SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR
SZÁMÍTÁSTECHNIKA SZAK

3D nyomtatott kéz vezérlése EMG jelekkel és képfeldolgozással

DIPLOMADOLGOZAT

Témavezető:
Dr. Fehér Áron, adjunktus

Végzős hallgató:
Magyar Roland István

2024

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE, TÎRGU-MUREȘ
PROGRAMUL DE STUDII CALCULATOARE

Reglarea mâinilor imprimate 3D cu ajutorul semnalelor EMG și a procesării imaginilor

PROIECT DE DIPLOMĂ

Coordonator ştiinţific:
Ș.l. dr. ing. Fehér Áron

Absolvent:
Magyar Roland István

2024

SAPIENTIA HUNGARIAN UNIVERSITY OF TRANSYLVANIA
FACULTY OF TECHNICAL AND HUMAN SCIENCE, TÎRGU-MUREȘ
COMPUTER SCIENCE SPECIALIZATION

Regulation of 3D printed hands with EMG signal measurement and image processing

DIPLOMA THESIS

Témavezető:
Dr. Fehér Áron Ph.D., eng.
assistant professor

Végzős hallgató:
Magyar Roland István

2024

UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA Facultatea de Științe Tehnice și Umaniste din Târgu Mureș Specializarea: **Calculatoare**	**Viza facultății:**

LUCRARE DE DIPLOMĂ

Coordonator științific: **ș.l. dr. ing. Fehér Áron**	Candidat: **Magyar Roland-István** Anul absolvirii: **2024**

a) Tema lucrării de licență:
Reglarea mâinilor imprimate 3D cu ajutorul semnalelor EMG și a procesării imaginilor

b) Problemele principale tratate:
- Studiu bibliografic privind achiziția de date EMG
- Proiectarea și testarea unei sisteme analogice de achiziție de date EMG
- Dezvoltarea unui software de luare a deciziilor care mișcă un braț protetic pe baza datelor EMG și a procesării imaginilor

c) Desene obligatorii:
- Spectrograma măsurătorilor EMG
- Diagrama Bode a filtrelor analogice
- Proiectarea schematică a interfeței analogice

d) Softuri obligatorii:
- Simularea filtrelor analogice in LTSpice
- Software de achiziție a datelor EMG și de imagini bazat pe Python

e) Bibliografia recomandată:
- Guobiao Yang, Zhenyuan Sun, Zhao Wang, Gangdong Sun, "Detection and Identification of the Human Surface Electromyography Signal".
- Muhammad Zahak Jamal, "Signal Acquisition Using Surface EMG and Circuit Design Considerations for Robotic Prosthesis".

f) Termene obligatorii de consultații: săptămânal
g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca,
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș
Primit tema la data de: ...15.05.2023.....
Termen de predare: .....10.07.2024.....

Semnătura Director Departament	Semnătura coordonatorului
Semnătura responsabilului programului de studiu	Semnătura candidatului

Declarație

Subsemnata/Subsemnatul Fehér Áron, funcția șef lucrări.,
titlul științific dr.ing declar pe propria răspundere că absolvent Magyar Roland István, de la specializarea Calculatoare a întocmit prezenta lucrare sub îndrumarea mea.

În urma verificării formei finale constat că lucrarea de licență/proiectul de diplomă/disertația corespunde cerințelor de formă și conținut aprobate de Consiliul Facultății de Științe Tehnice și Umaniste din Târgu Mureș în baza reglementărilor Universității Sapientia. Luând în considerare și Raportul generat din aplicația antiplagiat „Turnitin" consider că sunt îndeplinite cerințele referitoare la originalitatea lucrării impuse de Legea educației naționale nr. 1/2011 și de Codul de etică și deontologie profesională a Universității Sapientia, și ca atare sunt de acord cu prezentarea și susținerea lucrării în fața comisiei de examen de licență/diplomă/disertație.

Localitatea, Târgu Mureș

Data: Semnătura îndrumătorului

Declaraţie

Subsemnatul/a Magyar Roland Istvan, absolvent(ă) al/a specializării Calculatoare, promoţia 2024 cunoscând prevederile Legii învățământului superior nr. 199 din 2023 şi a Codului de etică şi deontologie profesională a Universităţii Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licenţă/proiect de diplomă/disertaţie se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informaţiile şi datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Localitatea: Targu Mures

Data: 19 iunie, 2024

Absolvent: Magyar Roland Istvan

Semnătura:

Kivonat

A bionika kutatása és fejlődése kulcsfontosságú azok számára, akiknek fizikai funkcióik helyreállítására vagy kiegészítésére van szükségük. Az emberi testrészek működése hasonló a robotokéhoz, amelyek különféle jelekre reagálnak és mozgásokat végeznek. Ugyanígy az emberi test is reagál elektromos impulzusokra, amelyeket megfelelően elkapva és irányítva, továbbíthatunk vagy eszközök működésébe integrálhatunk.

A dolgozatom célja egy 3D nyomtatott kéz vezérlésének bemutatása valós idejű videó és EMG jelek segítségével, lehetővé téve a robotkar ujjainak mozgását az emberi kéz mozgásával összhangban. Az eszköz már képes feldolgozni az EMG jeleket, és egy 3D modell, valamint a szükséges szoftver már elkészült, amely a kamera képét feldolgozva vezérli a motorokat.

A dolgozat bevezeti a bionika fogalmát, párhuzamot vonva az emberi testrészek és a robotok működése között, hangsúlyozva az elektromos impulzusok szerepét. Áttekinti a neuromuszkuláris folyamatokat, az anatómiát és az elektromiográfia (EMG) alapelveit. Bemutatja a projekt elektronikai komponenseit, tárgyalva a zajkezelést is. Feltárja a képfeldolgozási technikákat a robotkéz vezérléséhez, részletezve a digitális jelfeldolgozást és a rendszerintegrációt. Ismerteti a rendszer specifikációit és architektúráját, a modellek és motorok kiválasztását, valamint a mikrovezérlőket és EMG szenzorokat.

A rendszer tervezése magában foglalja a kéz 3D modelljének előkészítését, a motorok tesztelését és az EMG elektródák integrálását. A firmware fejlesztése a WiFi hozzáférési pont megvalósítására, az ADC olvasására és az EMG jel tüske (spike) alapú detektálására összpontosít. Bemutatja a képfeldolgozó szoftvert is, amely a mikrovezérlőnek továbbítja az adatokat, leírva a küszöbértéktechnológiákat az emberi kéz mozgásának észlelésére.

Az utolsó fejezet általános következtetéseket von le és javaslatokat tesz a jövőbeli munkákra, megvitatva a rendszer lehetséges fejlesztéseit és bővítéseit, például további szenzorok integrálását és fejlettebb jelfeldolgozó algoritmusok alkalmazását. Ez a projekt bemutatja, hogy lehetséges 3D nyomtatott robotkéz vezérlése EMG jelekkel és képfeldolgozással, új lehetőségeket nyitva meg a fejlett protézisek fejlesztése előtt.

Kulcsszavak: képfeldolgozás, digitális jelfeldolgozás, EMG, Python, ESP32

Extras

Cercetarea și dezvoltarea în bionică joacă un rol esențial în viața persoanelor cu abilități variate, ajutând la restaurarea sau completarea funcțiilor fizice. Funcționarea corpului uman este similară cu cea a roboților, ambele răspunzând la semnale electrice pentru a efectua mișcări. Prin capturarea și controlarea acestor impulsuri, putem transmite semnalele în zone unde conexiunea directă este întreruptă sau le putem integra în dispozitive.

Scopul tezei mele este de a demonstra controlul unei mâini imprimate 3D folosind video în timp real de la o cameră și semnale EMG pentru mișcarea degetelor robotice. Dispozitivul procesează semnalele EMG, iar un model 3D și un program controlează motoarele pentru mișcarea degetelor.

Teza introduce conceptul de bionică și relevanța sa, discutând analogia dintre funcționarea corpului uman și a roboților. Se revizuiește literatura de specialitate, analizând procesele neuromusculare, anatomia și principiile electromiografiei (EMG), tipurile de electrozi EMG și activitatea electrică a mușchilor. Se prezintă componentele electronice ale proiectului, precum amplificatoarele operaționale, filtrele și conversia analog-digitală, și se discută gestionarea zgomotului electronic. Se explorează tehnicile de procesare a imaginilor pentru controlul mâinii robotice și integrarea semnalelor digitale în sistem. Se detaliază arhitectura sistemului, cerințele funcționale și nefuncționale, modelele și motoarele utilizate, microcontrolerele și senzorii EMG. Designul sistemului include pregătirea modelului 3D al mâinii și testarea motoarelor, etapele de procesare a semnalelor și integrarea electrozilor EMG.

Capitolul despre dezvoltarea firmware-ului explică implementarea punctului de acces WiFi, citirea ADC și detectarea vârfurilor (spike) din semnalul EMG. Se dezvoltă software-ul pentru procesarea imaginilor și transmiterea datelor către microcontroler, descriindu-se tehnicile de prag pentru detectarea mișcărilor mâinii umane. Capitolul despre punerea în funcțiune a sistemului și rezultatele experimentale prezintă pașii necesari pentru instalare și măsurătorile efectuate pentru a evalua performanțele sistemului.

Ultimul capitol oferă concluzii generale și direcții pentru lucrări viitoare, discutând îmbunătățiri posibile, extinderea sistemului cu alți senzori și algoritmi avansați de prelucrare a semnalelor, și compararea cu dispozitive similare.

Acest proiect demonstrează că este posibil să controlăm o mână robotică imprimată 3D folosind semnale EMG și prelucrarea imaginilor, deschizând noi oportunități pentru dezvoltarea dispozitivelor protetice avansate.

Cuvinte cheie: prelucrare de imagini, prelucrare digitală de semnale, EMG, Python, ESP32

Abstract

Research and development in bionics are crucial for individuals with disabilities, aiding in the restoration or augmentation of physical functions. The functioning of human body parts is akin to robots, responding to various signals and performing movements. By capturing and controlling these electrical impulses, we can transmit them to areas where direct connection is severed or integrate them into devices.

The aim of my thesis is to demonstrate the control of a 3D-printed hand using real-time video from a camera, enabling robotic finger movements based on human hand movement. Additionally, it allows extraction of hand movement information from EMG signals, enabling control of the robotic hand with a microcontroller. The device processes EMG signals, and a 3D model and necessary software have been developed to process camera images and control the motors for appropriate finger movement.

The thesis introduces the concept of bionics and its relevance, drawing parallels between the human body and robots, and emphasizing the role of electrical impulses in movement. It reviews relevant literature, focusing on neuromuscular processes, anatomy, and principles of electromyography (EMG), including types of EMG electrodes and muscle electrical activity.

It provides an overview of the electronic components used, such as operational amplifiers, filters, and analog-to-digital conversion, discussing electronic noise management. Image processing techniques for controlling the robotic hand are explored, detailing digital signal processing and system integration. Specifications and system architecture are discussed, including functional and non-functional requirements, and choices for models and motors. The microcontrollers and EMG sensors used are also described.

System design includes preparing the 3D model of the hand, testing the motors, processing signals, and integrating EMG electrodes. Firmware development focuses on implementing the WiFi access point, ADC reading, and spike detection of the EMG signal. Software development for image processing and data transmission to the microcontroller is also covered, describing thresholding techniques for detecting hand movements.

The implementation and experimental results chapter outlines installation steps and system performance measurements.

The final chapter provides general conclusions and suggests future work, discussing possible system improvements and extensions, such as integrating additional sensors and advanced signal processing algorithms. It compares similar devices and analyzes potential development directions.

This project demonstrates the feasibility of controlling a 3D-printed robotic hand using EMG signals and image processing, opening new opportunities for advanced prosthetic devices.

Keywords: image processing, digital signal processing, EMG, Python, ESP32

Tartalomjegyzék

1. Bevezető **1**

2. Irodalom Kutatás **2**
2.1. A neuromuszkuláris folyamat 2
2.2. Anatómia 4
2.3. Elektromiográfia 5
2.3.1. Tűelektródák 6
2.3.2. Finom huzal elektródák 6
2.3.3. Felületi EMG elektródák 6
2.3.4. Izmok elektromos aktivitása 7
2.4. Elektronikai áttekintés 8
2.4.1. Zaj 8
2.4.2. Műveleti erősítők 9
2.4.3. Szűrők 11
2.4.4. Analóg-digitális átalakítás 17
2.5. Képfeldolgozás 18

3. A rendszer specifikációi és architektúrája **19**
3.1. Funkcionális követelmények 19
3.2. Nem funkcionális követelmények 20
3.3. A rendszer architektúrája 20
3.3.1. Model kiválasztása 20
3.3.2. Motorok kiválasztása 23
3.3.3. Motor 23
3.3.4. Mikrovezérlő 24
3.3.5. EMG szenzorok 24

4. Részletes tervezés **24**
4.1. Tömbvázlat 24
4.2. Hardver rész 27
4.2.1. Modell előkészitése 27
4.2.2. Motorok tesztelése 27
4.2.3. Jel feldolgozása 28
4.2.4. EMG elektródák 29
4.2.5. ESD védelem 30
4.2.6. DC elutasitás 31
4.2.7. Erősités 32

4.2.8. DRL és Shield . . . 34
4.2.9. 50 Hz-es hálózati zajszűrő . . . 34
4.2.10. 10Hz-1kHz sávszűrő . . . 35
4.2.11. Analóg-digitál átalakitó . . . 39
4.2.12. Tápegység áramkör . . . 39
4.3. Firmver . . . 40
4.3.1. Loop függvény . . . 40
4.3.2. WiFi Acces Point . . . 40
4.3.3. EMG jel kezelése . . . 43
4.3.4. ADC olvasás . . . 44
4.3.5. Négyzetes átlag számítás . . . 45
4.3.6. Spike Detektálás . . . 46
4.4. Szoftver rész . . . 47
4.4.1. Kép feldolgozása . . . 47
4.4.2. Adatok továbbitása a mikrovezérlőnek . . . 51
4.5. Küszöbérték (Thresholding) . . . 53

5. Üzembe helyezés és kísérleti eredmények 54
5.1. Üzembe helyezési lépések . . . 54
5.2. Kísérleti eredmények, mérések . . . 54

6. Következtetések 56
6.1. Megvalósítások . . . 56
6.2. Hasonló eszközök . . . 56
6.2.1. Ada Robotic Hand - Open Bionics . . . 56
6.2.2. Touch bioinics . . . 57
6.2.3. Saját eszköz . . . 57
6.3. További Fejlesztési Lehetőségek . . . 57

