Besondere Lernleistung:

Konzeption und Herstellung einer elektronischen Hand- und Unterarmprothese, die durch 3D-Druckverfahren hergestellt und drahtlos mit einem externen Handschuh mit Fingerbeugungssensoren gesteuert wird auf Basis des ESP32 Mikrocontrollers

Von Philipp Lemke
Abitur 2021

Unter Begleitung von Herrn Marc Lachmann

Besondere Lernleistung / 5. Abiturfach von Philipp Lemke (Abiturjahrgang 2021)

Adresse des Autors:
Philipp Lemke
Ziegelweg 6
47198 Duisburg
Deutschland

Adresse der Schule:
Gymnasium in den Filder Benden
Zahnstraße 43
47447 Moers
Deutschland

In Auftrag gegeben von Herrn Marc Lachmann (Mentor).
Finanziell unterstützt durch RehaMedia Duisburg.

Kontakt: info@philipplemke.com

Alle Quellen finden sich auf den letzten Seiten unter dem Kapitel „Quellenverzeichnis“. Bei Zitaten oder Verweisen, bzw. da wo eine Quellennennung notwendig ist, findet sich in der Arbeit eine Anmerkung durch eckige Klammern mit einer Zahl (z.B.: [1]). Die Quelle, bzw. der Link dazu kann im Quellenverzeichnis unter dieser Zahl gefunden werden. Zitate, bzw. Eins zu Eins Nennungen eines Originaltextes werden kursiv gedruckt und können ebenfalls durch die Zahl in eckigen Klammern im Quellenverzeichnis gefunden werde. Ein „[…]“ verweist darauf, dass ein Teil des Zitats entfernt wurde.

Inhaltsverzeichnis

1 Konzeption

1.1 Vorgeschichte

Schon seit 2017 beschäftige ich mich intensiv mit dem 3D-Druck und der Mikroelektronik über die Arduino Plattform. Anfang Mai 2020 habe ich über das Thema „3D-Druck Handbuch für junge Einsteiger“ ein Taschenbuch geschrieben und auf Amazon veröffentlicht, was bis dato (7.4.2021) über 250-mal an Privatleute in Form eines Taschenbuches oder E-Books verkauft wurde und es bis auf Platz 6 der Amazon-Bestsellerliste in der Kategorie „Hardware & Technik“ geschafft hat [1].

Nach ein paar Monaten der Arbeit mit dem 3D-Druck in der Schule stoß ich im Mai 2018 auf ein YouTube-Video von „Clixoom Science & Future“ mit dem Titel „Kommen jetzt die Roboter? InMoov - Der Roboter für Jedermann“ [2]. Es geht um einen Roboter eines französischen Bildhauers, der die 3D-Druck Dateien auf der Website Thingieverse.com veröffentlichte. Man hat die Möglichkeit mit seinem eigenen 3D-Drucker die Teile für einen Roboter auszudrucken und diesen zu programmieren. Diese Idee hat mich sofort beeindruckt und Ich besuchte die Website von Gaël Langevin unter www.inmoov.fr.

1.2 Idee

Mit der Zeit entwickelte sich die Idee einer Handprothese, die als geschlossenes Produkt entwickelt wird, was bedeutet, dass z.B. keine Kabel heraushängen und mit einem einfachen Knopfdruck startet und keine Wartung oder Ähnliches benötigt. Zum Tag der offenen Tür 2018 war dann auch der erste Prototyp der Prothese fertiggestellt. Allerdings nur als ein Produkt, das ständige Wartung benötigt und keine Elektronik im inneren besaß und auf eine externe Stromzufuhr angewiesen war. Die Anmeldung für die besondere Lernleistung stand auch schon bald an. Somit entschied ich mich die Handprothese nochmal neu zu bauen und einiges in der Konstruktion zu verändern. Um diese nicht nur mit vorher einprogrammierten Werten zu steuern, entschied ich mich dafür einen Handschuh zu bauen, der über Beugungssensoren an den Fingern verfügt. Diese soll die Werte verarbeiten und drahtlos zur Prothese schicken, die in Echtzeit die Fingerbewegung ausführt. Somit kann man die Prothese, die eine rechte Hand und einen Unterarm darstellt mit dem Handschuh auf der linken Hand steuern.

1.3 Inmoov

Das Modell der Prothese basiert auf dem von Gael Langevin. Er ist französischer Bildhauer und Designer und arbeitet seit 25 Jahren für große Marken und Firmen. Inmoov ist sein persönliches Projekt, an dem er schon seit Januar 2012 arbeitet. Er veröffentlichte seine erste Handprothese unter einer Open-Source Lizenz. Seine Vision: Jeder mit einem 3D-Drucker mit einer minimalen Druckgröße von 12cmx12cmx12cm kann sich eine Handprothese drucken, um damit zu lernen und diese weiterzuentwickeln. [3] Laut Wikipedia gilt „Das InMoov Projekt […]

innerhalb der Maker-Szene als sehr anspruchsvoll“ [6] Auch an mehreren Hochschulen wurden schon Projekte durchgeführt [6]. Doch all dies hielt mich nicht davon ab und ich fing an.

Doch zuallererst startete ich damit die Teile für den Kopf herzustellen, den er später designte (ca. 2015). Früh erkannte ich, dass mir vieles, was Langevin designte nicht gefällt und startete mit Tinkercad und später mit Fusion 360 sein Design umzuändern und darauf anzupassen, viel Elektronik, anstatt Mechanik einzubauen. Mit Langevin’s Design war es möglich, dass sich Augen, Mund und der Kopf selbst in die verschiedensten Richtungen bewegen kann. Ich wollte aber nur eine einfache Mundbewegung, damit ich Platz hatte um unteranderem einen Arduino MEGA, einen Raspberry Pi und einen Akku einzubauen. Somit habe ich den ganzen Innenraum selbstdesignt. Später diente der Kopf auch als Steuerung für den Arm, den ich danach baute.

Die Bauteile für den Inmoov Roboter sind nach *Attribution-NonCommercial 3.0 Unported (CC BY-NC 3.0)* lizensiert [8]. Dies erlaubt das Teilen bzw. die Vervielfältigung und Weiterverbreitung des Materials in jedwedem Format oder Medium und das Bearbeiten des Materials, was remixen, verändern und darauf aufbauen beinhaltet. Das Material darf nicht für kommerzielle Zwecke genutzt werden [9].

1.4 Arduino

Arduino ist eine aus Soft- und Hardware bestehende Physical-Computing-Plattform. Beide Komponenten sind quelloffen. Die Hardware besteht aus einem einfachen E/A-Board mit einem Mikrocontroller und analogen und digitalen Ein- und Ausgängen. Die Entwicklungsumgebung basiert auf Processing und soll auch technisch weniger Versierten den Zugang zur Programmierung und zu Mikrocontrollern erleichtern. Die Programmierung selbst erfolgt in einer C- bzw. C++-ähnlichen Programmiersprache, wobei technische Details wie Header-Dateien vor den Anwendern weitgehend verborgen werden und umfangreiche Bibliotheken und Beispiele die Programmierung vereinfachen. […] [4]

Schon in der 8. Und 9. Klasse habe ich viel mit Arduino gearbeitet. Damals war dies sogar auch Teil des Lernplans. Wir bauten kleine Schaltkreise mit LED‘s, die wir dann mit simpler Programmierung zum Blinken brachten.

