            index_b.transform.localRotation =
                → Quaternion.Lerp(index_b.transform.localRotation,
                → toRotation, Time.time * speed);
        }
    }
}

```

```
    index_c.transform.localRotation =
    ↵    Quaternion.Lerp(index_c.transform.localRotation,
    ↵    toRotation, Time.time * speed);
}

}
```