A. Függelék 60

Ábrák jegyzéke

1. Neuromuszkuláris folyamat . . . 2
2. Az izomrostok akciós potenciáljának időbeli lefutása. . . . 4
3. Alkarban levő izmok . . . 5
4. Frekvencia szerinti besorolása . . . 5
5. Tűelektróda és finom huzal elektródák . . . 6
6. Géles és száraz elektróda . . . 7
7. Izom elektromos aktivitása . . . 7
8. Spektromgram . . . 8
9. Műveleti erősítő . . . 10
10. Műszererősítő . . . 10
11. Szűrő részei . . . 11
12. Szűrő rendje . . . 12
13. Elsőrendű aktív aluláteresztő szűrő . . . 13
14. Elsőrendű aktív aluláteresztő szűrő feszültség erősitéssel . . . 13
15. Aluláteresztő szűrő frekvenciamenetválasza . . . 15
16. Elsőrendű aktív felüláteresztő szűrő . . . 15
17. Elsőrendű aktív felüláteresztő szűrő feszültség erőditéssel . . . 16
18. Felüláteresztő szűrő frekvenciamenetválasza . . . 17
19. OpenCV kéz tájékozódási pontjai . . . 18
20. A rendszer architektúrája . . . 20
21. K1 kéz . . . 21
22. K2 kéz . . . 21
23. Gael Langevin és az InMoov robot . . . 22
24. K3 kéz . . . 23
25. A rendszer egy tömbként . . . 25
26. A jelfeldolgozó rész folyamat ábrája . . . 25
27. A képfeldolgozó rész folyamat ábrája . . . 26
28. Motor ágy és motor kerekek 3D modelje . . . 27
29. Izom szenzor modul . . . 28
30. A jelfeldolgozó rendszer egy tömbként . . . 29
31. Átfogó tervezési tömbvázlat . . . 29
32. Az EKG-jel jellemzői . . . 30
33. Feszültséghullámformák ESD védelemmel és anélkül . . . 31
34. Passzív AC-kapcsoló hálózat . . . 32
35. Ina826 erősitő áramköri rajza . . . 33
36. 50hz kivágó áramkör . . . 34
37. 50hz kivágó áramkör szimuláció . . . 35

38. Szűrő és másodlagos erősítő áramkör 36
39. Sallen key 37
40. 50hz kivágó áramkör szimuláció 38
41. Referencia hozzáadása a jelhez 38
42. ADC használata mikrokontrolerrel 39
43. Jelfeldolgozás fázisai [10] 46
44. A kép feldolgozásának tömbvázlata 47
45. Küszöbérték választás 53
46. Kapcsolási rajz 54
47. A nyers jel 55
48. A megszűrt jel 55
49. Ada Robotic Hand 56
50. Bemeneti védelem, insztrumentális erősítés és közös módusú zaj visszacsatolás 60
51. Analóg szűrő és analóg-digitális jelátalakító 61
52. Tapegység áramkör 62

Táblázatok jegyzéke

1. Motor Teszt 28

1. Bevezető

A bionika kutatása és fejlődése rendkívül fontos szerepet játszik azoknak az embereknek az életében, akiknek eltérő képességeik vannak, és akiknek fizikai funkcióik helyreállítására vagy kiegészítésére van szükségük. Az emberi testrészek működése hasonló a robotokhoz, amelyek különféle jelekre reagálnak és mozgásokat hajtanak végre. Ugyanígy az emberi test is reagál különböző jelekre, amelyek különböző műveleteket indítanak el. Ezeket a jeleket elektromos impulzusoknak nevezzük. Ha ezeket az impulzusokat megfelelően érzékeljük és irányítjuk, képesek vagyunk például olyan területekre továbbítani, ahol a közvetlen kapcsolat megszakadt, vagy feldolgozni és beépíteni egy eszköz működésébe. Emellett a képfeldolgozás technikái lehetővé teszik, hogy kamerafelvételek alapján is felismerjük és interpretáljuk a testmozgásokat vagy a gesztusokat, így a kart nemcsak elektromos jelekkel, hanem vizuális információkkal is lehet irányítani. Ezáltal az emberi test és a technológia határai közötti kapcsolódási pontok, az ún. "ember-gép interfészek", lehetőséget teremtenek az ember és a gépek közötti egyre szorosabb együttműködésre és kölcsönös támogatásra. Ez a fejlődés áttöréseket hozhat az egészségügyben és az életminőség javításában, valamint új perspektívákat nyithat meg a mozgássérült és más kihívásokkal küzdő emberek számára.

Az (EMG) elektromiográf jelek gyűjtésének és feldolgozása lehetővé teszi, hogy megszakadt kapcsolatot helyreállítsunk vagy elektromos jelek alapján vezéreljünk egy műszaki eszközt. Ebben a projektben azt célozzuk meg, hogy egy olyan eszközt fejlesszünk ki, amely képes az EMG jelek fogadására és feldolgozására, majd ezek alapján irányítani egy robotkart. A projekt fő hangsúlya a rehabilitációs területre esik, ahol olyan személyeknek segíthet, akik veleszületett rendellenességekkel vagy amputációval küzdenek.

A projekt egyik célja, hogy megvalósitsunk egy olyan rendszert, amely lehetővé teszi a kari EMG jelek mérését és feldolgozását. Ennek érdekében tervezünk egy lapot, amely képes lesz érzékelni az EMG jeleket, és azokat megfelelően feldolgozni. Emellett célunk egy olyan alkalmazás kifejlesztése, amely egy mikrokontroller segítségével értelmezi az EMG jeleket, és összekapcsolja azokat a rendszerrel. Ezáltal lehetőségünk nyílik egy teljes körű rendszer létrehozására, amely képes az EMG jelek alapján irányítani egy robotkéz mozgásait.

A projekt másik célja a képfeldolgozás alkalmazása a robotkéz irányításában. Ennek érdekében olyan megoldást tervezünk, amely lehetővé teszi, hogy valós idejű videofelvétel alapján irányítsuk a robotkéz mozgásait szintén egy mikrokontroller segítségével.

2. Irodalom Kutatás

Ahhoz hogy elkezdjem a projektet, szükséges pár alapvető dolgot tisztázni, még pedig hogy milyen részeket érint a projekt. Ez pedig a kar és a kéz anatómiájánál kezdődik. Azt mindenki tudja, hogy a karunkat az izmaink segítségével mozgatjuk, de azt már kevesebben tudják, hogy az izmainkat hogyan mozgatjuk.

2.1. A neuromuszkuláris folyamat

Először is az agy kémiai-elektromos jelet állít elő. Ezután a jel az agyból a gerincvelőbe kerül. Végül a neuronok hosszú axonjaikon keresztül továbbítják a jelet a célizomhoz. Amikor a jel megérkezik az izomba, az izom összehúzódását okozza (1. ábra).

1. ábra. Neuromuszkuláris folyamat [1]

Amikor úgy döntünk, hogy megmozgatunk egy izmot, egy pozitív töltésű nátrium ion tartalmú anyag beköltözik a neuronokba. Ezeknek a pozitív töltéseknek az idegsejtbe való mozgása apró „kapuk” nyílását okozza az axon teljes hosszában, lehetővé téve, hogy még több pozitív töltésű nátrium ion kerüljön be az idegsejtekbe. Amikor ez az elektromos-kémiai töltés az izom közelébe érkezik, egy kémiai jel szabadul fel a neuronból, amely az izom összehúzódását vagy mozgását okozza. Az, hogy az izom milyen erősen húzódik össze, attól függ, hogy hány jel érkezik és milyen gyorsan érkeznek. Ha felveszünk egy tollat, az agy csak néhány jelet küld, mert nem kell sok erő ahhoz, hogy valami ilyen könnyű dolgot mozgasson. De ha

felveszünk egy nehezebb tárgyat, akkor gyorsabban több jel kerül kiküldésre, mivel nehezebb és több erőfeszítést igényel [1].

Az izomsejteket egy félig áteresztő lipid membrán, a sarcolemma veszi körül. Pihenő állapotban a membránon átívelő feszültség van, úgy hogy a sejtmembrán belseje körülbelül -90mV-ra van a külsőhöz képest. Normál pihenőállapotban a Na+ ionok koncentrációja nagyobb a membrán külső oldalán, mint a belső oldalon; az ellenkezője történik a K+ ionokkal (relatíve alacsony a külső oldalon és magas a belső oldalon). Ezek az állapotok hozzák létre a pihenő potenciált. Amikor az izomszövetet aktiválják, az izomrostok belső része ideiglenesen pozitívabbá válik. Ehhez először a membránon lévő kapuk megnyílnak, ami lehetővé teszi a nátriumionoknak a bejutást az izomsejtbe. Ha ez az átalakulás elég erős, és elérik a megfelelő küszöbértéket, akkor a nátriumkapuk kinyílnak, és a nátriumionok beáramlanak a sejtbe, ideiglenesen megfordítva a sejt polaritását. Ezután egy rövid időszak következik, amikor a sejt nem tud újabb ingerre reagálni, ez az úgynevezett refrakter periódus. A feszültségvezérelt nátriumcsatornák zárva vannak, és a káliumcsatornák megnyílnak, lehetővé téve a K+ ionok keresztülhaladását a membránon, ki a külső felé. Ebben a pontban a helyzet az ellenkezője, mint a kezdeti állapotban: a K+ magas koncentrációja kívül van, és a Na+ magas koncentrációja a szövet belsejében van. Végül a nátrium-kálium pumpa visszaállítja a Na+ és K+ koncentrációkat energiaköltséggel. Mivel több izomrost innervezált, az ilyen típusú neuron tüzelése több izomrost aktivál. Az összes izomrost tevékenységének összege hozza létre a motoros egység akciós potenciálját (MUAP) [1].

Látható a 2. ábrán, hogy az első szakaszban (1) a membrán még pihenő állapotban van -90mV-on. A második szakaszban (2) depolarizáció megtörténik és ha elérik a küszöbértéket, a potenciál +10 mV-ig emelkedik. A harmadik szakaszban (3) a nátriumcsatornák bezáródnak. Majd a negyedik szakaszban (4) repolarizáció, amely azonos kezdeti feltételekkel végződik az (5) szakaszban [1].

[1] https://dysphagiacafe.com

2. ábra. Az izomrostok akciós potenciáljának időbeli lefutása. [1]

2.2. Anatómia

Mivel a projektem a kéz és alkarra fektet hangsúlyt, ezért ezeket a részeket vettem át.

A **Flexor Digitorum** a felső végtagokban megtalálható, az alkar belső felén. Az izom hosszú inakon keresztül fut a hajlított ujjak irányába. A hajlítás során az izom összehúzódik, az ujjak a tenyér felé hajlanak, ez lehetővé teszi a tárgyak megfogását (3. ábra) .

Az **Extensor Digitorum** az alsó karcsontokon található, és hosszú inak segítségével fut az ujjakig, áthaladva a kéz hátoldalán. Amikor az izom összehúzódik, az inakon keresztül nyújtó hatást fejtenek ki, és az ujjakat kinyújtják a kézfej felé, ez segit a stabilitásban (3. ábra) [2].

[2] https://www.assh.org/handcare/safety/muscles
[3] https://backyardbrains.com/experiments/muscleactionpotential

3. ábra. Alkarban levő izmok [3]

2.3. Elektromiográfia

Az EMG vagy elektromiográfia egy olyan eljárás, amely segít az izmok elektromos aktivitásának mérésében és rögzítésében. Az EMG segítségével az izmok által termelt elektromos jeleket lehet észlelni és rögzíteni. Ezek a jelek az izmok összehúzódásakor vagy mozgásakor jönnek létre az idegsejtek aktivitása miatt. Ezek a jelek spektrum frekvenciájúak 10 Hz és 1 kHz között változnak, és a feszültségcsúcsok 1 µV és 100 mV között vannak. A fő zaj a környezeti források és a jelátalakító zaja[4].

4. ábra. Frekvencia szerinti besorolása

Az EMG elektródáknak két fő típusa van: **invazív** (vagy bőrelektródák) és **nem invazív** elektródák. A behelyezett elektródáknak további két típusa van: tű és finom huzal elektródák.

2.3.1. Tűelektródák

A tűelektródákat széles körben használják klinikai eljárásokban neuromuszkuláris értékeléseknél. A tűelektródákból érkező jel minősége viszonylag jó a többi rendelkezésre álló típushoz képest. A tűelektródáknak két fő előnye van. Az egyik hogy kis felvevőfelülete lehetővé teszi az elektróda számára az egyes MUAP-ok észlelését kis erejű összehúzódások során. A másik, hogy az elektródák kényelmesen áthelyezhető más izomba. Hátránya hogy hordozásuk kellemetlen.

2.3.2. Finom huzal elektródák

A huzal elektródák rendkívül finomak, könnyen beültethetők és kivehetők az izmokból, és általában kevésbé fájdalmasak, mint a tűelektródák [2] (5. ábra).

5. ábra. Tűelektróda és finom huzal elektródák [2]

2.3.3. Felületi EMG elektródák

Ezek az elektródák kémiai egyensúlyt alkotnak a detektáló felület és a test bőre között elektrolitikus vezetés révén, így az áram befolyhat az elektródába. Mivel ezeket az elektródákat a bőrre helyezik, ezért általában csak a felületes izmok kezelésére használják.

Kétféle felületi elektródáka létezik, mégpedig:

A **géles** felületi elektródákat (6. ábra bal kép) egy vezető gél réteggel látják el pl Ag/AgCl amelyet közvetlenül a bőrre helyeznek. Ez a gél segít csökkenteni az elektromos ellenállást és javítja az elektromos kapcsolatot a bőr és az elektódák között. A géles elektródák nagyon jól érzékelik az elektromos jeleket, és a gél csökkenti a zavaró zajokat azáltal, hogy javítja az érintkezést a bőrrel.

A **száraz** felületi elektrodákat (6. ábra jobb kép) gél vagy más vezetőanyag nélkül használják, ilyenek pl az arany elektródák. A száraz elektrodák általában nagyobb elektromos ellenállással rendelkeznek, ami befolyásolhatja az érzékelés minőségét és lehetővé teheti a zavarokat [2].

6. ábra. Géles és száraz elektróda [2]

2.3.4. Izmok elektromos aktivitása

Az izom elektromos aktivitása, vagyis az EMG jel, két fő összetevőből áll: az alapszintből (baseline) és az izomaktivitásból (muscle activity), ezt szemlélteti a 7. ábra is.

Az alapszint a nyugalmi állapotban lévő izom elektromos aktivitását jelenti. Ez a jelrészlet általában alacsony amplitúdójú és zajszintű, és jelen van akkor is, amikor az izom nincs aktívan megfeszítve. Az izomaktivitás az izom összehúzódásakor jelentkező elektromos aktivitás, ami sokkal magasabb amplitúdójú jeleket eredményez, mint az alapszint.