Die Arduino Plattform ermöglicht es mithilfe eines Entwicklungsboards wie dem Arduino UNO (siehe Abb. 1) elektrische Signale zu erzeugen und somit Elektronik mit einer Spannung bis zu 5 Volt anzusteuern. Mit der kostenfreien Arduino IDE (= „Integrierte Entwicklungsumgebung“) kann man den Arduino programmieren.

Abbildung 1 [5]

Die Arduino Plattform ist somit die ideale Entwicklungsumgebung, um eine Handprothese zu bauen, wie in Kapitel 1.2 beschrieben. Ein Arduino verfügt über einen Mikrocontroller, der einfache Rechenoperationen schnell durchführen kann. Mittlerweile gibt es zahlreiche Arduinos von verschiedenen Herstellern auf dem Markt, da Arduino Open-Source ist. Das große Spektrum an verschiedenen Arduinos ermöglicht es, auch schnellere oder spezialisiertere Arduinos zu erwerben. Wie z.B. welche mit einem 32 Bit Prozessor, mehr Flash-Speicher oder integriertem OLED-Display. Die verschiedenen digitalen und analogen Ein- und Ausgänge des Arduinos ermöglichen die Kommunikation und Ansteuerung mit verschiedenen Bauteilen. Hier kann also Daten gesendet und eingelesen werden. Die serielle Schnittstelle (TX und RX) wird genutzt, um mit dem Computer zu kommunizieren. Hierüber wird der Arduino nicht nur programmiert, sondern kann auch serielle Datenpakete an den Computer verschicken oder vom Computer empfangen, was ein einfaches Debugging oder die Kommunikation mit mehreren Arduinos ermöglicht.

1.5 Der 3D-Druck

Zur Herstellung aller Bauteile aus Plastik verwende ich den 3D-Druck. Im Mai 2020 habe ich zu der Thematik 3D-Druck ein Buch [10][11] herausgebracht. Mit über 250 Verkäufen, einer Bestsellerposition in der entsprechenden Kategorie und einer 4,1 von 5 Sterne Nutzerbewertung auf Amazon (Stand 7.4.2021) nutze ich dieses Buch als Literaturbezug für den Großteil aller Quellen zum Thema 3D-Druck in meiner Arbeit. Vieles im 3D Druck basiert auf Eigenerfahrung und auch so habe ich mir alles selbst durch experimentieren angeeignet.

1.5.1 FDM-Druck

Am meisten habe ich das FDM-Druckverfahren genutzt, da es das häufigste Druckverfahren, das einfachere und viel vorteilhaftere zu anderen Verfahren ist. FDM heißt „Fused Deposition Modeling“. Dieses beschreibt ein schichtweises Aufbauen von Kunststoff und dessen Verschmelzung mit anderen Schichten.

Ein solcher 3D-Drucker verfügt über drei Bewegungsachsen, an denen ein Druckkopf befestigt ist. Mithilfe von Servomotoren kann sich dieser in den Achsen bewegen und alle Punkte des Druckbereichs anfahren. Der Druckkopf oder Printhead ist an den x- und y-Achsen befestigt. Seine Hauptaufgabe ist es, das kalte stangenförmige Kunststoff, auch genannt Filament zu erhitzen und an bestimmten Positionen zu extrudieren. Meistens wird er auch nur Hotend genannt. Diese Bezeichnung bezieht sich aber eigentlich nur auf den Teil, der erhitzt wird. Der Printhead besteht also aus einem elektrisch erhitzten Rohr, Lüfter und der Düse, auch genannt Nozzle. Die Nozzle ist in dem erhitzen Rohr festgeschraubt, sodass sie einfach gewechselt werden kann. Die Nozzle wird also auf eine Betriebstemperatur zwischen 190° und 250° Celsius erhitzt.

Nachdem das Filament erhitzt ist, wird auf die Druckfläche, auch Printbed genannt aufgetragen. Das Printbed ist die Arbeitsfläche. Sie wird durch den Servomotor der z-Achse vertikal bewegt. Das Printbed selbst besteht aus einer

Edelstahl- oder Aluminiumplatte, auf der sich eine spezielle Folie befindet, die sich durch elektrischen Strom erhitzen kann. Die meisten Drucker arbeiten mit einer weiteren Platte aus PLA - Kunststoff, die es dem Nutzer erlaubt, auch ohne Erhitzung zu drucken. Die Firstlayer ist, wie der Name schon sagt, die erste Schicht, die gedruckt wird. Auch manchmal als mainlayer bezeichnet ist diese Schicht die wohl wichtigste des ganzen Drucks. Wenn sie Fehler aufweist, wird der gesamte Druck fehlerhaft oder beschädigt sogar den Drucker. Es ist essenziell, dass die Firstlayer komplett auf dem Printbed haftet. Damit dies funktioniert ist das Printbed, abhängig vom Filament Typ auf z.B. 60° C. erhitzt. Kunststoff dehnt sich bei verschiedenen Temperaturen aus, bzw. zieht sich zusammen. Das sehr heiße Filament wird aus der Nozzle auf das kältere Printbed extrudiert. Dabei zieht es sich durch die schnelle, relative Abkühlung zusammen und haftet perfekt auf dem Printbed. Nicht jedes Filament ist gleich und somit muss auch die Temperatur manuell eingestellt werden, damit sie der entspricht, bei der das Filament haften bleibt. Der Hersteller gibt aber immer eine ungefähre Temperatur an. Dennoch sollte man sich hier nicht immer auf die Herstellerwerte verlassen. Die Erfahrung zeigt meist immer was anderes und genaueres, denn die Werte können sich bei verschiedenen Modellen und 3D-Druckern verändern, sodass die Werte immer überprüft und spezifisch eingestellt werden müssen

Die Firstlayer kühlt leicht ab. Der Druckkopf wandert eine Schicht nach oben und druckt die nächste Schicht. So entsteht in diesem additiven Fertigungsprozess ein dreidimensionales Objekt. Es gibt verschiedene Filamente, die je nach Zusammensetzung. Die meistgenutzte Filament-Art ist PLA (Polylactide oder Polymilchsäuren). Die Kompostierfähigkeit des Filaments macht es besonders beliebt. Es ist einfach zu handhaben und kann schnell und ohne viel Erfahrung gedruckt werden, weshalb es auch bei den meisten 3D-Druckern direkt mitgeliefert wird. Zudem ist es noch günstig. Leider kommen die einzelnen Arten auch mit Nachteilen. PLA ist z.B. weit entfernt von bruchsicher, womit es sich eigentlich nur für Prototypen eignet. Für eine stabilere Produktion eignet sich dann besser ABS. Doch der Umgang mit diesem Filament ist besonders kompliziert. Er kann z.B. auch nur in belüfteten Räumen gedruckt werden und neigt zu ungewünschten Verformungen bei falscher Anwendung. [11]

Im Späteren werde ich auch auf die verschieden Plastik-, bzw. Filamentarten eingehen, die ich genutzt habe.