7. ábra. Izom elektromos aktivitás

8. ábra. Spektromgram

Az EMG jelek frekvenciakomponenseinek elemzésére a STFT (Short Time Fourier Transform) módszert alkalmazzuk, amely az FFT (Fast Fourier Transform) ablakozott változata. Ez lehetővé teszi, hogy a jelet rövid időintervallumokra bontva, folytonosan ábrázoljuk anélkül, hogy a Gibb-hatással kellene számolnunk.

A STFT segítségével előállított spektrum grafikusan ábrázolja, hogy az eredeti időbeli jel milyen frekvenciaösszetevőket tartalmaz az adott időablakokban. Az alábbi ábra szemlélteti a STFT alkalmazását az EMG jelre, amely révén megfigyelhetők a jel domináns frekvenciái időben. A spektrogram színkódokkal jelöli a frekvenciatartományok erősségét vagy intenzitását. A 8 ábrán az élénk színek a domináns aktivitást emelik ki, jelezve a jelentős frekvenciakomponenseket.

2.4. Elektronikai áttekintés

Az EMG jelekről, az EMG mérésekről és ezek fontosságáról érdemes egy áttekintést végezni. A tervezési folyamatban érintett elméleti szempontok, mint például a zaj és a zavarok felvétele, differenciális mérés stb. lesznek tárgyalva.

2.4.1. Zaj

Zajnak nevezünk, minden olyan nem kívánt jelet, ami elhomályosítja vagy torzítja a vizsgált rendszer által kiadott választ. A zavarok gyakran külső forrásokból származnak, például elektromos vagy elektrosztatikus kapcsolódásból más eszközökkel vagy külső zajforrásokkal vagy akár mechanikai rezgésekből. Ezeknek a zavaroknak a nagy része „ember által készített”, és minimalizálható vagy kiiktatható. A zaj viszont véletlenszerű jel, és nehéz előrejelezni. A

zaj szinte minden elektromos rendszerben jelen van, és bár nem lehet teljesen megszüntetni, megpróbálhatjuk minimalizálni vagy befolyásolni. A zaj különösen problémás lehet érzékeny méréseknél, mert megnehezítheti a valódi jelek pontos észlelését és feldolgozását [1].

Három fő zajmechanizmus létezik: Hőzaj, alacsony frekvenciájú vagy „flicker” zaj, közülük az első a leggyakoribb.

Termikus zaj: A termikus zajt a vezetőkben mozgó elektronok véletlenszerű mozgása okozza, amelyet a hőmérséklet hatására történő termikus gerjesztés eredményez. Ez a zaj a vezető anyagok hőmozgásából ered, és arányos a hőmérséklettel. Ez a leggyakoribb zajforrás az elektronikai áramkörökben, és független a vezető áramtól.

Alacsony frekvenciájú vagy „flicker” zaj: Az alacsony frekvenciájú zaj olyan zajtípus, amely alacsony frekvencián változik, és gyakran tartalmazza a hosszú távú ingadozásokat. Flicker zajként is ismert, és a spektrum sűrűsége növekszik, ahogy a frekvencia csökken. Ez a zaj jelen van minden aktív eszközben, a tranzisztoroktól és diódáktól egészen biológiai rendszerekig.

Lövési zaj: Ez a zajtípus gyakori diódákban, tranzisztorokban és más elektronikai eszközökben, amelyek "potenciális akadályokkal" rendelkeznek. A lövési zaj azokban az eszközökben bekövetkező véletlenszerű potenciális akadály-átugrást eredményezi, amelyekben a töltéshordozók nem egyenletes vagy folyamatos módon áramlanak, hanem egy nagyszámú véletlenszerűen független áramimpulzus összegének hatásaként [1].

2.4.2. Műveleti erősítők

A **differenciális jel** két jel különbségét jelenti, míg a **közös módú jel** a két jel átlaga. A differenciális jel az, ami általában érdekel, mert ez tartalmazza az információt, amit továbbítani vagy mérni szeretnénk. A közös módú jel viszont olyan interferenciák és zajok eredménye, amelyek a két jel közös földpontja és környezete miatt jelentkeznek, és általában nem tartalmaznak semmilyen hasznos információt, ezért ezt el kell távolítani vagy elnyomni.

Az **op-amp** (műveleti erősítő) és az **in-amp** (műszererősítő) mindkettő olyan elektronikai eszközök, amelyek a jelek erősítésére szolgálnak, de különböző alkalmazásokhoz használják őket.

Az **op-amp** (9. ábra) egy általános célú erősítő, amely sokféle alkalmazásban megtalálható, például hangszóró előerősítő fokozatában, szűrőkben vagy érzékelők kimeneti jeleinek erősítésében. Az op-ampnak két bemenete van, egy inverz és egy nem inverz, és egy kimenete. Ideális esetben az op-amp végtelen bemeneti impedanciával, végtelen erősítéssel, és zérus kimeneti impedanciával rendelkezik. Ezek a tulajdonságok azt jelentik, hogy az op-amp ideális módon erősíteni tudja a differenciális jeleket, azaz a két bemenet közötti különbséget. Az op-ampok kimeneti jele a bemeneti jelek különbségének egy megszorzott verziója.

[4] https://www.electrorules.com/understanding-negative-feedback-in-operational-amplifiers-op-amps/

9. ábra. Műveleti erősítő [4]

Az **in-amp**-ok (10. ábra) a legalkalmasabbak gyenge jelek felerősítésére. Ezek az eszközök hatékonyan kiszűrik és kizárják azokat a zavaró jeleket, amelyek mindkét bemeneten jelen vannak. Általában magas bemeneti és alacsony kimeneti impedanciával rendelkeznek, valamint alacsony a bemeneti bias áramuk. Az in-amp kimeneti jele csak a két bemeneti jel közötti különbséget erősíti, és figyelmen kívül hagyja a közös jeleket. Ezért ideálisak gyenge jelek pontos mérésére és feldolgozására.

10. ábra. Műszererősítő [5]

A bemeneti **bias** áram az a tényleges áram, amely beáramlik az erősítőbe, hogy polarizálja az összes belső tranzisztort. Fontos biztosítani egy DC utat ennek az áramnak. Egy erősítő, amely nem rendelkezik ezzel az útvonallal, nem tud megfelelően működni. A megfelelő működéshez szükséges DC földelés hiánya gyakori hiba a tervezésben.

[5] https://en.wikipedia.org/wiki/Operational_amplifier_applications

2.4.3. Szűrők

A szűrő egy olyan eszköz, amely bizonyos frekvenciájú vagy frekvenciatartományú elektromos jeleket enged át, miközben megakadályozza mások áthaladását. A digitális adatgyűjtő rendszerekben az anti-aliasing szűrők mindig szükségesek az optimális mintavételezés biztosításához, de más szakaszokban is használják a jelek előkészítésére, például a hasznos információk sávjának kiválasztására.

Számosféle szűrő létezik, és a funkciójuktól függően osztályozhatók: sávzáró, alul-, felül-, vagy sáváteresztő szűrők stb. Egy másik osztályozás a felhasznált komponensek típusától függ: passzív vagy aktív szűrők. A passzív szűrők ellenállásokból, kondenzátorokból és tekercsekből állnak, ezért RLC szűrőknek is nevezik őket. [1].

Általában három része van (11. ábra):

- áteresztő tartomány (Passband)
- átmeneti tartomány (Transition band)
- vágási tartomány (Stop band)

11. ábra. Szűrő részei

A **szűrőrend** az, hogy hány pólusa van a szűrőegyenletnek, és azt mutatja, mennyire gyorsan vált a szűrő az áteresztő és vágási tartomány között. Ha magasabb a rendje a szűrőnek, akkor gyorsabban vált az átmeneti sávban, de több alkatrészt és szakaszt igényel (12. ábra). Három fő analóg szűrő típus van: Butterworth, Chebyshev és Bessel, amelyeket a szűrőegyenletben található pólusok paramétereinek változtatásával állítanak be [1].

12. ábra. Szűrő rendje

2.4.3.1. Alul áteresztő szűrő Az alul áteresztő szűrő olyan eszköz, ami engedi, hogy alacsonyabb frekvenciájú jelek átmenjenek rajta, de blokkolja a magasabb frekvenciákat. Az ideális esetben a frekvenciaválasz görbéje a vágási frekvenciánál esik le. Gyakorlatilag a jel nem esik hirtelen, hanem fokozatosan esik át az átmeneti régióból az elnyomási sáv régiójába.

A jel elnyomása, azaz a kimeneti jel amplitúdója kisebb, mint a bemeneti jel amplitúdója a passzív áramkörben. A passzív szűrő e hátrányának leküzdése érdekében aktív szűrőt terveznek. Az aktív aluláteresztő szűrő egy passzív aluláteresztő szűrő, amelyet vagy inverz vagy nem inverz műveleti erősítőhöz kapcsolnak[6].

Az elsőrendű aktív aluláteresztő szűrő egyetlen műveleti erősítőből és egy RC áramkörből áll (13. ábra).

13. ábra. Elsőrendű aktív alulátersztő szűrő

A műveleti erősítőnek egységnyi egyenáramú erősítése van, Av = +1 vagy egységnyi erősítés, ellentétben a passzív RC szűrővel, amelynek egyenáramú erősítése kevesebb, mint egységnyi. A műveleti erősítő magas bemeneti impedanciája megakadályozza, hogy túlzott terhelést jelentsen a szűrő kimenetén, míg alacsony kimeneti impedanciája megakadályozza, hogy a szűrő vágási frekvenciapontja megváltozzon a terhelés impedanciájának változásával. Bár ez a konfiguráció jó stabilitást biztosít a szűrőnek, fő hátránya, hogy nincs 1 feletti feszültségerősítése és a teljesítményerősítés nagyon magas, mivel a kimeneti impedanciája sokkal kisebb, mint a bemeneti impedanciája[6].

14. ábra. Elsőrendű aktív alulátereszto szűrő feszültség erősitéssel

Az áramkör frekvenciaválasza ugyanaz lesz, mint a passzív RC szűrőé, kivéve, hogy a kimeneti amplitúdó megnő a szűrő áteresztő tartományával (Passband) 11. ábra, az erősítő által. Egy nem inverz erősítőáramkör esetén a szűrő feszültségnyeresége abszolút értéke a visszacsatolási ellenállás (R2) és a hozzá tartozó bemeneti ellenállás (R1) hányadosaként adódik.

[6] https://www.electronics-tutorials.ws/filter/filter_5.html

$$A_f = 1 + \frac{R2}{R1} \tag{1}$$

Az aktív aluláteresztő szűrő erősitése a következőképpen kapható meg a frekvencia komponensek alapján:

$$\text{Erősítés}(A_v) = \frac{V_{\text{out}}}{V_{\text{in}}} = \frac{A_f}{\sqrt{1 + \left(\frac{f}{f_c}\right)^2}} \tag{2}$$

Ahol:

- Af = áteresztő sáv erősitése
- f = a bemeneti jel frekvenciája
- fc = a vágási frekvencia
- Vout = kimeneti feszültség
- Vin= bemeneti feszültség

Alacsony frekvenciákon, azaz amikor a bemeneti jel frekvenciája kisebb a vágási frekvenciánál ($f < fc$), akkor

$$\frac{V_{\text{out}}}{V_{\text{in}}} \cong A_f \tag{3}$$

Amikor a bemeneti jel frekvenciája egyenlő a vágási frekvenciával (f = fc), akkor

$$\frac{V_{\text{out}}}{V_{\text{in}}} = \frac{A_f}{\sqrt{2}} = 0.707A_f \tag{4}$$

Amikor a bemeneti jel frekvenciája sokkal nagyobb a vágási frekvenciánál ($f > fc$), akkor

$$\frac{V_{\text{out}}}{V_{\text{in}}} < A_f \tag{5}$$

Így az aktív aluláteresztő szűrőnek állandó erősitése van (AF), 0 Hz-től a magas frekvenciájú vágási pontig, fc-ig. fc-nél az erősités 0,707AF, és fc után egyenletesen csökken a frekvencia növekedésével. Tehát amikor a frekvencia tízszeresére nő, a feszültség erősités tízzel oszlik. Minden tízszeres frekvencianövekedés esetén, az erősités 20 dB-el csökken. A -3 dB pont a szűrő átviteli függvényén az a pont, ahol a jelerősség eléri a maximumot. Ez a pont jelöli a szűrő használható tartományának a kezdetét. Ahogy eltávolodunk a -3 dB-es ponttól, a szűrő egyre kevésbé erősíti a jelet, és végül erősen csillapítja azt a leállítási sávban (15. ábra) [7].

[7] https://www.electronicshub.org/active-low-pass-filter/

15. ábra. Aluláteresztő szűrő frekvenciamenetválasza

2.4.3.2. Felüláteresztő szűrő Az alul áteresztő szűrőkre vonatkozó összes fogalom alkalmazható a felül áteresztő szűrőkre is. A tervezési folyamata rendkívül egyszerű, mert ugyanazt követi, mint az alul áteresztő szűrők esetén, csak az a különbség, hogy a komponensek cserélődnek fel: ellenállások helyére kondenzátorok kerülnek, és fordítva.

A felül áteresztő szűrő azokat a frekvenciákat engedi át, amelyek magasabbak a vágási frekvenciánál, és elnyomja azokat a frekvenciákat, amelyek alacsonyabbak a vágási frekvenciánál. Az elnyomás mértéke vagy a passz sáv tartománya a szűrő tervezési paramétereitől függ [8].

Az elsőrendű aktív felüláteresztő szűrő egyetlen műveleti erősítőből és egy RC áramkörből áll (16. ábra).

16. ábra. Elsőrendű aktív felülátersztő szűrő

Hasonlóképp itt is egységnyi az erősités:

$$A_{\max} = 1$$

[8] https://www.electronics-tutorials.ws/filter/filter_6.html

Ahhoz, hogy az erősitésünk nagyobb legyen egységnyitől, ugyanazt az eljárást alkalmazzuk mint az aktiv elsőrendű aluláteresztő szűrőnél, a visszacsatolásba beteszünk két ellenállást az erősités érdekében.

17. ábra. Elsőrendű aktív felülátersztő szűrő feszültség erősitéssel

Az aktív felüláteresztő szűrő erősités a következőképpen fog kinézni:

$$\text{Erősités}(A_v) = \frac{V_{\text{out}}}{V_{\text{in}}} = \frac{A_f\left(\frac{f}{f_c}\right)}{\sqrt{1+\left(\frac{f}{f_c}\right)^2}} \tag{6}$$

Ahol:

- A_f = áteresztő sáv erősitése
- f = a bemeneti jel frekvenciája
- f_c = a vágási frekvencia
- V_{out} = kimeneti feszültség
- V_{in}= bemeneti feszültség

Alacsony frekvenciákon, azaz amikor a bemeneti jel frekvenciája kisebb a vágási frekvenciánál ($f < fc$), akkor

$$\frac{V_{\text{out}}}{V_{\text{in}}} < A_f \tag{7}$$

Amikor a bemeneti jel frekvenciája egyenlő a vágási frekvenciával (f = fc), akkor

$$\frac{V_{\text{out}}}{V_{\text{in}}} = \frac{A_f}{\sqrt{2}} = 0.707A_f \tag{8}$$

Amikor a bemeneti jel frekvenciája sokkal nagyobb a vágási frekvenciánál ($f > fc$), akkor [9]

$$\frac{V_{\text{out}}}{V_{\text{in}}} \cong A_f \tag{9}$$

Itt a jel alacsony frekvenciákon elnyomódik vagy blokkolódik, miközben a kimenet +20dB/-dekád ütemben nő, egészen addig, amíg a frekvencia el nem éri a vágási pontot (fc) (18. ábra). Ennek a szűrőnek a frekvenciamenet görbéje azt sugallja, hogy a szűrő képes minden jelet továbbítani a végtelenig. Azonban a gyakorlatban a szűrő válaszfüggvénye nem terjed a végtelenig, hanem az alkalmazott alkatrészek elektromos jellemzői által korlátozott[9].

18. ábra. Felüláteresztő szűrő frekvenciamenetválasza

2.4.4. Analóg-digitális átalakítás

Az analóg-digitális átalakítás folyamata során az analóg jeleket, mint például a hőmérséklet, sebesség, erő vagy EMG szenzorok feszültségét digitális formává alakítjuk át, hogy azokat számítógépek és digitális rendszerek továbbíthassák, feldolgozhassák, tárolhassák és megjeleníthessék. Ezt az átalakítást egy olyan eszköz segítségével végezzük, amelyet analóg-digitális átalakítónak (ADC) nevezünk.

Számos típusú analóg-digitális átalakító létezik, mindegyik más és más módszert alkalmaz a folytonos analóg jelek diszkrét digitális megfelelővé alakítására. Azonban mindegyiknek vannak közös jellemzői és korlátai, mégpedig:

Mintavételi ráta: A mintavételi ráta, más néven mintavételi frekvencia, az új diszkrét értékek gyakorisága, amelyek a folytonos jelhez tartozó diszkrét értékek létrehozására szolgálnak. A magasabb mintavételi ráták több mintát jelentenek egységnyi idő alatt, ami lehetővé teszi a

[9] https://electronicspost.com/explain-various-types-of-high-pass-filters/

gyors változások jobb rögzítését a jelben. A Nyquist-Shannon mintavételi tétel szerint a mintavételezési frekvenciának legalább kétszerese kell legyen a jelenlegi legmagasabb frekvenciájú jelnek, hogy elkerüljük az aliasingot.

Aliasing: Az aliasing akkor fordul elő, amikor két különböző folytonos jel nem különböztethető meg egymástól diszkrét formájukban. Ez az információvesztést és jelentős problémákat okozhat a rendszerben. Az aliasing elkerülése érdekében antialiasing szűrőket használnak, amelyek blokkolják a jel magasfrekvenciás összetevőit az ADC átalakítás előtt.

Felbontás: Az ADC felbontása azon diszkrét értékek száma, amelyeket az analóg bemeneti értéktartományon belül képes létrehozni. A magas felbontás, általában bitben kifejezve, pontosabb méréseket eredményez. Egy n bites ADC átalakító esetén 2^n különböző diszkrét szint lesz az analóg jel reprezentálására, és ennek köszönhetően egy adott bemeneti jel esetén a több bit magasabb pontosságot eredményez. Például egy 8-bites ADC átalakító 256 különböző diszkrét szintet tud előállítani, míg egy 12-bites átalakító 4096 szintet. [1]

2.5. Képfeldolgozás

Az OpenCV szerint a kéz tájékozódási pont modell általában 21 kulcspontot azonosít és biztosít a kéz különböző pontjainak reprezentációjához. Ezek a pontok magukban foglalják az ujjak végpontjait, csuklópontjait és a kéz egyéb jellegzetességeit. Összesen 21 tájékozódási pont van, beleértve a csuklópontot is ami látható a 19. ábrán is. Ezek a pontok információkat nyújtanak a kéz tartásáról és konfigurációjáról, lehetővé téve különféle kézzel kapcsolatos alkalmazásokat, mint például a gesztusfelismerés, az ujjak követése.

0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

19. ábra. OpenCV kéz tájékozódási pontjai [10]

Minden tájékozódási pont a (x, y, z) koordináták képviselnek, általában normalizálva a kép dimenzióihoz. Ezeket a koordinátákat programozás útján lehet elérni annak érdekében, hogy kézpozícióbecslést, ujjak számlálását vagy kézgesztus felismerést végezzünk. Az OpenCV Hand Landmarkok előre tanított mélytanulási modelleket használnak a kéz és az ujjak hatékony

[10] https://developers.google.com/mediapipe/solutions/vision/hand_landmarker

azonosítására. A modellt körülbelül 30 ezer valós világbeli képre tanították, valamint több renderelt szintetikus kézmodellre, amelyeket különféle háttérrel helyeztek el [10]. Ebben van egy kéz-tájékozódási pontokat tartalmazó modellcsomag, amely tartalmaz egy tenyérérzékelési modellt és egy kézlandmarkok észlelési modellt. A tenyérérzékelési modell megtalálja a kéz területeket a bemeneti képen, míg a kézlandmarkok észlelési modell az észlelt kézterületen azonosítja az illető tájékozódási pontokat.

Mivel a tenyérérzékelési modell futtatása időigényes, a Hand Landmarker a videó- vagy élő adatfolyam futtatási módjában a kézlandmark modell által meghatározott határoló dobozt alkalmazza a kézterületek lokalizálására a következő képkockákban. A Hand Landmarker csak akkor indítja újra a tenyérérzékelési modellt, ha a kézlandmark modell már nem azonosítja a kéz jelenlétét, nincs a kamera képben, vagy nem sikerül a kéz követése a képkockákban. Ennek következtében csökken a Hand Landmarker által kiváltott tenyérérzékelési modell futtatásainak száma [10].

3. A rendszer specifikációi és architektúrája

3.1. Funkcionális követelmények

Kézmozgás érzékelése: A rendszernek képesnek kell lennie az emberi kézmozgások érzékelésére és követésére a megfelelő szenzorok segítségével.

EMG jelek rögzítése és feldolgozása: A rendszernek képesnek kell lennie az EMG jelek rögzítésére és feldolgozására, hogy meg lehessen határozni az izomaktivitást értelmezhesse és az alapján irányíthassa a robotkart.

Kézmozgás alapú vezérlés: A rendszernek lehetővé kell tennie a felhasználó számára, hogy vezérelje és irányítsa a rendszer által vezérelt robotkart a kézmozgások alapján.

Kamera alapú kézmozgás követése: A rendszernek lehetőséget kell biztosítania a kézmozgások követésére és azok alapján történő irányításra a kamera képe alapján.

Váltás a vezérlési módok között: A rendszernek lehetőséget kell biztosítania a felhasználónak, hogy válasszon az EMG alapú és a képfeldolgozáson alapuló vezérlési módok között, és jelezze, hogy melyik módban működik éppen.

Pontosság és megbízhatóság: A rendszernek pontosan és megbízhatóan kell érzékelnie és követnie a kézmozgásokat különböző környezeti feltételek mellett is.

Egyszerre csak egy vezérlési mód aktív: A rendszernek garantálnia kell, hogy egyszerre csak egy vezérlési mód legyen aktív, és automatikusan vagy manuálisan váltson a két módszer között.

Felhasználói felület a módváltáshoz: A rendszernek biztosítania kell egy felhasználói felületet vagy kezelőpanelt, ahol a felhasználó könnyedén kiválaszthatja és módosíthatja a vezérlési módot.

3.2. Nem funkcionális követelmények

A rendszernek pontosan és megbízhatóan kell érzékelnie és követnie az EMG jeleket, valamint azonosítania és követnie kell a kézmozgásokat a kamera segítségével, hogy biztosítsa a helyes és következetes irányítást.

A rendszernek képesnek kell lennie a különböző felhasználói környezetekben és körülmények között történő működésre, például változó fényviszonyok mellett vagy zajos környezetben.

A rendszernek képesnek kell lennie a későbbi bővítésekre és fejlesztésekre, hogy a jövőbeni igényekhez és követelményekhez alkalmazkodni tudjon.

3.3. A rendszer architektúrája

20. ábra. A rendszer architektúrája

A kézmozgás követéséhez két módszert fogunk alkalmazni: az EMG szenzorok és a kamera képfelismerését. Az EMG szenzorok segítségével az izomaktivitást érzékeljük és követjük, míg a kamera kép alapján a kéz fizikai pozícióját és mozgását figyeljük. Ezeket a módszereket egy logika alapján fogjuk váltani annak megfelelően, hogy melyik mód legyen éppen aktív.

3.3.1. Model kiválasztása

A model megtervezését nem tartottam különösebben a feladatomnak, ezért is kerestem nyilt forrású 3d modeleket amelyeket bárki elérhet, viszont rengeteg különböző modelt találtam amiből csak egyet választhattam ki.

A K-1 kéz (21 . ábra) egy antropomorf kéz (emberi alakja van), amely alapvető anyagokat és 3D nyomtatást használ, nincs fém hardver, és elektronika. Ez a kéz nem volt alkalmas mivel mechanikailag működött, és csak az ujjak hiányát kompenzálta.

A K-2 kéz (22. ábra) már alkalmasabbnak tűnt az előzöhöz képest, mivel itt már egy teljes kezet tudunk a modelel helyettesiteni. A mechanikus részt is felváltja egy hardveres rész

[11] https://openscholarship.wustl.edu/cgi/viewcontent.cgi?article=109&context=mems411
[12] https://www.instructables.com/Robotic-Arm-3D-Printed-DIY-Initial-Prosthetic-Prot/

21. ábra. K1 kéz [11]

22. ábra. K2 kéz [12]

ami motorokat hajt meg, ezek már képesek vezérlni az ujjakat, viszont mivel ez egy prototipús volt a dokumentációban is megjelent hogy akadhatnak hibák és ugyan ebben a dokumentációban utalást kaptam egy oldalra ez az InMoov volt ahonnan ő is kiindulást kapott, itt hasonló kézprotézis modelt találtam, és itt sokkal részletesebben voltak kidolgozások.

Gael Langevin egy francia szobrász és tervező, aki több mint 25 éve dolgozik a legnagyobb márkáknak. Az InMoov az ő személyes projektje, 2012 januárjában indították útjára, mint az első nyílt forráskódú protézis kéz, olyan projektekhez vezetett, mint a Bionico, az E-Nable vagy az inMoov robot (23. ábra). Az InMoov egy 3D nyomtatott életnagyságu robot amelynek nyilt forráskodja van, ezáltal megteremtett egy olyan közösséget ahol mindenki képes megépiteni magának a saját robotját [13].

23. ábra. Gael Langevin és az InMoov robot [13]

Itt már egy olyan részletesen kidolgozott 3D modelt találtam amit már könnyedén fel tudok használni és lépésenként beleépiteni a hardver részt (24. ábra). A forumon lépésről lépésre megtalálható a model nyomtatása után az összerakás is. Mivel nyilt forráskódu, ezért a model részeit leölthettem és egy modellező szoftverben vizualizálni tudtam. Ezt a változatot azért találtam előnyösebbnek, mivel itt több dokumentáció volt és meg lehetett nézni hogy mások milyen problémákra milyen megoldásokat találtak, ezért ugy döntöttem hogy ennél a változatnál maradok.

Ugyancsak ehhez a modelhez tartozik egy olyan kezdő szett ami segit a végső projekt motorjainak a tesztelésében, ennek a modelnek a lényege hogy megnézzük hogy elegendő erős-e a motor egy uj mozgatására. Hogyha az ujnak a súját a motor megbirja mozditani többször is probléma nélkül akkor biztosak lehetünk benne, hogy a teljes modelben sem lesz gondunk az ujjak mozgatásában.

Újj kezdőszett [13]

[13] https://inmoov.fr

24. ábra. K3 kéz [13]

3.3.2. Motorok kiválasztása

Különböző szervomotorok közül választottam ki az alábbiakat: HK15298B, Hitec HS805BB, MG996 és MG90S.

A HK15298B jellemzői között megtalálható a megbízható működés, a jó nyomaték-súly arány és a sokoldalú alkalmazhatóság. Ez a szervomotor minősége középkategória felett helyezkedik el, és akár 7,4V-os tápfeszültségen is üzemeltethető, 11,5 kgf*cm nyomatékot kifejtve. Ugyanakkor árban nem számít a leggazdaságosabbnak.

Az HS805BB is egy minőségi szervomotor, ami 6V-on akár 24,7 kgf*cm nyomatékot is tud generálni, azonban ezt sem találtam gazdaságos választásnak. Maradt még a MG996 és a MG90S. Az első 6V-on 11 kgf*cm nyomatékot képes generálni, míg az MG90S 2,2 kgf*cm nyomatékot ad le ugyancsak 6V-on. Végül az MG90S mellett döntöttünk, mivel már más projektek során mások használták az egyetem körében, és költséghatékonyan cserélhető. Ez a motor kompakt méretéhez képest erős és megbízható működést nyújt.

3.3.3. Motor

A szervomotorokat egy adafruit motor vezérlő segítségével hajtjuk meg, mivel ez a megoldás több előnnyel jár a közvetlen mikrovezérlőre csatlakoztatáshoz képest. Az adafruit motor vezérlő biztosítja a megfelelő áramellátást és védelmet a motoroknak, így növelve a rendszer teljesítményét és tartósságát. Emellett egyszerűbb interfészt kínál a motorok vezérléséhez, ami kevésbé bonyolult, mint a mikrovezérlővel való közvetlen kommunikáció. Egyszerre több motor kezelése is lehetséges a driverrel. Emellett beépített védelmi mechanizmusokkal rendel-

kezik, amelyek megvédik a motorokat túláram, túlmelegedés vagy más káros hatások ellen. Ezáltal biztonságosabbá teszi a rendszert és csökkenti a károsodás kockázatát.