1.5.2 SLA-Druck

Außerdem nutze ich auch ein anderes Druckverfahren, der sogenannte SLA-Druck (Stereolithografie-3D-Druck). Bei diesem befindet sich ein Kunstharz in einer durchsichtigen Wanne auf einem UV-Belichtungsdisplay. Ein Printbed fährt über einen Servomotor in die Kunstharzwanne. Eine Schicht wird durch das Display belichtet. Das Harz erhärtet. Nun kann das Printbed etwas hochfahren und eine weitere Schicht kann belichtet werde. Das SLA-Verfahren hat den Vorteil, dass extrem genaue Modelle gebaut werden können. Mit dem bloßen Auge sieht man den Modellen meist gar nicht an, dass sie 3D-gedruckt sind.

Allerdings ist dieser Herstellungsprozess sehr kostenintensiv. Alle Teile des Druckers und des Drucks, die mit Kunstharz in Kontakt gekommen sind, müssen nach einmaliger Benutzung mit 2-Propanol Alkohol gereinigt werden, was den gesamten Prozess zusätzlich gefährlicher und zeitintensiver macht. Auch das Kunstharz, vor allem billige sind in den meisten Fällen krebserregend und oder gesundheitsschädlich bei Einatmen der Dämpfe oder Kontakt von Körper und Kunstharz. Es ist also zwingend notwendig, dass sämtliche Schutzausrüstung, wie Atemfilterschutz, Handschuhe und Augenschutz getragen werden muss. Alle diese Aspekte macht das SLA Verfahren ziemlich komplex, weshalb Ich es für die Herstellung der Prothese nur selten angewendet habe.

1.6 Der erste Prototyp

Wie in Kapitel 1.2 schon angesprochen, habe ich für den Tag der offenen Tür 2018 einen ersten Prototyp angefertigt. Dieser basierte noch stark auf dem Modell, das einst Langevin entwarf. Der Roboterkopf war zu diesem Zeitpunkt auch schon fertig und besaß die nötige Elektronik und Stromversorgung, um den Arm zu steuern. Der Entwurf des Arms war somit nicht als Prothese, sondern als Teil eines humanoiden Roboters konstruiert. [7] Für alle Teile, die durch 3D-Druck hergestellt wurden, wurde das FDM-Verfahren genutzt.

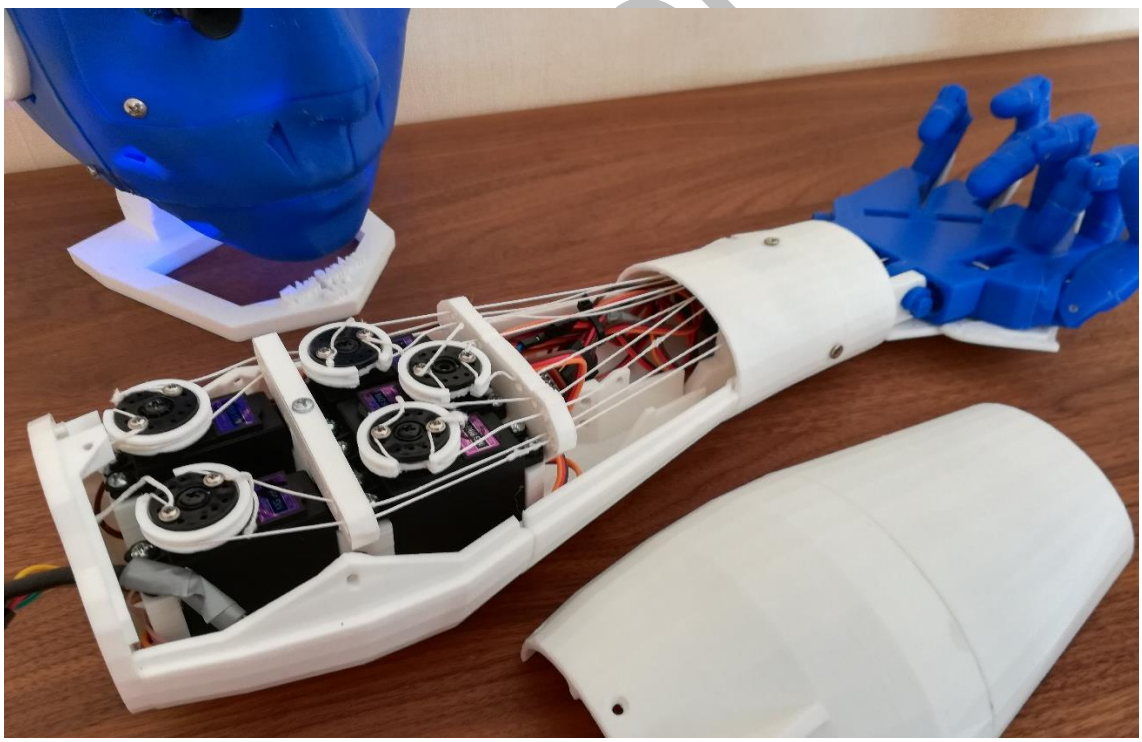

Abbildung 2 [7]

Begeisterung unter den BesucherInnen des Tags der offenen Tür 2018 machte sich bemerkbar. Allerdings konnte die Hand sich nur öffnen und schließen. Ein kleines Programm lief ab, dass jeden Finger einzeln öffnet und schließt und dann dies danach mit allen Fingern gleichzeitig durchführte. Wie es aber bei Vorführobjekten ist, geht auch schonmal etwas kaputt. Vor allem dann, wenn versucht wird Finger mit Gewalt zu öffnen. Die Seile, die ich nutzte, waren nicht sehr beanspruchbar und rissen ständig. Nicht nur das, sondern viele Mängel

fielen auf und ich arbeitete ständig an Verbesserungen. Für den Tag der offenen Tür 2019 hatte ich dann vorbereitet, dass es möglich war den Roboterarm mit einer Browser-Applikation zu steuern. Bzw. die BesucherInnen konnten sich mit einem Tablet mit dem Roboterkopf verbinden und auf dem Tablet verschiedene Button drücken, um die Finger einzeln zu öffnen und zu schließen. Viele weitere Prototypen folgten, bis ich so weit war den aktuellen und meinen letzten Prototypen vorzustellen, der als fertiges und Produkt funktioniert und als ferngesteuerte Prothese nutzbar ist. Diese Dokumentation zeigt den Verlauf meiner Konzeption und Konstruktion dieses letzten Prototypens.