A választott driver az Adafruit 16-Channel 12-bit PWM/Servo Driver amely egy hatékony eszköz a PWM jelek és szervomotorok vezérlésére. Az I2C interfésznek köszönhetően könnyedén kommunikál más eszközökkel, ami nagy rugalmasságot és könnyű integrációt biztosít a rendszerünkben. Ez a driver lehetővé teszi akár 16 csatorna független vezérlését, minden csatornán 12-bites pontossággal. A beépített PCA9685 chip hatékony és megbízható működést biztosít, támogatva az automatikus beállítást és a beépített PLL funkciót, ami precíziós időzítést biztosít a PWM jelek számára.

3.3.4. Mikrovezérlő

A mikrovezérlőként az ESP32-WROOM-ot választottam, mivel ennek egy 2 magos processzora van, amely lehetővé teszi a többszálú futtatást és optimalizált teljesítményt. Emellett rendelkezik beépített WiFi modullal is, amely széles körű kapcsolódási lehetőségeket kínál, és lehetővé teszi a vezeték nélküli kommunikációt más eszközökkel és hálózatokkal.

3.3.5. EMG szenzorok

A választott EMG szenzorok géles szenzorok, ezek általábban a leggyakrabban használt szenzorok, így most is ezt fogjuk használni. A géles szenzorok könnyen és kényelmesen illeszkednek a bőrhöz, és stabil érzékelést biztosítanak az izommozgás során. Emellett általában könnyen tisztíthatók és újrahasználhatók, ami praktikus lehet a hosszú távú alkalmazásokban.

4. Részletes tervezés

4.1. Tömbvázlat

Rendszerünk két módban fog müködni, az egyik lesz a jelfelfolgozó része, ahol a karunkból rögzitett jeleket fogjuk feldolgozni.Itt a jelet a jelfeldolgozó lap feldolgozza, amit a mikrovezérlő kiolvas. Ezután a mikrovezérlő megfelelően feldolgozza az adatot, majd eldönti, melyik motort mozgassa a robot kézben. Valamint a másik módszer a képfeldolgozó rész, ahol a kéz mozdulatait vételezzük a kamerával valós időben majd ezt a számitógép segitségével feldolgozzuk és megmondjuk a mikrovezérlőnek, hogy melyik motort mozgassa a robot kezen. Ezt szemlélteni a (25. ábra). A projekt során az ESP32-WROOM mikrovezérlőt használjuk, amely rendelkezik beépített Wi-Fi modullal. Ennek segítségével a mikrovezérlő képes lesz kommunikálni a számítógéppel.

A jelfeldolgozó rész folyamatábrája a 26. ábrán látható. Itt elsősorban a nyers jel végigmegy egy feldolgozási folyamaton, erről részletesebben fogunk beszélni a következőkben. Mi-

25. ábra. A rendszer egy tömbként

után a jel fel van dolgozva, kiolvassuk a jelfeldolgozó eszközből, és tüskék (spike) detektálását végezzük. Ha a detektálás során a jelben hirtelen változások történnek, azokat küszöbértékkel hasonlítjuk össze. Amennyiben a változások meghaladják a küszöbértéket, ennek függvényében mozgatjuk a szervomotorokat, és szükség esetén várunk egy rövid időt, mielőtt visszaállítjuk őket az eredeti állapotba.

26. ábra. A jelfeldolgozó rész folyamat ábrája

A képfeldolgozás folyamatábrája a 27. ábrán látható. Először inicializálunk egy Wi-Fi elérési pontot az ESP32-vel és egy kamerát, majd elindítjuk a Wi-Fi szervert. Miután a szerver működésbe lépett, egy eszközzel, jelen esetben egy számítógéppel, csatlakozunk hozzá. Ezután a kamera felvételéből képkockákat (frame-eket) vágunk ki, amelyeket feldolgozunk. A feldolgozott frame-eken a MediaPipe segítségével felismerjük a kezet, és tájékozódási pontokat helyezünk el. Ezek alapján meghatározzuk az ujjak állapotát. Az ujjak állapotáról kapott adatokat Wi-Fi-n keresztül továbbítjuk az ESP32-re, amely a kapott értékek függvényében mozgatja a szervomotorokat.

27. ábra. A képfeldolgozó rész folyamat ábrája

4.2. Hardver rész

4.2.1. Modell előkészitése

Mivel eltérő motorokat használunk, mint amelyekre eredetileg tervezték a 3D modellt, ezért néhány alkatrész módosítása szükségessé vált. Az új alkatrészek tervezését az Autodesk Fusion tervezőprogramban végeztem. Az alkatrészek amelyeket újra kellett tervezzek azok a motor ágy és a motor tengelyére illeszkedő kerék volt amelyek a kábelek vezetését biztosítják ezek láthatóak a 28. ábrán.

28. ábra. Motor ágy és motor kerekek 3D modelje

A motor ágy modelljét úgy kellett megtervezni, hogy a motorok elférjenek benne, és a rájuk szerelt kerekek ne ütközzenek, miközben figyelembe vettük, hogy korlátozott hely áll rendelkezésre a kar belső terében. Emellett fontos szempont volt a motorok táplálási kábeljének elhelyezése is, melyeket a modell alján vezettem ki.

4.2.2. Motorok tesztelése

Annak érdekében, hogy megbizonyosodjam arról, hogy a motorok elegendő erőt generálnak és biztonságosan elérnek a gyártó által meghatározott értékeket, különféle teszteket végeztem rajtuk. Ezek a tesztek lényegében fokozatosan terhelték a motorokat különböző súlyokkal, míg el nem érték a gyártó által előírt értékeket. A súlyokat egyforma tömegű csavarok alkották (12.6 gramm), melyek pontos súlymérését precíz mérleggel végeztük. A "Teszt 0" az állapotot jelöli a horog és a zacskó súlya nélkül, vagyis a csavar/súly nélküli állapotot. A narancssárgával jelölt nyomaték azt jelzik, hogy még a gyártó által meghatározott nyomaték alatt vagyunk, a barnával jelölt nyomaték pedig azt, hogy közel vagyunk a meghatározott nyomatékhoz, míg az piros színnel jelölt nyomaték azt mutatják, hogy már túlléptük azt. A diagram alapján (lásd 1. táblázat) látható, hogy a "Teszt szám" oszlopban a motorok még 16 darab csavarral is működtek, ami kb. 204 g tömegnek felel meg. Ez jóval meghaladja az adatlapokon szereplő értéket, mely 2,2 kg/cm, így arra következtethetünk, hogy a szervomotorok megfelelő teljesítményt nyújtanak.

1. táblázat. Motor Teszt

Teszt sorszám	Súly(gram)	Kar hossza (cm)	Nyomaték(kg/cm)
0	3	1.3	0.039
1	15.6	1.3	0.202
2	28.2	1.3	0.366
3	40.8	1.3	0.530
4	53.4	1.3	0.694
5	66	1.3	0.858
6	78.6	1.3	1.021
7	91.2	1.3	1.185
8	103.8	1.3	1.349
9	116.4	1.3	1.513
10	129	1.3	1.677
11	141.6	1.3	1.840
12	154.2	1.3	2.0
13	166.8	1.3	2.168
14	179.4	1.3	2.332
15	192	1.3	2.496
16	204.6	1.3	2.659

4.2.3. Jel feldolgozása

Már léteznek olyan eszközök, amelyek képesek a feldolgozási feladatok elvégzésére, például a Muscle Sensor V3, amely egy olyan Arduino-hoz készült jelfeldolgozó eszköz, amelyet az Arduino analóg digitális átalakítóján keresztül olvashatunk be. Ezt szemlélteti a 29. ábra. Bár ez egy könnyen használható modul, általános célú, és én inkább egy olyan eszközt kívántam létrehozni, amely képes specifikus paraméterekkel rendelkező jeleket feldolgozni és amely pontosan megfelel a projekt specifikus követelményeinek.

29. ábra. Izom szenzor modul [14]

[14] https://www.sparkfun.com/products/retired/11776

A rendszerünk fő feladata az emberi karban lévő jelek érzékelése és azok feldolgozása, melyeket bemenetként kezelünk. A jelek feldolgozása után az eredményeket továbbítjuk a számítógép felé, hogy azok hasznos információkként szolgáljanak számára. Az ábrán egyetlen nagyobb blokk látható, amely az egész rendszert ábrázolja (30. ábra). Itt az X a bemenetet jelenti, ahol a jelek belépnek a rendszerbe, míg az Y a kimenetet, ahol a számítógép számára szükséges értékek kerülnek kimenetként. Ez a blokk magában foglalja az összes részletet és funkciót, amelyek együttműködve biztosítják a rendszer megfelelő működését.

30. ábra. A jelfeldolgozó rendszer egy tömbként

Az alábbiakban áttekintjük a tömbvázlatot, amely átfogóan ábrázolja a jelfeldolgozó rendszerünket (31. ábra). Minden egyes blokk egy modult jelöl, amelynek saját feladata van. A tömbvázlat segítségével könnyebben megérthetjük a rendszer felépítését és működését. Az egyes modulok összekapcsolása és együttműködése kulcsfontosságú ahhoz, hogy hatékonyan feldolgozzuk a bemeneti jeleket és elérjük a kívánt eredményeket.

31. ábra. Átfogó tervezési tömbvázlat

4.2.4. EMG elektródák

A felületi EMG elektróda az izom elektromos aktivitását vagy az ionáramot érzékeli, és azt elektronikus jelekké alakítja. Fő célja az EMG jel pontos rögzítése, miközben minimalizálja az eltéréseket a DC potenciálokat és a környezeti zajt. Az elektróda minősége több tényezőtől függ, mint például az alakja, mérete, anyaga stb., de minden elektróda azonos jellemzőket oszt meg.

A rendszertervezés első lépése az, hogy teljes mértékben megértsük azokat a jeleket, amelyeket a rendszer feldolgozásra fog kapni. Az (32. ábra) az EKG jelet mutatja be, ahogy az megjelenik az EKG mérési rendszer bemenetén. Ez a jel három fő összetevőből áll: az aktuális (differenciális) EKG jelből, a differenciális elektromos offsetből és a közös módú jelből. Az aktuális differenciális EKG jel a bőrön lévő elektródák közötti elektronikus szigetelő rétegben megjelenik, és általában ±5 mV-es magnitúdójú és 0,05 Hz-től 150 Hz-ig terjedő frekvenciájú.

Ennek a jelnek a magnitúdója és a rendszer által igényelt felbontás meghatározza az elsődleges tartomány igényét, míg a jel frekvenciája az analóg előtét sávszélességét határozza meg. A bőr-elektromos elektródok interfésze további kb. 300 mV-os DC-offsetet ad. Ennek az offsetnek a kezelése során két mód lehetséges, a rendszerben használt ADC típusától függően: vagy teljesen eliminájuk, vagy megőrizzük [3].

32. ábra. Az EKG-jel jellemzői [3]

4.2.5. ESD védelem

Az elektrosztatikus kisülés (ESD) az elektromosság hirtelen kibocsátása egy feltöltött tárgyból egy másikba, amikor a két tárgy érintkezik egymással. Az érzékeny integrált áramkörök (IC-k) számára az ESD lövések magas csúcspontú feszültsége és árama meghibásodásokat okozhatnak. Ha az ESD védelem nincs jelen egy rendszerben, az ESD lövés magas feszültsége egy interfész csatlakozón keresztül nagy áramimpulzust indít meg közvetlenül az IC-be, ami károsodást okozhat. Az érzékeny áramkörök védelme érdekében az elektrosztatikus kisülés elleni védelmi diódák minden jelvezetéket összekapcsolnak az interfész csatlakozó és az IC között, ez a megvalósitás (50. ábra) látható 1-el jelölve. Egy ESD lövés esetén az ESD dióda elromolhat és alacsony impedanciájú utat hoz létre, amely korlátozza a csúcspontú feszültséget és áramot a földre terelő áram útjával, ezzel megvédve az IC-t. Az alábbi ábra (33. ábra) összehasonlítja az ESD lövés csúcspontú feszültségét védelem nélkül (kék) az ESD diódával védett jelvezetékű ESD lövéshez (piros) [15].

[15] https://www.ti.com/lit/sg/sszb130d/sszb130d.pdf?ts=1711266785142

33. ábra. Feszültséghullámformák ESD védelemmel és anélkül [15]

4.2.6. DC elutasitás

Egy teljesen differenciális passzív AC-kapcsoló hálózat egy olyan áramkör, amelyet a jelek differenciális formában történő továbbítására használnak, miközben blokkolják a DC összetevőket. Ez általában két felül áteresztő szűrőből áll, amelyek tükörképes elrendezésben vannak egymásra fordítva.

Fontos hogy a jelek differenciális formában továbbítódjanak, vagyis az azonos potenciálokra ható zajokat vagy interferenciákat el kell távolítani. Ennek megvalósításához használhatunk például két egyenáramú (DC) elutasításra képes kondenzátort és egy közös módú komponenstől független feszültségadót[5]. Ezáltal a rendszer tisztább és pontosabb jeleket továbbít (34. ábra) valamint (50. ábra) 2-el jelölve.

Kondenzátorok (C és C'): A kondenzátorok arra szolgálnak, hogy blokkolják az egyenáramú (DC) jeleket, így csak az általunk kívánt váltakozó áramú (AC) jelek juthatnak át az áramkörön. A kondenzátorok tárolják az elektromos energiát elektromos térben, és az AC jelek áthaladására engedélyezik a kapcsolatot.

Ellenállások (R1 és R2): Az ellenállásoknak szerepük van az áram áramlásának korlátozásában és a kondenzátorok töltésének szabályozásában. Ezek segítenek megakadályozni a túl nagy áramok kialakulását és védenek az áramkör túlzott terhelésétől. Ugyan ezt a szerepet játsza R1' és R2' is.

34. ábra. Passzív AC-kapcsoló hálózat [5]

4.2.7. Erősités

Mivel a felületi EMG jel nagyon gyenge és különféle zajok megzavarhatják a jelet, ezért a használt insztrumentális-erősítőnek magas közös módú elnyomási aránnyal kell rendelkeznie annak érdekében, hogy hatékonyan csökkentse a közös módú interferenciát, és hatékony differenciális jelet szerezzen [1]. Az EMG jel minősége részben az erősítési folyamat jellemzőitől függ, a legfontosabb szakasz gyakran az előerősítés lehet ez látható a (50. ábra) 3-al jelölve. Az előerősítés azt jelenti, hogy az erősítés első szakaszát, közvetlenül a jelforráshoz közel végezzük el. Ebben a tervezésben az INA826 erősítőt használjuk a Texas Instruments-től, ez egy nagy pontosságú, alacsony zajszintű, differenciális bemenetű insztrumentális erősítő. Továbba alkalmazunk egy aktiv műveleti erősitőt integrátoral amit vissza vezetünk az INA826-ba, ez segit hogy a jelünk simább legyen ezt (50. ábra) 5-el jelölve láthatjuk. A (35. ábra) alapján az erősités a következő képpen fog alakulni:

$$\text{Erosités} = 1 + \frac{49.4k\Omega}{R_g} \tag{10}$$

Ahol:

- R_g - az erősités ellenállása

A mi esetünkben 500 ohmos ellenállásunk van, tehát az erősitésünk körülbelül 100-szoros. Továbbá:

- Közös módú elutasítás: 100 dB

- Tápfeszültség: 3 V - 36 V
- Bemeneti impedancia: 100 kΩ [6]

A műveleti erősítő képlete alapján észre vehetjük, hogy az ina826 erősitő erősítése a következő képlet alapján is számolható:

$$A = \frac{V\text{out}}{V_2 - V_1} = (1 + \frac{2R1}{Rg})\frac{R3}{R2} \tag{11}$$

Ahol:

$$R2 = R3 = 50\,k\Omega \tag{12}$$

$$R1 = 24.7\,k\Omega \tag{13}$$

Ebből adódoan megkapjuk a [10] képletet:

$$A = 1 + \frac{49.4k\Omega}{Rg} \tag{14}$$

35. ábra. Ina826 erősitő áramköri rajza [6]

4.2.8. DRL és Shield

A driven-right-leg (DRL) áramkör segít csökkenteni az áramkörön belüli impedanciát, ami lehetővé teszi a közös módú feszültség csökkentését és minimalizálja a lehetséges impedanciaarány-eltérést. A DRL áramkör visszavezeti a közös módú feszültséget a rögzítő elektródákba, és segít elkerülni az impedanciaarány-eltérést, míg a Shield áramkörök különböző technikákkal csökkentik az elektromágneses interferenciát a vezetékekben[1]. Emellett a DRL áramkör további védelmet nyújt a páciensek számára, mivel csökkenti a közös módú feszültséget és minimalizálja a lehetséges interferenciát az EMG mérések során, ez látható a (50. ábra) 4-al jelölve. Itt hasznos használni egy buffert is, hogy a közös módú feszültséget terhelés nélkül vegyük le.

4.2.9. 50 Hz-es hálózati zajszűrő

Az 50-60 Hz-es jelek általában a hálózati frekvenciájú zajok, amelyek az elektromos hálózatból származnak. Az EMG mérések során az ilyen frekvenciájú zajok befolyásolhatják az eredményeket, ezért ezeket ki kell szűrjük hogy tisztább jeleket kapjunk. Erre az egyik alkamazási módszer az úgy nevezett "twin-T notch filter" vagy sávkivágó szűrő (36. ábra).

36. ábra. 50hz kivágó áramkör

A vágási frekvencia a következő képpen fog kinézni:

$$f = \frac{1}{2\pi RC} \text{ Hz} \tag{15}$$

Vagyis:

$$f = \frac{1}{2\pi \cdot 145k\Omega \cdot 22nF} \text{ Hz} = 49.91 \text{ Hz} \tag{16}$$

Ahol:

$$R = R_1 = R_2 = \frac{R_3}{2} \tag{17}$$

$$C = C_1 = C_2 = C_3 \cdot 2 \tag{18}$$

A 37 ábrán látható, hogy az 50 Hz-es szűrő szimulációja hatékonyan szűri az 50 Hz-es frekvenciát. Már körülbelül 48 Hz-nél kezd hatékonyan szűrni. A szűrő átviteli görbéje szimmetrikus az 50 Hz-es frekvencia körül.

37. ábra. 50hz kivágó áramkör szimuláció

4.2.10. 10Hz-1kHz sávszűrő

Az aktiv sávszűrők egy magas- és egy alacsonyfrekvenciás aktiv áteresztő szűrő kombinációjából állnak, ezeket aszűrőket nevezik még Sallen-Key szűrőnek is nevezik. A_V mindig nagyobb lesz, mint 1 de 3-nál kisebbnek kell lennie, különben instabillá válik [16]. A magasfrekvenciás áteresztő szűrő kondenzátorai engedik át a magas frekvenciájú jeleket, míg az alacsonyfrekvenciás áteresztő szűrő ellenállásai alacsonyabb frekvenciájú jeleket engednek át, míg a magasabb frekvenciájú jeleket blokkolják vagy levezetik a földre. Itt azonos kondenzátor- és ellenállásértéket kell használni (38. ábra). A jel megfelelő szűrésének biztosítása érdekében kétszer alkalmazzuk az alul- és felül-áteresztő szűrőket, és az ADC számára helyes mintavételezett értékek átadásához szükséges, hogy a jelet egy referenciafeszültséggel emeljük meg. Ellenkező esetben a negatív jelünk levágásra kerül. Ez a művelet az U3-as erősítő előtt történik. Ez a rész látható a (51. ábra) 7-el jelölve.

A frekvenciasáv felső tartománya f_H a következőképpen fog alakulni:

[15] https://www.electronics-tutorials.ws/filter/band-stop-filter.html
[16] https://www.electronics-tutorials.ws/filter/sallen-key-filter.html

38. ábra. Szűrő és másodlagos erősítő áramkör

$$f_H = \frac{1}{2\pi\sqrt{R_4R_5C_3C_4}}\ \text{Hz} = \frac{1}{2\pi\sqrt{R_{10}R_{11}C_7C_8}}\ \text{Hz}^{16} \tag{19}$$

Vagyis:

$$f_H = \frac{1}{2\pi\sqrt{3.9k\Omega \cdot 3.9k\Omega \cdot 47nF \cdot 47nF}}\ \text{Hz} = 868.275\ \text{Hz} \tag{20}$$

A frekvenciasáv alsó tartománya f_L a következőképpen fog alakulni:

$$f_L = \frac{1}{2\pi\sqrt{R_1R_3C_1C_2}}\ \text{Hz} = \frac{1}{2\pi\sqrt{R_6\frac{R_9R_8}{R_9+R_8}C_1C_2}}\ \text{Hz} \tag{21}$$

Vagyis:

$$f_L = \frac{1}{2\pi\sqrt{5.6k\Omega \cdot 12k\Omega \cdot 2.2uF \cdot 2.2uF}}\ \text{Hz} = 8.824\ \text{Hz} \tag{22}$$

Ahol:

- f_H - a maximális átengedett frekvencia
- f_L - a minimális átengedett frekvencia

Továbbá:

$$C_1 = C_2 = C_5 = C_6 \tag{23}$$

$$C_3 = C_4 = C_7 = C_8 \tag{24}$$

$$R_1 = R_6 = \frac{R_3}{2} = \frac{R_9 R_8}{R_9 + R_8} \tag{25}$$

$$R_4 = R_5 = R_{10} = R_{10} \tag{26}$$

A szűrő alsó vágási frekvenciája körülbelül 10 Hz-nél helyezkedik el, pontosabban 8.8 Hz-nél, amint azt kiszámoltuk. Ugyancsak, a felső vágási frekvencia 1 kHz alatt, méghozzá 868 Hz-nél. A két frekvencia között látható, hogy a frekvenciasáv 20 dB-en van, azaz ezeket nem szűri ki, míg azokat, amelyeket a szűrő már érint, a csökkenés jelez. Így azokat már nem engedi át ez látható a 39. ábrán is.

39. ábra. Sallen key

A 40 ábrán láthatjuk az utolsó felüláteresztő szűrőt a referencia feszültség hozzáadása előtt és a referencia feszültség hozzáadása után, itt szükséges volt 2 darab 24k ellenállás hozzáadása párhuzamosan, hogy maradjon meg a 12k-os ellenállás érték.

40. ábra. 50hz kivágó áramkör szimuláció

Az 41. ábrán jól látható, hogy miután bevezettük a referenciafeszültséget, az áramkör feszültsége +ref értékkel emelkedett meg. A zöld színű szinusz a referenciafeszültség hozzáadása előtti feszültséget mutatja, míg a piros a referenciafeszültség hozzáadása utáni állapotot ábrázolja. Így az ADC konverter helyesen fogja kiolvasni a feszültségértékeket, és nem fog levágni, mint korábban, amikor a feszültség értéke 0 alatt volt.

41. ábra. Referencia hozzáadása a jelhez

4.2.11. Analóg-digitál átalakító

Az MCP3201 Analóg-digitál átalakító 12-bites felbontása lehetővé teszi az analóg bemenetről származó feszültség szintjének 4096 különböző digitális értékkel való reprezentálását ez az alkatrész látható a (51. ábra) 8-al jelölve.

FIGURE 6-1: *SPI Communication using 8-bit segments (Mode 0,0: SCLK idles low).*

FIGURE 6-2: *SPI Communication using 8-bit segments (Mode 1,1: SCLK idles high).*

42. ábra. ADC használata mikrokontrolerrel [7]

Az ADC-ből történő adatkiolvasáshoz két módszert alkalmazhatunk. Az első módszerben először a CS (Chip Select) lábat nullára kell állítani, majd 16 órajel-impulzus után a Dout (Data Out) lábon megkapjuk az adatot. Az adat első két órajelére a HI-Z (High Impedance) érték kerül, majd az első bitet figyelmen kívül hagyjuk, mivel ez mindig NULL BIT. Az utolsó két bitet sem vesszük figyelembe, mivel ezek a LSB (Least Significant Bit) értékei. A második módszer hasonló, de ott csak egy LSB-t kell figyelembe venni, mivel az utolsó órajel nem tér vissza nullára (42. ábra).

4.2.12. Tápegység áramkör

Az INA826 műszererősítő és az MCP6004 integrált műveleti erősítő kétutas tápegységgel működnek. Az AZ1117H-3.3TREI dióda a bemeneti feszültséget szabályozza és egy stabil 3.3V-os kimeneti feszültséget biztosít. Ezenkívül az AZ1117CH-2.5TRG1 dióda 2.5V-os feszültséget biztosít.

4.3. Firmver

Kezdetben az Arduino UNO-val kezdtük a mikrovezérlő fejlesztését, és először csak a szervomotorokat teszteltük vele. Később az Adafruit motorvezérlőt is beintegráltuk, hogy könnyen tudjuk irányítani az öt szervomotort. Itt a jelentős része az volt hogy tudjuk mozgatni a 3D modell ujjait megfelelő módon a szervomotrokkal, vagyis találjunk egy olyan szöget amelyben biztonságosan elforoghatnak a motrok ahhoz hogy ne huzzák túl az ujjakat. De amikor nekiálltunk a képfeldolgozásnak, rájöttünk, hogy az Arduino UNO nem elég erős a feldolgozáshoz és SPI kommunikáción sem elegáns az adat átküldés.

Ezért úgy döntöttünk, hogy váltunk egy erősebb platformra, az ESP32-re. Ez azért volt jó döntés, mert az ESP32-nek van beépített WiFi-je, ami lehetővé teszi a vezeték nélküli kapcsolatot a számítógéppel. Ráadásul a kétmagos processzora is van, ami sokkal több számítási erőt biztosít.

4.3.1. Loop függvény

A loop függvény az Arduino programok fő ciklusa, amely folyamatosan ismétlődik a program futása során. Ebben a függvényben döntés születik arról, hogy az EMG adatokat kell feldolgozni, vagy WiFi klienssel kell foglalkozni, a kapcsoló állapotától függően.

```
void loop() {
  useEMG = digitalRead(switchPin) == LOW; /

  if (useEMG) {
    handleEMG();
  } else {
    handleWiFi();
  }

  delay(20);
}
```

4.3.2. WiFi Acces Point

Az adatok WiFi-n keresztüli továbbításához az ESP32-nek létre kell hoznia egy WiFi hozzáférési pontot. Ez azt jelenti, hogy az ESP32 készít egy saját WiFi hálózatot, amelyre más eszközök, például egy laptop is csatlakozhatnak. A kódban a const char *ssid és const char *password változók definiálják a létrehozott WiFi hálózat nevét (SSID) és jelszavát. Ezeket a változókat az ESP32 konfigurálása során használjuk a hálózat létrehozásához.

```
const char *ssid = "ESP32-Access-Point";
const char *password = "password123";
WiFiServer server(80);
```

A WiFiServer objektum létrehozása során az ESP32 beállít egy TCP szervert a 80-as porton. Ez a szerver fogadja majd az adatokat a WiFi hálózaton keresztül és továbbítja azokat az ESP32-nek.

A WiFiServer objektum és a hozzá tartozó beállítások a szoftver részét képezik, amely lehetővé teszi az ESP32 számára, hogy fogadhassa az adatokat a WiFi hálózaton keresztül.

Eztkövetően be kell állitani és elinditani az elérést. A kódrészlet arra szolgál, hogy létrehozzon egy WiFi hálózati pontot az ESP32-en, és kiírja annak IP-címét a soros monitorra. A WiFi.softAP(ssid, password) függvény beállítja a hálózati pontot a megadott SSID-val és jelszóval, míg a WiFi.softAPIP() függvény lekéri az IP-címet, amit aztán kiírunk a soros monitorra a Serial.println() segítségével. Majd a server.begin() parancs elinditja a szervert amelyre az adatot lehet küldeni [17].

```
  WiFi.softAP(ssid, password);
  IPAddress IP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(IP);

  server.begin();
```

A következő rész a szerver kódban arra szolgál, hogy kezelje az új kliens kapcsolatokat, amelyek csatlakoznak a szerverhez.

Először létrehozunk egy WiFiClient típusú objektumot server.available() függvénnyel. Ez az objektum reprezentálja a kliens kapcsolatot, amely a szerverhez csatlakozott.

Ezután ellenőrizzük, hogy valóban létrejött-e kapcsolat a klienssel, a if (client) feltétellel. Ha igen, azaz ha van kapcsolat, akkor kiírjuk a soros monitorra, hogy új kliens kapcsolódott.

```
   WiFiClient client = server.available();

   if (client) {
    Serial.println("New client connected");
```

Ezt követően egy while ciklusban figyeljük a kliens kapcsolatot (while (client.connected())). Amíg a kliens kapcsolat aktív és nyitott, addig a ciklus tart. Ha a kliens megszakítja a kapcsolatot, a while ciklusból kilépünk, és a program folytatódik.