1.7 Entwicklungsverlauf

Für die Planung und die Konzeption habe ich einen leicht abweichenden Verlauf zu anderen Entwicklungen gewählt. Für mein Projekt benötige ich viele verschiedene Bauteile und Produkte, die gekauft, bzw. bestellt werden müssen. Allerdings habe ich diese nicht erst nach Beendigung der Planung gekauft, sondern währenddessen. Bei Entwicklungen dieser Art gibt es keinen stringenten Verlauf. Es muss experimentiert werden und deshalb habe Ich auch verschiedene Produkte ausprobiert. Generell behandelt diese Dokumentation den „kürzesten Weg zum Ziel“, das heißt, es gab wie bei jeder Entwicklung sogenannte „Branches“, ein Begriff, der aus der Softwareentwicklung kommt, bzw. „Abzweigungen“ in andere Versuche, um auf das gleiche Projektziel zu kommen. Alle Branches würden in ihrer Komplexität und Länge vermutlich den Rahmen dieser Dokumentation sprengen und wurden verworfen, da sie nicht auf das gewünschte Ergebnis gekommen sind, weshalb ich mich auf nur den einen Lösungsweg konzentriere, den ich gewählt habe. Allerdings werde Ich auch einige Anknüpfungen zu anderen Branches machen, um zu untermauern, warum ich mich für eine bestimmte Methode entschied.

1.8 Einkauf

In diesem Kapitel führe ich zuerst die Bauteile und Produkte auf, die ich im fertigen Projekt (Hauptbranche) nutze und erkläre, wofür sie sind und welche besonderen Eigenschaften vorhanden sind, weshalb ich mich für einen Kauf entschieden habe. Anschließend alle anderen Produkte, die ich für andere Branches genutzt habe, die es aber nicht in das fertige Projekt geschafft haben und warum diese nicht benutzt wurden.

Viele der Produkte stammen von AZ-Delivery, einem in Bayern ansässigen Lieferanten für Mikroelektronik.

1.9 Einkauf für die Prothese

1.9.1 Servomotoren MG995

Die Motoren, die die Finger bewegen sind so genannte Servomotoren. Diese können sich um 180° drehen und werden sehr präzise eingestellt. Sie benötigen keine zusätzliche Elektronik und können einfach mit einem PWM-Signal (Pulsweitenmodulation) über den Arduino angesteuert werden. [12]

Abbildung 3 [12]

Produktname	MG995 Micro Digital Servo Motor
Hersteller	AZ-Delivery
Anzahl	5
Kaufdatum	9.9.2020
Gesamtpreis	29,79€
Eingangsspannung	5V
Ausgangsspannung	keine
Besondere Eigenschaften	• Preis/Leistung optimal • Ausreichende Zugkraft • Präzise Einstellung der Drehposition • Metall-Getriebe

[12]

1.9.2 Battery Shield

Für die Stromversorgung der Prothese ist ein Battery Shield nötig. Es dient als Halterung für die Akkus, die im Endeffekt die Stromversorgung übernehmen. Außerdem kann das Shield das Lademanagement übernehmen, weshalb es nicht nötig ist, die Prothese zu öffnen damit man die Akkus herausnehmen und dann erst laden kann. [14]

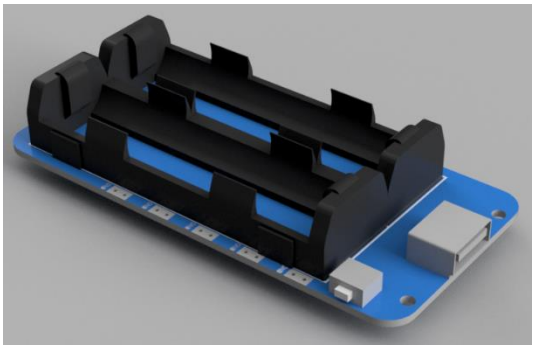

Abbildung 4 [13]

Produktname	18650 Battery Shield V8
Hersteller	diymore
Anzahl	1
Kaufdatum	13.10.2020
Gesamtpreis	10,50€
Eingangsspannung	5V
Ausgangsspannung	5V
Ausgangsstrom	Maximal 3A
Besondere Eigenschaften	• Integrierter Überspannungsschutz • Lademanagement • Hoher Ausgangsstrom (max. 3A)

[14]

1.9.3 Akkus

Die Stromquelle der Prothese sind zwei Standard VTC6 Akkus. Sie befinden sich im Battery Shield (K. 1.8.1.2). [16]

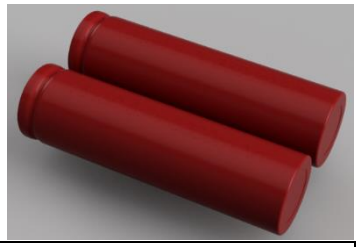

Abbildung 5 [15]

Produktname	VTC6 Akku
Hersteller	Tatopa
Anzahl	2
Kaufdatum	13.10.2020
Gesamtpreis	17,99€
Eingangsspannung	3,7V
Ausgangsspannung	Maximal 4,2V
Ausgangsstrom	Maximal 30A
Maximale Kapazität	Maximal 3080mAh
Besondere Eigenschaften	• Hohe Belastbarkeit, bzw. hoher Maximalstrom • Hohe Kapazität • Nach UN38.3 auf Sicherheit getestet

[16]

1.9.4 ESP32 Dev Kit C V4 NodeMCU

Die Hauptrecheneinheit der Prothese ist der ESP32 NodeMCU. Er ist ein Arduino Nachbau und verfügt über zahlreiche Besonderheiten, die ihn zu einem perfekten Entwicklungsboard machen. Er besitzt einen dual-core 32 bit Prozessor und einen integrierten Controller für Wi-Fi und Bluetooth. Die Programmierung über die Arduino IDE macht ihn besonders zugängig. Verschieden Bibliotheken für die Nutzung der WLAN und Bluetooth Funktionen sind auch verfügbar. Außerdem besitzt er viel mehr Anschlüsse (Pins) als ein normaler Arduino (wie in K. 1.4). [18]

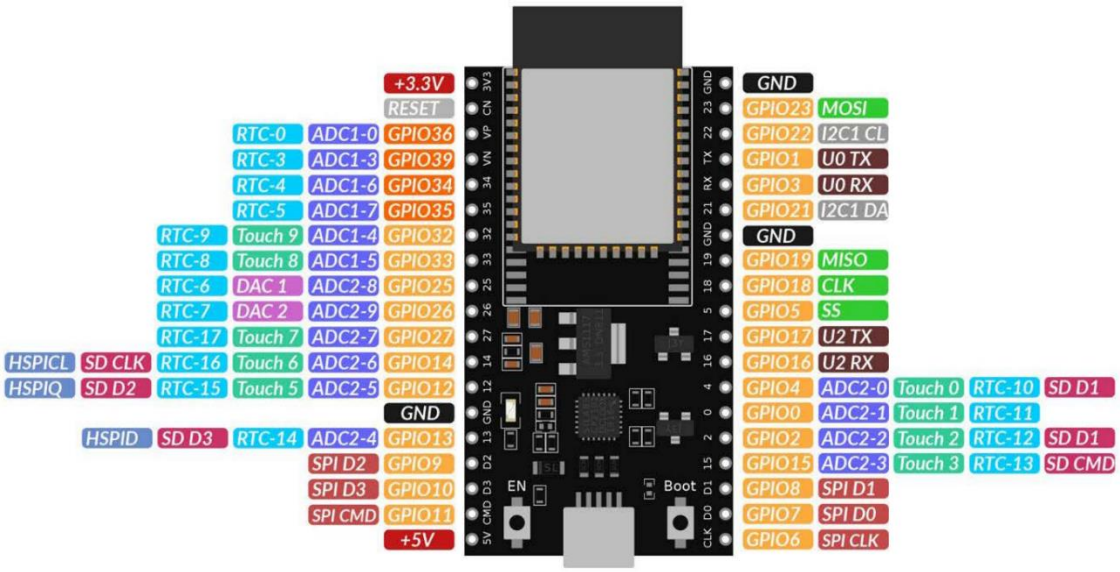

Abbildung 6 [17]

Die Pinbelegung kann aus dem Datenblatt entnommen werden und wird später für die Programmierung benötigt, damit die Pins, an denen die Bauteile angeschlossen sind, richtig adressiert werden.