[17] https://randomnerdtutorials.com/esp32-access-point-ap-web-server/

```
while (client.connected()) {
    useEMG = digitalRead(switchPin) == LOW;
    if (useEMG) {
      Serial.println("Switch to EMG mode detected, breaking WiFi
   loop");
      break;
    }
```

A kapcsoló állapotát folyamatosan ellenőrzi. Ha a kapcsoló be van kapcsolva (LOW), a program kilép a WiFi kezelés ciklusából, és visszatér a loop függvénybe, hogy az EMG adatokat kezelje.

```
    if (client.available()) {
      request = client.readStringUntil('\r');

    if (request.length() > servoCount) {
      request = request.substring(request.length() - servoCount);
    }

    if (request.length() == servoCount) {
      for (int i = 0; i < servoCount; i++) {
        char servoState = request.charAt(i);
        if (servoState == '1') {
          moveServo(i, 0);
        } else if (servoState == '0') {
          moveServo(i, 120);
        } else {
          Serial.println("Invalid servo state");
        }
      }
    } else {
      Serial.println("Invalid data length");
    }

client.stop();
```

```
Serial.println("Client disconnected");
```

Az if (client.available()) feltétellel ellenőrizzük, hogy érkezett-e adat a kliens felől. Ha igen, akkor olvassuk be az adatot a client.readStringUntil() függvénnyel, ami elvégez egy sor kiolvasását. Ezáltal a szerver képes fogadni az üzeneteket, amelyeket a kliens küld. Eztkövetően a kapott adatokat feldolgozzuk, ellenőrizzük hogy helyes legyen és meghajtjuk a szervomotorokat. Ha a beérkezett adat hossza megegyezik a szervók számával, akkor minden karaktert külön feldolgoz. Ha a karakter '1', a megfelelő szervót 0 fokos pozícióba állítja. Ha a karakter '0', a szervót 120 fokos pozícióba állítja. Végül, ha a kliens lekapcsolódik, a stop() függvény hívódik meg, és kiírja a soros monitorra, hogy a kliens lekapcsolódott.

4.3.3. EMG jel kezelése

A handleEMG() függvény az EMG jelek kezelésére szolgál.

```
void handleEMG() {
  int adcValue = readADC();
  Serial.println(adcValue);

  unsigned long currentTime = millis();

  if (abs(adcValue - lastAdcValue) > spikeThreshold) {
    if (!servosMoved) {
      for (uint8_t i = 0; i < servoCount; i++) {
        moveServo(i, 120);
        delay(20);
      }
      servosMoved = true;
    }
    lastMoveTime = currentTime;
  } else {
    if (servosMoved && currentTime - lastMoveTime >= 1000) {
      for (uint8_t i = 0; i < servoCount; i++) {
        moveServo(i, 0);
        delay(20);
      }
      servosMoved = false;
```

```
    }
  }

  lastAdcValue = adcValue;
}
```

Az adcValue változó az ADC értékét olvassa be, amely az izom aktivitását méri. Ezt követően a jelben lévő tüskéket (spike) detektáljuk, hogy azonosítsuk a hirtelen változásokat. Ha a detektált tüske meghaladja a küszöbértéket, és az előző mérések alapján nem volt tüske (!servosMoved), akkor minden szervót 120 fokos pozícióba állítunk, majd beállítjuk a servosMoved változót igazra itt mind addig marad abban a pozicióban ameddig a tüskék érkeznek. Ha a tüske detektálása során az érték alacsonyabb vagy egyenlő a küszöbértékkel, és a szervók már mozgatva lettek és az utolsó tüskétől számítva már eltelt 1 másodperc, akkor a servosMoved változót hamisra (false) állítjuk, és minden szervót 0 fokos pozícióba állítunk.

4.3.4. ADC olvasás

A jefeldolgozó lapról a jeleket eddig arduinóval olvastuk ki, viszont csak a lábkiosztást szükséges módosítani ahoz, hogy ESP32-vel is tudjuk használni a kódot.

Az `adcRead` függvény a programban az analóg jelek digitális formába történő átalakítására szolgál az ESP32-vel használt külső analóg-digitális átalakító (ADC) modul segítségével.

```
uint16_t readADC() {
  uint16_t result = 0;

  // Start SPI transaction
  SPI.beginTransaction(SPISettings(1000000, MSBFIRST, SPI_MODE0));
  digitalWrite(CS_PIN, LOW); // Select the ADC
```

Először is, a függvény elkezdi az SPI kommunikációt az ADC-vel. Az ESP32-n az ADC olvasása SPI (Serial Peripheral Interface) protokollt használva történik. A beginTransaction függvény beállítja az SPI kommunikációs paramétereket, például a sebességet (1000000 bit/mp), a legjelentősebb bitet először (MSBFIRST), és a módját (SPI_MODE0).

Az CS_PIN digitális tüske alacsony logikai szintre állítása kiválasztja az ADC-t, így a kommunikáció az ADC-vel történik.

```
  // Read 2 bytes from the ADC
  result = SPI.transfer16(0x0000);
```

Az SPI.transfer16(0x0000) függvény segítségével 16 bites adatot olvasunk az ADC-ből. Az ADC által adott értékek két bájtban érkeznek, amelyeket egy uint16_t típusú változóban tárolunk el (result).

```
  digitalWrite(CS_PIN, HIGH); // Deselect the ADC
  SPI.endTransaction();

  return result;
}
```

Miután befejeztük az ADC olvasását, a CS_PIN digitális tüskét magas logikai szintre állítjuk, hogy lezárjuk az ADC-vel való kommunikációt. Az SPI.endTransaction() függvény pedig befejezi az SPI tranzakciót, elengedve az SPI buszt más eszközök számára. Végül visszaadjuk az olvasott ADC értéket, ami a result változóban van tárolva.

4.3.5. Négyzetes átlag számítás

Eddig a nyers jelünkkel dolgoztunk, amelyet először szűrnünk kellett annak érdekében, hogy eltávolítsuk a zajos vagy a jel minőségét rontó komponenseket. A szűrés után beolvastuk az értékeket a feldolgozó eszközünkről, és egy egyenirányított, nullától nagyobb értékeket kaptunk, amelyek a feszültségszint (12-bites) megfelelői. Ezeket a fázisokat szemlélteti a 43. ábra is.

Ezek az értékek már reprezentálják az izomaktivitást, de még mindig nem alkalmasak a robot kéz irányítására, mivel az izomaktivitás jelei hirtelen változhatnak, ami a robot kéz ujjainak gyors és nem kívánt mozgásához vezethet. Szükséges egy olyan szűrési technika alkalmazása, amely segít az izomaktivitás jelének stabilizálásában és a hirtelen változások mérséklésében. Az egyik lehetséges módszer a négyzetes átlag (RMS) számítása, amely a jelek simítására és a gyors ingadozások csökkentésére szolgál, ez a 27. kifejezés szemlélteti.

A négyzetes átlag (RMS) az alábbi lépésekből áll:

- Minden egyes mintát négyzetre emeljük.
- A négyzetre emelt minták átlagát kiszámítjuk.
- Az átlagolt négyzetes értékekből kiszámítjuk a négyzetgyököt, amely megadja a simított jelet.

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \tag{27}$$

Ahol:

- N - a minták száma
- x - a minta értéke

43. ábra. Jelfeldolgozás fázisai [10]

Az RMS szűrő hatékonyan csökkenti a jel zaját és a gyors változásokat, ami hasznos lehet a robot kéz irányításához. Az RMS szűrés során azonban érdemes megjegyezni, hogy bár az oszcillációk és zajok mértéke csökken, a jel még mindig tartalmazhat kisebb ingadozásokat és zajokat. Az RMS szűrő tehát nem biztosít teljesen sima jelet, hanem csökkenti a hirtelen változásokat és a jel energiáját méri, amely a stabilabb jelérték elérésére szolgál.

A négyzetes átlag módszerének bemutatása mellett végül a spike detektálására építettük a megoldást, mivel ez a megközelítés közvetlenebb és egyszerűbb módot nyújtott a robot kéz irányítására. A spike-ok észlelése lehetővé tette, hogy a robot kéz az izomaktivitás gyors változásaira reagáljon, míg az RMS szűrő módszerét alternatívaként vizsgáltuk a jel stabilitásának javítására.

4.3.6. Spike Detektálás

Az RMS szűrés elméleti vizsgálata után végül úgy döntöttünk, hogy a spike detektálására építjük a megoldást. A spike detektálása egy közvetlenebb és egyszerűbb megközelítést kínál az izomaktivitás jeleinek kezelésére, amely nem igényli a jel előfeldolgozását, mint például átlagolást.

A spike-ok az izomaktivitás hirtelen növekedését jelzik, amely közvetlenül alkalmazható a robot kéz ujjainak mozgására. A spike detektálása az aktuális és az előző ADC érték közötti jelentős különbségek alapján történik, amely lehetővé teszi a robot kéz ujjainak reakcióját az izomaktivitás hirtelen változásaira.

A végső megoldásban a következő lépések alapján dolgozunk:

- Az EMG jel ADC értékeit folyamatosan olvassuk.
- Az aktuális ADC érték és az előző ADC érték közötti különbséget ellenőrizzük.
- Ha az eltérés meghaladja a spikeThreshold értéket, spike-ot detektálunk.
- Ha a spike detektálódik, a servomotorok elfordulnak.
- Ha egy másodperc elteltével nincs spike, a servomotorok visszatérnek az alapállapotba.

4.4. Szoftver rész

4.4.1. Kép feldolgozása

A rendszer funkciójához tartozik a képfeldolgozás amely segitségével olyan adatot küldünk a mikrokontrollernek amellyel vezérelni tudjuk a robot kart. A számitógép feldolgozza a kamera által rögzitett videót és ezeket az információkat továbbitja a mikrovezérlőnek, ezután a mikrovezérlő végrehajtja a szükséges műveleteket a kapott adatok alapján.

A projekt során a kamera képeinek beolvasásához a Python nyelvben a cv2.VideoCapture() függvényt használjuk. A beolvasott képek feldolgozását az OpenCV (cv2) és a Mediapipe képfeldolgozó könyvtárral végezzük el. A Mediapipe lehetőséget biztosít a kézi követéshez és az ujjak számának meghatározásához.

A feldolgozott adatokat átalakítjuk olyan formátumba, amelyet a mikrokontroller könnyen értelmezhet. Ezeket az adatokat a mikrokontroller fogadja a számítógéptől, majd feldolgozza és végrehajtja a szükséges műveleteket, vagyis motorok meghajtását a megfelelő módon (44. ábra).

44. ábra. A kép feldolgozásának tömbvázlata

A következő módszer egy Python kódot mutat be amely egy képfeldolgozó alkalmazást ír le.

```
import numpy as np
import mediapipe as mp

mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
```

```
hands = mp_hands.Hands(static_image_mode=False, max_num_hands=1,
    min_detection_confidence=0.5, min_tracking_confidence=0.5)
```

- `import numpy as np`: Ez az import parancs a NumPy könyvtárat hozza be a projektbe. A NumPy a Python egyik alapvető könyvtára, amely hatékony matematikai műveleteket tesz lehetővé többdimenziós tömbökkel és azokkal való műveletekkel.
- `import mediapipe as mp`: Ez az import parancs a Mediapipe könyvtárat hozza be a projektbe. A Mediapipe egy nyílt forráskódú keretrendszer, amely különféle gépi tanulási algoritmusokat és modulokat kínál a gépi látás és a médiaelemzés feladatokhoz.
- `mp_hands = mp.solutions.hands`: Ez a sor létrehoz egy kézérzékelő objektumot a Mediapipe könyvtár segítségével, amelyet a későbbi kézérzékelési feladathoz fogunk használni.
- `mp_drawing = mp.solutions.drawing_utils`: Ez a sor létrehoz egy rajzolási segédprogram objektumot a Mediapipe könyvtárból, amelyet a kézérzékelés eredményeinek megjelenítésére fogunk használni.
- `hands = mp_hands.Hands(static_image_mode=False, max_num_hands=1, min_detection_confidence=0.5, min_tracking_confidence=0.5)`: Ez a sor létrehoz egy "hands" nevű kézérzékelő objektumot, és beállítja annak paramétereit. Például kikapcsolja a statikus képmódot, megadja a maximális kéz számát (1), és beállítja az érzékelés és nyomon követés minimális bizalmi szintjét. Ez az objektum lesz felelős a kézérzékelés végrehajtásáért a későbbi műveletek során.

A count_fingers függvény az adott kéz landmarkjainak alapján számolja ki az ujjak számát és állapotát. A függvény részletei a következők [18], [19]:

```
def count_fingers(landmarks):
    fingertip_indices = [4, 8, 12, 16, 20]
    finger_base_indices = [2, 5, 9, 13, 17]
    finger_status = []

    thumb_tip_idx = fingertip_indices[0]
    thumb_base_idx = finger_base_indices[0]
    thumb_tip = landmarks[thumb_tip_idx]
    thumb_base = landmarks[thumb_base_idx]
    index_finger_base = landmarks[finger_base_indices[1]]
```

[18] https://python.plainenglish.io/finger-counting-with-python-mediapipe-and-opencv-c26bc29fbb2

```
 vector_thumb_base_tip = np.array(thumb_tip) - np.array(thumb_base)
 vector_thumb_base_index = np.array(index_finger_base) - np.array(
thumb_base)

 dot_product = np.dot(vector_thumb_base_tip, vector_thumb_base_index)
 mag_thumb_base_tip = np.linalg.norm(vector_thumb_base_tip)
 mag_thumb_base_index = np.linalg.norm(vector_thumb_base_index)
 angle_rad = np.arccos(dot_product / (mag_thumb_base_tip *
mag_thumb_base_index))
 angle_deg = np.degrees(angle_rad)
 if angle_deg < 90:
     finger_status.append(0)
 else:
     finger_status.append(1)

 for idx, (tip_idx, base_idx) in enumerate(zip(fingertip_indices[1:],
finger_base_indices[1:])):
     if landmarks[tip_idx][1] < landmarks[base_idx][1]:
         finger_status.append(1)
     else:
         finger_status.append(0)

 return finger_status
```

- `fingertip_indices` és `finger_base_indices`: Ezek a listák tárolják az ujjhegyek és az ujjak tövének landmarkjainak indexeit.
- `thumb_tip_idx`, `thumb_base_idx`, és `index_finger_base`: Ezek a változók tárolják a hüvelykujj hegyének és tövének, valamint az mutatóujj tövének koordinátáit.
- `vector_thumb_base_tip` és `vector_thumb_base_index`: Ezek a vektorok az ujjhegy és az ujj tövének közötti távolságot és irányt jelölik meg.
- `dot_product`: Ez a skaláris szorzat az ujjhegy és az ujj tövének vektorai között.
- `mag_thumb_base_tip` és `mag_thumb_base_index`: Ezek a változók az ujjhegy és az ujj tövének közötti távolságot jelölik meg.
- `angle_rad` és `angle_deg`: Ezek a változók tárolják az ujjak közötti szöget radiánban és fokban.