Produktname	ESP32 Dev Kit C V4 NodeMCU
Hersteller	AZDelivery
Anzahl	1
Kaufdatum	20.03.2021
Gesamtpreis	9,49€
Eingangsspannung	3,3V oder 5V
Ausgangsspannung	Maximal 5V
Mikroprozessor	Tensilica Xtensa LX6
Prozessor-geschwindigkeit	240 MHz
Wireless Standards	FCC, CE, IC, TELEC, KCC, SRRC, NCC
Frequenzbereich	2,4 – 2,5 GHz
Bluetooth Protokoll	Bluetooth v4.2, BR/EDR, BLE
Speicher	4mB Flash, 520kB SRAM

Physische Schnittstellen	UART, I2C, SPI, I2S, PWM, SDIO, GPIO, ADC, DAC
Besondere Eigenschaften	• Hohe Leistung im Gegensatz zu anderen Arduinos • Viele Verbindungsmöglichkeiten • Kostengünstig • Programmierung durch Arduino IDE möglich

[17], [18]

1.9.5 Leiterplatine

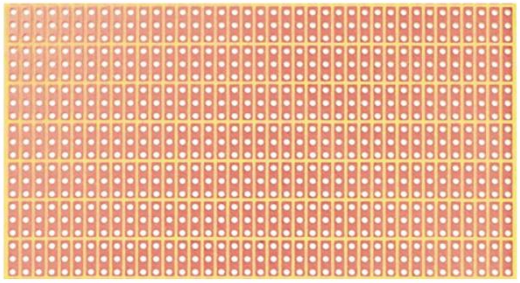

Abbildung 7 [19]

Die Elektronik wird fest auf einer Leiterplatine eingelötet, um zu garantieren, dass keine Kurzschlüsse oder ein Ablösen von Steckverbindungen vorkommen. Das 3er Streifenraster sorgt dafür, dass viel mehr Elektronik auf eine kleinere Fläche passt, was benötigt wird, da die Prothese nur einen kleinen Raum für Elektronik besitzt. Die End-Größe entspricht nicht der Originalgröße der Platine, sie wird zugeschnitten, damit es passt.

Produktname	790-1 3er Streifenraster Leiterplatte
Hersteller	Rademacher
Anzahl	25
Kaufdatum	23.03.2021
Gesamtpreis	26,30€
Größe	50 x 100 x 1,6 mm
Rastermaß	2,54 mm
Material	Kupfer mit einer Stärke von 35 µm
Besondere Eigenschaften	• Qualitativ hochwertige Leiterplatinen aus deutscher Herstellung • 3er Streifenraster (nicht durchgängig)

[19]

1.9.6 USB-Buchse

Damit die Prothese zum Aufladen nicht geöffnet werden muss, ist es nötig eine Micro USB-Buchse anzubringen. Dadurch kann ein herkömmliches USB 2.0 Micro-B Kabel angeschlossen werden, dass mit 5V betrieben wird.

Abbildung 8 [20]

Produktname	USB 2.0 Micro-B Buchse
Hersteller	DeLock
Anzahl	1
Kaufdatum	13.10.2020
Gesamtpreis	5,57€
Länge	25cm
Besondere Eigenschaften	• Einfacher Einbau durch vorgebohrte Schraublöcher an der Buchse

[20]

1.9.7 PLA Carbon Filament

Das Filament für den 3D-Drucker ist das Carbonfaser PLA Filament von Enotepad. Wie der Name schon sagt, beinhaltet es Carbonfaser, bzw. Kohlenstofffaser. Sie besitzen nur eine Dicke von 5 bis 9µm und haben die Eigenschaft besonders reißfest, steif und äußerst leicht zu sein [22]. Dies macht das Material zu einem idealen Zusatz zu normalem PLA und ist perfekt, um damit eine Prothese zu bauen, da alle genannten Eigenschaft benötigt werden. Die Carbon-schwarze Farbe verleiht der Prothese außerdem einen futuristischen Stil.

Abbildung 9 [21]

Produktname	PLA Carbon Fiber 3D Drucker Filament 1.75mm, 1kg
Hersteller	Enotepad
Anzahl	2
Kaufdatum	5.09.2020
Gesamtpreis	65,98€
Gewicht Pro Rolle	1000g
Kunststoff	PLA (Polylactide)
Zusatzstoff	Carbonfaser
Besondere Eigenschaften	• Äußerst starr und bruchsicher • Hitzeresistenter als normales PLA

[21]

1.9.8 Gehärtete Edelstahl Nozzle

Abbildung 10 [24]

Damit man mit Carbon-verstärktem Filament drucken kann muss auch die Nozzle verstärkt werden. Sie ist die Öffnung, durch die das Filament austritt. Standardmäßig ist eine Nozzle aus Messing. Carbon ist aber härter als Messing, was dafür sorgt, dass die Messing-Nozzle stark abgenutzt wird und sich die Öffnung der Nozzle (Standard 0,4mm) auf mehrere Millimeter erweitert, was dazu führt, dass Drucke sehr ungenau werden und die Qualität dieser stark sinkt. Aus diesem Grund gibt es Edelstahl-Nozzle. Diese sind härter und halten der Abnutzung durch das Carbon länger stand. Normalerweise werden Edelstahl-Nozzlen nur selten, bzw. nur bei solchen speziellen Filamenten genutzt, da Edelstahl nicht so stark wärmeleitend ist wie Messing.

Produktname	3D Drucker Düsen/Nozzle Volcano Set Edelstahldüsen
Hersteller	ruthex
Anzahl	1 Paket. 6x Messingdüsen 6x Edelstahldüsen
Kaufdatum	2.09.2020
Gesamtpreis	16,99€
Material	Edelstahl
Filament-durchmesser	1,75mm
Nozzle-Form	Volcano
Besondere Eigenschaften	• Härter als Standard-Nozzlen

[23]

1.9.9 LM2596S DC-DC Netzteil


Abbildung 11 [30]

Wie Ich auch noch im späteren Verlauf der Dokumentation erklären werde, wies das Battery Shield (K. 1.9.2) im Laufe der Entwicklung einen Defekt auf. Allerdings benötigt das gleiche Modell eine viel zu lange Lieferzeit und es hat sich herausgestellt, dass der versprochene Strom des Battery Shields (3A) nicht dem entspricht, was der Hersteller angegeben hat. Weshalb ich mich kurz vor Fertigstellung entschieden habe, die zwei Akkus (K. 1.9.3) in Kombination mit einem LM2596S Netzteil zu nutzen. Dieses ist klein genug, sodass alles in den vorher vorgesehen Platz passt. [30]

Das Netzteil hat nur den Nachteil, dass es nicht über ein Lademanagementsystem verfügt, wie das Battery Shield, weshalb die Akkus zum Laden entnommen werden müssen.

Produktname	LM2596S DC-DC Netzteil
Hersteller	AZDelivery
Anzahl	1
Kaufdatum	14.04.2021
Gesamtpreis	6,79€
Eingangsspannung	4.5 - 35V
Ausgangsspannung	4-34V (Unter Eingangsspannung)
Ausgangsstrom	Normal 2A; kurzzeitig maximal 3A
Besondere Eigenschaften	• Ausgangsspannung einstellbar • Hoher effektiver Ausgangsstrom

[30]

1.10 Einkauf für die Hand

1.10.1 Flex Sensoren

Um die Bewegungen der Hand zu erkennen, befinden sich auf dem Handschuh Flex Sensoren, bzw. Beugungssensoren. Diese verändern ihren Widerstand abhängig davon, wie stark sie gebogen werden. Auf dem Streifen befindet sich eine Graphitschicht. Der Abstand der Graphitkristalle verändert sich bei Beugung, was dazu führt, dass Elektronen schwerer über die Atome geleitet werden können, was in einem erhöhten Widerstand endet. [25], [26]

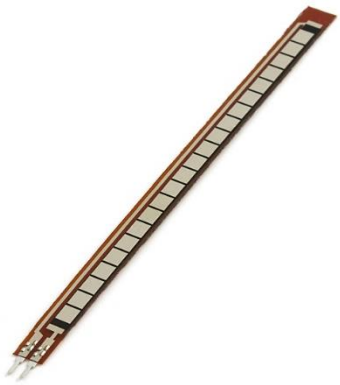

Abbildung 12 [25]

Die Hand benötigt dementsprechend fünf dieser Sensoren. Leider sind die Sensoren schwer zu erhalten, weshalb sie in den USA bestellt werden mussten. Dabei fallen allerdings Zollgebühren an, die dafür sorgen, dass die grundsätzlich teuren Sensoren noch teurer werden.

Produktname	Flex Sensor 4.5"
Hersteller	Sparkfun
Anzahl	5
Kaufdatum	12.10.2020
Gesamtpreis	104,41€ + 27,70€ Zollgebühren = 132,11€
Länge	112,24 mm
Flacher Widerstand	10.000 Ohm
Besondere Eigenschaften	• Hohe Genauigkeit • Bis zu 1 Million Mal biegbar

[25], [26]

1.10.2 Heltec ESP32 HTIT-WB32

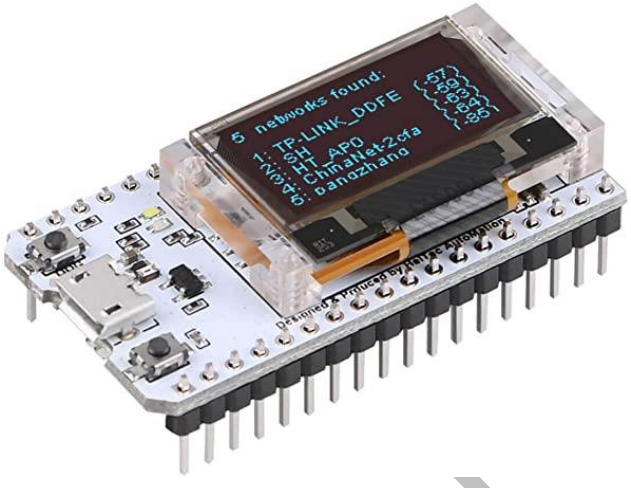

Abbildung 13 [28]

Die Hauptrecheneinheit der Hand ist der ESP32 HTIT-WB32 von Heltec Automation. Er ist ein Arduino Nachbau und verfügt über zahlreiche Besonderheiten, die ihn zu einem perfekten Entwicklungsboard machen. Wie der ESP32 von AZ-Delivery besitzt er einen dual-core 32 bit Prozessor und einen integrierten Controller für Wi-Fi und Bluetooth. Im Prinzip ist er gleich mit dem Board der Prothese. Ein großer Unterschied besteht darin, dass er ein integriertes OLED Display besitzt. Dieses ist hilfreich, um z.B. Daten direkt anzeigen zu lassen. Außerdem muss kein extra Display verkabelt werden und das Board ist somit ziemlich kompakt. Eine weitere Besonderheit dieses Boards ist das integrierte SH1.25-2 Lademanagement. Dies erlaubt das einfache Verbinden eines 3,7V Modellbau-Akkus, der als Stromversorgung dient. Der Akku muss nicht entfernt und aufgeladen werden, dies übernimmt das Board selbst. Die Programmierung über die Arduino IDE macht ihn besonders zugängig. Verschieden Bibliotheken für die Nutzung der WLAN, Bluetooth und Oled-Display Funktionen sind auch verfügbar. Außerdem besitzt er viel mehr Anschlüsse (Pins) als ein normaler Arduino (wie in K. 1.4). [27], [28]

Diese Eigenschaften und vor allem das integrierte Display machen ihn geeignet für die Verwendung auf dem Handschuh, weshalb kein weiterer ESP32 von AZ-Delivery verbaut wird.

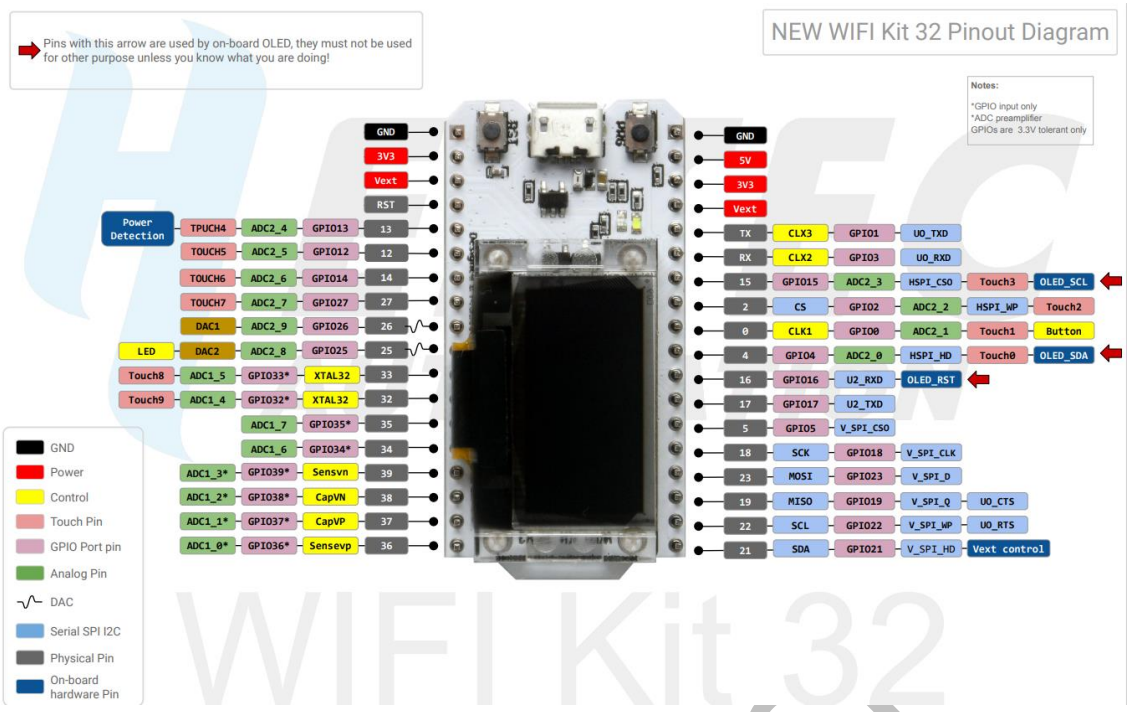

Abbildung 14 [29]

Die Pinbelegung kann aus dem Datenblatt entnommen werden und wird später für die Programmierung benötigt, damit die Pins, an denen die Bauteile angeschlossen sind, richtig adressiert werden.