- `finger_status`: Ez a lista tárolja az egyes ujjak állapotát: 0, ha az adott ujj hajlított, és 1, ha kinyújtott.
- A `for` ciklusban megvizsgálja az összes ujjhegy és töv koordinátáit, és az alapján, hogy az ujjhegy magasabb vagy alacsonyabb-e, hozzáadja az `finger_status` listához az adott ujj státuszát.
- Végül visszaadja az `finger_status` listát, mely tartalmazza az ujjak állapotát az adott kézben.

Ez a rész a képfeldolgozó algoritmus fő ciklusát tartalmazza a kézmozgás követésére [19], [18]
.

```
cap = cv2.VideoCapture(0)

while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        continue

    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    results = hands.process(frame_rgb)

    if results.multi_hand_landmarks:
        for hand_landmarks in results.multi_hand_landmarks:
            mp_drawing.draw_landmarks(frame, hand_landmarks, mp_hands.
   HAND_CONNECTIONS)
            landmarks = np.array([[lmk.x, lmk.y, lmk.z] for lmk in
   hand_landmarks.landmark])
            finger_status = count_fingers(landmarks)

            for idx, status in enumerate(finger_status):
                cv2.putText(frame, f'Finger {idx+1}: {status}', (10, 30 +
   30 * idx), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

    cv2.imshow('Hand Tracking', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

[19] https://medium.com/@amineouahidialaoui/build-your-own-hand-detector-with-python-in-seconds-6c8ddb486d7d

```
cap.release()
cv2.destroyAllWindows()
```

Először inicializálunk egy videoforrást a cv2.VideoCapture() függvénnyel, amely a 0 indexű kamerát nyitja meg (általában a beépített webkamera). Ezután elindítunk egy végtelen while ciklust, amely addig fut, amíg a videoforrás el van zárva (cap.isOpened() feltétel igaz).

A ciklusmagban először olvasunk egy új képkockát a cap.read() függvénnyel. Ha a képkocka sikeresen beolvasásra kerül (ret értéke igaz), akkor folytatjuk a feldolgozást.

A beolvasott képkockát átkonvertáljuk RGB formátumba a cv2.cvtColor() függvénnyel, mivel a Mediapipe kézérzékelési modell RGB formátumban dolgozik. Ezután a Mediapipe keretrendszerrel (hands) feldolgozzuk a képkockát, és visszakapjuk a kézpozíciókat és az azokhoz tartozó landmarkokat.

Ha a feldolgozás eredményeként találtunk kézpozíciókat (results.multi_hand_landmarks értéke igaz), akkor a for ciklusban végigmegyünk mindegyik kézpozíción, és azokhoz rajzoljuk a Mediapipe által meghatározott landmarkokat (mp_drawing.draw_landmarks() függvény).

Ezután az enumerate() függvénnyel végigmegyünk az ujjak státuszain (finger_status) és megjelenítjük azokat a képkockán az cv2.putText() függvénnyel.

Végül a képkockát megjelenítjük a cv2.imshow() függvénnyel.

4.4.2. Adatok továbbítása a mikrovezérlőnek

A következő Python kód egy socket létrehozását és egy ESP32-eszközhöz történő csatlakozását valósítja meg [20].

```
ESP32_IP = "192.168.4.1"
ESP32_PORT = 80

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect((ESP32_IP, ESP32_PORT))
```

Az ESP32_IP és az ESP32_PORT változókban tároljuk az ESP32 hozzáférési pontjának IP címét és portszámát.

A socket modulból importált socket.socket() függvénnyel létrehozunk egy socket objektumot. Ez a socket objektum lehetővé teszi a hálózati kommunikációt.

A socket.socket() függvénynek két paramétere van:

Az AF_INET paraméter a socket címzésének típusát határozza meg, ebben az esetben az IPv4-et. Az SOCK_STREAM paraméter pedig a socket típusát, ebben az esetben TCP típusú socketet hoz létre. A client_socket.connect((ESP32_IP, ESP32_PORT)) sorban létrehozzuk a

[20] https://www.dfrobot.com/blog-994.html

kapcsolatot az ESP32-eszközzel. A connect() függvény paraméterként várja az ESP32 hozzáférési pontjának IP címét és portszámát, majd létrehozza a kapcsolatot az ESP32-rel.

A send_data() függvény folyamatosan elküldi az ujjállapotokat egy ESP32-eszköznek. Továbbá egy háttérszálat indítunk, amely a send_data() függvényt hivja meg.

```
def send_data():
global finger_data
previous_finger_data = None
while True:
    if finger_data and finger_data != previous_finger_data:
        client_socket.sendall(finger_data.encode())
        previous_finger_data = finger_data
    time.sleep(1)

send_thread = threading.Thread(target=send_data)
send_thread.start()

finger_data = set()
```

A send_data() függvényben globális változóként van definiálva a finger_data változó, amely tárolja az ujjállapotokat. A previous_finger_data változó pedig a függvényen belül tárolja az előzőleg elküldött ujjállapotokat ez azért fontos, hogy ne küldjük el ugyan azt az állapotot kétszer.

A ciklus minden iterációjában a függvény ellenőrzi, hogy van-e friss ujjállapot (finger_data), és hogy ez az állapot eltér-e az előzőleg elküldötttől (previous_finger_data).

Ha van friss ujjállapot, akkor a függvény elküldi az ujjállapotot az ESP32-nek a client_socket.sendall() függvény segítségével.

Az time.sleep(1) függvénnyel 1 másodperces várakozást vezetünk be a függvényben, hogy ne terhelje túl a rendszert a folyamatosan futó ciklus.

Végül elindítjuk a háttérszálat (send_thread) a send_data() függvénnyel, amely a háttérben fut, így nem blokkolja a fő program végrehajtását. Ezáltal a videofelvétel objektum és a fő program folyamatosan fut, míg a send_data() függvény a háttérben küldi az ujjállapotokat az ESP32-eszköznek.

Mielőtt a program lezáródik felszabadítjuk és bezárjuk az erőforrásokat:

```
cap.release()
cv2.destroyAllWindows()
client_socket.close()
```

- `cap.release()`: felszabadítjuk a videofelvétel objektumot.

- `cv2.destroyAllWindows()`: bezár minden aktív OpenCV ablakot.
- `client_socket.close()`: lezárja a socket kapcsolatot.

4.5. Küszöbérték (Thresholding)

A robotprotézis vezérlése a küszöbölési technika segítségével történik. Miután a jel digitális formátumban érkezik meg, és az előzőleg leírt összes szükséges megfontolást figyelembe vettük, egy megfelelő küszöböt alkalmazunk az adott kvantált digitális jelre. Az EMG jel leolvasása során a megfelelő küszöbérték meghatározása számos tényezőtől függ és jelentős kihívást jelent. Az EMG jelek amplitúdója és frekvenciaösszetétele nagyban különbözhet egyénektől függően, ezért a küszöbérték választás előtt a digitálisan kvantált jelet alaposan meg kell figyelni. A küszübértéket ez alapján állitjuk majd be.

45. ábra. Küszöbérték választás

Amikor az aktuális és az előző digitális jel abszolút különbségének az értéke meghaladja ezt a küszöbértéket, a mikrovezérlő egy PWM jelet küld a motorvezérlőnek és meghajtja az 5 szervomotor. Például ha az előző jelünk értéke 2300 és az aktuális 2350, akkor egy 30-as értékű tresholdnál motorvezérlés fog történni. Ha a különbség meghaladja a spikeThreshold értéket, akkor az egy spike eseménynek számít, amely kiváltja a motorok mozgatását.

46. ábra. Kapcsolási rajz

5. Üzembe helyezés és kísérleti eredmények

5.1. Üzembe helyezési lépések

Az üzembe helyezéshez szükséges, hogy a 46. ábra alapján legyen a kapcsolás. Szükséges a jelfeldolgozó lapnak és a motor vezérlőnek egy 5V os tápláló és egy közös föld.

5.2. Kísérleti eredmények, mérések

A 47. ábrán látható a nyers EMG jel, amelyet az EMG szenzorok rögzítenek. Ez a jel még nem lett szűrve, ezért tartalmaz különböző zajokat és interferenciákat is, amelyeket a szenzorok érzékelnek a környezetből.

A 48. ábrán már látható, hogy a jel tisztább és strukturáltabb. Itt már jól látható, mikor kezdődik és mennyi ideig tart az izomaktivitás. Az eszközünk a zajokat és a felesleges interferenciákat eltávolította a jelből, így pontosabb és megbízhatóbb adatokat kapunk az izomaktivitás időbeli alakulásáról és intenzitásáról.

47. ábra. A nyers jel

48. ábra. A megszűrt jel

6. Következtetések

6.1. Megvalósítások

A projekt során egy 3D robotkarral dolgoztunk amelynek pár alkatrészét meg is terveztük, majd ezt elektromiográfiai jelek és képfeldolgozás segítségével irányítjuk. Ezen kívül egy jelfeldolgozó eszközt is létrehoztunk, amely lehetővé teszi a jelek szűrését, amelyeket majd használunk a robotkar irányítására. A jelfeldolgozó rész jelenleg csak egy csatornán működik, ami azt jelenti, hogy csak nyitott vagy zárt kézpozíciót lehet elérni vele. Azonban a képfeldolgozó rész lehetővé teszi bármilyen új mozgások utánzását a kamera képe alapján, ebben a részben egy számítógéptől fogadjuk az adatokat, amely a kamera kép alapján meghatározza hogy milyen állapotba legyenek az ujjak. Az adatok wireless-en keresztül jutnak át a számítógépről a mikrovezérlőre használva az esp32 WIFI modulját mint elérési pont.

6.2. Hasonló eszközök

6.2.1. Ada Robotic Hand - Open Bionics

Az Ada Robotic Hand az Open Bionics által fejlesztett, nyílt forráskódú protézis robotkéz, amely innovatív megoldásokat kínál a végtagveszteséggel élők számára. A rendszer lineáris motorokat használ, ezek a motorok egy lineáris mechanizmust használnak a mozgás előállításához, ellentétben a hagyományos forgó motorokkal, amelyek körkörös mozgást hoznak létre [21]. Mind a motorok, mind a PCB a kézfejben találhatók, ez egy kompakt kialakítást eredményez, ezt láthajuk a 49. ábrán.

49. ábra. Ada Robotic Hand

[21] https://www.instructables.com/Ada-Robotic-Hand-Open-Bionics/

6.2.2. Touch bioinics

A Touch Bionics egy vezető protézis kézgyártó vállalat [22].

- az ujjak titániummal megerősítettek és külön-külön motorizáltak
- az ujjak sebessége akár 30%-kal is növelhető
- app vezérlés
- triggereket használ az ujjak specifikus fogásainak aktiválásához
- 12 különböző automatizált fogási lehetőség
- árkategória: 10 000 USD körül

6.2.3. Saját eszköz

Az általam készített robotkéz egy egyszerű, megoldás a kéz mozgásának utánzására, amely a következő jellemzőkkel rendelkezik:

Egycsatornás jel feldolgozás: Az eszköz egyszerűsége miatt csak két állapotot tud érzékelni és megjeleníteni: nyitott és zárt kéz.

Kicsi és könnyű szervomotorok: Az eszköz kisméretű és könnyű szervomotorokat használ, amelyek mégis elegendő erőt biztosítanak a kéz mozgásának kivitelezéséhez.

Kameravezérlés: Az egyik legjellegzetesebb különbség az eszközöm és más hasonló megoldások között az, hogy az enyém kamerával is irányítható, itt már több kézmozdulat is lehetséges.

Olcsó előállítás: Mivel saját alapanyagokkal volt előállítva és nem megvásárolt eszközökkel így sokkal olcsóbb a végtermék.

6.3. További Fejlesztési Lehetőségek

Az egyik legfontosabb fejlesztési lehetőség egy többcsatornás EMG eszköz alkalmazása vagy kifejlesztése, amellyel növelhetjük a rendszer képességeit. Az új eszközzel különböző kézmozdulatokról mintákat gyűjtünk, majd ezeket egy neurális hálózatnak tanítjuk meg. A neurális hálózat segítségével pontosabban és többféle kézmozdulatot tudunk felismerni és azonosítani, amelyekkel a robot kezet irányíthatjuk.

A képfeldolgozási módszerek továbbra is fontos szerepet játszhatnak a fejlesztésben. Az EMG jelek és a képfeldolgozás kombinálásával még pontosabb eredményeket érhetünk el.

További fejlesztési lehetőség a csukló szerinti forgás implementálása. Miután már rendelkezünk egy többcsatornás jelfeldolgozó eszközzel, további szenzorokat integrálhatunk a csukló

[22] https://www.ossur.com/en-us/prosthetics/arms/i-limb-access-titanium

forgásának mérésére. Ezáltal a robot kéz nemcsak nyitott és zárt állapotban, valamint különböző kézmozdulatokban lesz irányítható, hanem a csukló elfordításával is. Ezáltal irányítani tudjuk a robot kezet EMG jelekkel és képfeldolgozás módszerekkel.

Hivatkozások

[1] Sebastián Aced López, "Design and Construction of an EMG Multichannel Acquisition System Prototype".

[2] Muhammad Zahak Jamal, "Signal Acquisition Using Surface EMG and Circuit Design Considerations for Robotic Prosthesis".

[3] Texas Instruments, "Analog Front-End Design for ECG Systems Using Delta-Sigma ADCs".

[4] Jorge R. B. Garay, Arshpreet Singh, Moacyr Martucci, Hugo D. H. Herrera, Gustavo M. Calixto, Stelvio I. Barbosa, Sergio T. Kofuji, "An Electromyography Signal Conditioning Circuit Simulation Experience".

[5] Enrique Mario Spinelli, Ramon Pallàs-Areny, Miguel Angel Mayosky, "AC-Coupled Front-End for Biopotential Measurements".

[6] Texas Instruments, "INA826 Precision, 200-µA Supply Current, 3-V to 36-V Supply Instrumentation Amplifier With Rail-to-Rail Output - 2011".

[7] Texas Instruments, "MCP3201 - 2.7V 12-Bit A/D Converter with SPI Serial Interface Data Sheet".

[8] Guobiao Yang, Zhenyuan Sun, Zhao Wang, Gangdong Sun, "Detection and Identification of the Human Surface Electromyography Signal".

[9] Muhammad Zahak Jamal, "Signal Acquisition Using Surface EMG and Circuit Design Considerations for Robotic Prosthesis".

[10] MuscleDrive: A Proof of Concept describing the Electromyographic Navigation of a Vehicle Ritwik Raj Saxenaa, Stanley Sujith Nelavalab, Ritcha Saxenac

A. Függelék

50. ábra. Bemeneti védelem, insztrumentális erősítés és közös módusú zaj visszacsatolás

51. ábra. Analóg szűrő és analóg-digitális jelátalakító

52. ábra. Tapegység áramkör