Produktname	ESP32 HTIT-WB32
Hersteller	Heltec Automation
Verkäufer	MakerHawk
Anzahl	1
Kaufdatum	9.09.2020
Gesamtpreis	17,99€
Eingangsspannung	3,3V oder 5V
Ausgangsspannung	Maximal 5V
Mikroprozessor	Tensilica Xtensa LX6
Prozessor-geschwindigkeit	240 MHz
Wireless Standards	FCC, CE, IC, TELEC, KCC, SRRC, NCC
Bluetooth Protokoll	Bluetooth v4.2, BR/EDR, BLE
Speicher	4mB Flash, 520kB SRAM
Physische Schnittstellen	UART, I2C, SPI, I2S, PWM, SDIO, GPIO, ADC, DAC, GPI
Besondere Eigenschaften	• Alle besonderen Eigenschaften wie der ESP32 von AZ-Delivery • 0,96-inch OLED Display • SH1.25-2 Akkuschnittstelle mit Lademanagement

[27], [28]

1.10.3 Akku

Die Stromquelle des Handschuhs ist ein Standard 1S 3C Lipo Akku. Der Akku verfügt über einen JST 1,25 Stecker, der an das Heltec Board angeschlossen werden kann. Diese Akkus können nur in einem Paket von vier erworben werden. [31]

Abbildung 15 [31]

Produktname	VTC6 Akku
Hersteller	Seamuing
Anzahl	4
Kaufdatum	20.03.2021
Gesamtpreis	21,99€
Eingangsspannung	3,7V
Ausgangsspannung	Maximal 4,2V
Ausgangsstrom	Maximal 1,5 A
Maximale Kapazität	Maximal 1100mAh
Besondere Eigenschaften	• Kleiner Formfaktor • Günstiger Preis • Geeignet für ESP32-Boards

[31]

1.10.4 Golfhandschuh

Der Handschuh ist ein Golfhandschuh. Hier wird die Elektronik und die Beugungssensoren befestigt. Außerdem verleiht der Golfhandschuh dem Projekt einen futuristischen Look.

Produktname	Herren Gt Xtreme Golfhandschuh
Hersteller	Footjoy
Anzahl	1
Kaufdatum	15.10.2020
Gesamtpreis	19,67€

[32]

Abbildung 16 [32]

1.11 Allgemeine Produkte

In diesem Kapitel werde ich grob auflisten, welche weiteren allgemeinen Produkte Ich gekauft habe, bzw. welche die verwendet, allerdings nicht extra erworben werden mussten.

Bezeichnung	**Preis**
Lötzinn, Lötpaste und Isolierband Set [35]	8,99€
Jumperwire und 40pin Stiftleisten [36]	7,99€

Alle in Kapitel 1 nicht aufgelisteten Produkte sind mein Eigentum, das ich unteranderem zur Herstellung benötigte, wie z.B. einen 3D-Drucker, weitere Filamente, Kleber, Nylonseile, Schrauben oder Lötkolben.

1.12 Nicht verwendete Produkte

Im Folgenden Kapitel werden ich die nicht verwendeten Produkte auflisten und erklären, warum sie nicht genutzt wurden, bzw. welche Rolle sie für die Entwicklung spielten

Bezeichnung	**Preis**	**Verwendungszweck**
PrimaCreator PrimaSelect - NylonPower Carbon Fibre [33]	46,49 €	Ursprünglich war geplant alle 3D-gedruckten Teile mit einem Nylon Filament zu drucken, das auch Kohlefaser enthält. Dieses ist um einiges bruchsicherer. Allerdings stellte sich heraus, dass der Drucker, um Nylon zu drucken eine Box benötigt, die dafür sorgt, dass absolut kein Windzug das Modell trifft und die Umgebungstemperatur relativ hoch ist. Die Druckergebnisse waren somit ungenügend und ich musste anderes Filament nutzen.
3 mal AZDelivery HC-05 HC-06 Bluetooth Wireless RF-Transceiver-Modul [34]	16,79€	Erst im späteren Verlauf entschied Ich mich dazu, Wlan als Übertragungsprotokoll zu nutzen und nicht Bluetooth. Diese BL-Module hätten die Verbindung einfacher gemacht, da diese einen eigenen Speicher besitzen und man somit eine feste Kopplung einprogrammieren kann. Wlan ist aber deutlich schneller und benötigt kein extra Modul. Die ESP32-Boards enthalten zwar auch jeweils Bluetooth-Chips, welche allerdings in der Programmierung nicht einfach zu handhaben sind.

Da die Entscheidung diese Produkte nicht zu nutzen erst während des Entwicklungsprozesses standfand und die Produkte schon genutzt wurden, bestand leider keine Möglichkeit mehr einer Rückgabe an den Händler.

1.13 Gesamtkosten

Die Gesamtkosten belaufen sich auf 398,14€ für die genutzten Produkte. Mit den nicht genutzten Produkten steigen die gesamten **Entwicklungskosten auf 461,42€**. Bei der Entwicklungskostenberechnung wurden nur Produkte einberechnet, die extra für das Projekt gekauft werden mussten. Alle weiteren Produkte befinden sich in meinem Besitz und wurden dazu genutzt. Strom- und Wartungskosten wurden auch nicht mit einberechnet.

Würde ich das Projekt nochmal herstellen kann das Produkt mit den noch übrigen Produkten und geschätzten Kosten von 250€ neu gebaut werden.

Die genutzte Software ist kostenlos.

2 Digitale Konstruktion

Nicht nur alle 3D-Druck Objekte müssen vorher digital designt werden. Ich habe mich dazu entschieden, das gesamte Projekt digital zu planen und zu Konstruieren. 3D-Druck dauert lange und viele Bauteile sind ebenso kostspielig, weshalb es sinnvoll ist, alles vorher digital zu konzeptionieren und zu konstruieren. Auch die Elektronik habe ich zuerst mit Fritzing geplant.

2.1 CAD

CAD (von engl. computer-aided design, zu Deutsch rechnerunterstütztes Konstruieren). bezeichnet die Unterstützung von konstruktiven Aufgaben mittels EDV zur Herstellung eines Produkts. [37] Für das Design und das digitale Konstruieren nutze ich Fusion 360, ein für Bildungseinrichtungen kostenloses CAD-Programm. In der Dokumentation verwende ich Konstruktionsbilder und verarbeitete Bilder. Die verarbeiteten Bilder (gerendert) wurden durch den Computer so berechnet, dass sie einem richtigen Produkt ähneln. Außerdem unterscheidet man zwischen perspektivischer und orthogonaler Ansicht.

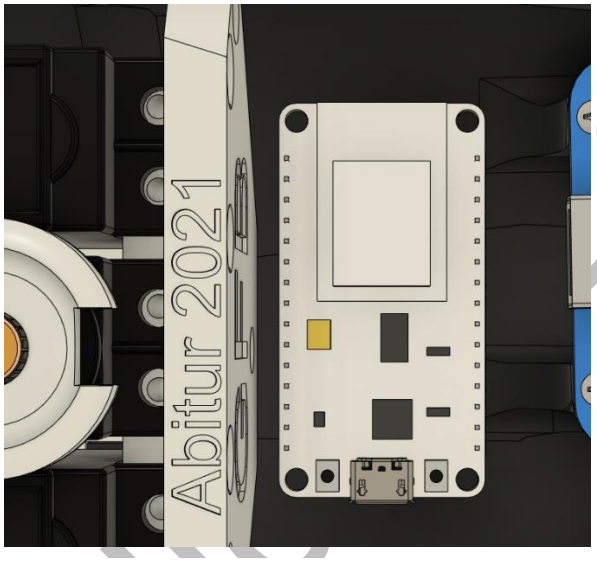

Abbildung 17 perspektivische Ansicht

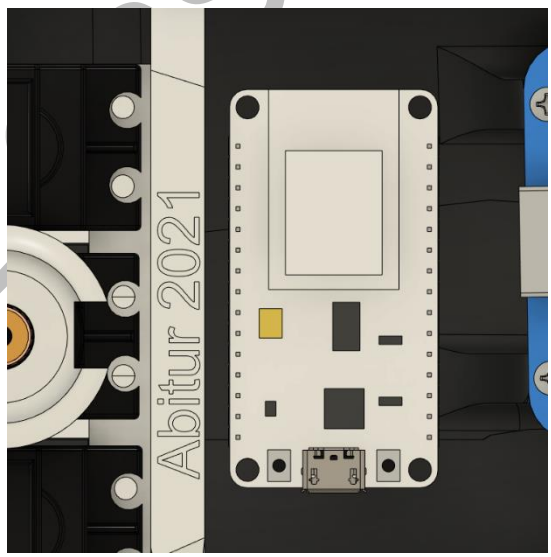

Abbildung 18 orthogonale Ansicht

Beide Bilder wurden in der Konstruktionsoberfläche aufgenommen und besitzen beiden denselben Blickpunkt. Bei der perspektivischen Ansicht, wie in Abbildung 17 werden Objekte so dargestellt, als würde man sie mit dem bloßen Auge beobachten. Dies sorgt für einen realistischen Eindruck. Jedoch nutzt man beim Konstruieren eine orthogonale Ansicht. Bei dieser sind an jedem Punkt am Bildschirm Kanten gleich groß, bzw. eine nahe Kante mit der Länge 10cm wird mit gleich vielen Pixeln am Bildschirm dargestellt wie eine gleichlange Kante, die weiter entfernt ist. Auch kann man nicht auf Seiten sehen, sondern, wie hier nur die Draufansicht. Somit ist die orthogonale Ansicht optimal, um Objekte vor allem passgenau zu modellieren.

CAD-Programme arbeiten mit Vektoren, mathematischen Funktion und Objektdefinitionen, um Objekte darzustellen. Dies bedeutet, dass gerade Kanten

als Vektor, Kurven und Rundungen als mathematische Funktionen angegeben werden. Zeichnet man z.B. einen Kreis wird dieser nicht in viele kurze Vektoren aufgeteilt und so gespeichert, sondern es wird quasi die Information abgespeichert, dass es ein Kreis mit gewissen Eigenschaften (Radius, Position, Winkel relativ zu einem Objekt) ist. Möchte man nun ein Objekt nicht als Projektdatei speichern, sondern als 3D-Modell, muss das Modell in Vektoren umgerechnet werden, sodass es eine große Kompatibilität mit Programmen hat, in denen es weiter genutzt wird. Dad vorgeschlagene Dateiformat von Fusion 360 ist .stl . Bei diesem Format werden nur Vektoren abgespeichert. Drei Vektoren bilden somit immer ein Dreieck. Hierbei spricht man von Facetten. Je genauer man das Modell abgespeichert haben möchte, desto höher muss die Anzahl an Vektoren, bzw. an Facetten sein. Dabei werden die einzelnen Vektoren und Facetten kleiner, wobei sich die Dateigröße erhöht, da mehr Informationen abgespeichert werden. Im gesamten Projekt arbeite ich mit einer maximalen Facettenanzahl, damit die bestmögliche Qualität der Modelle erreicht wird.

2.2 CAD Konstruktion der Prothese

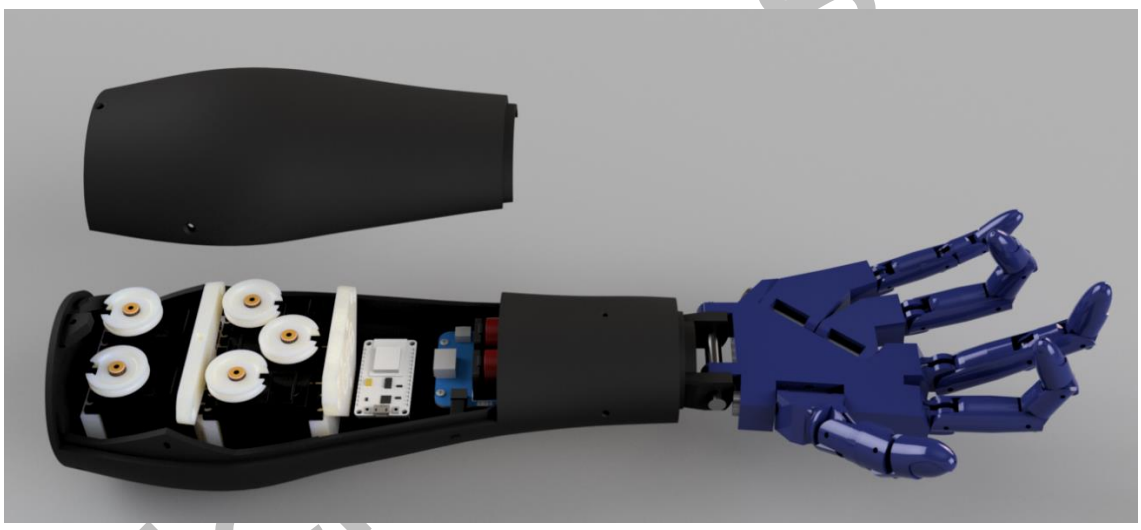

Abbildung 19 gerendert & perspektivisch

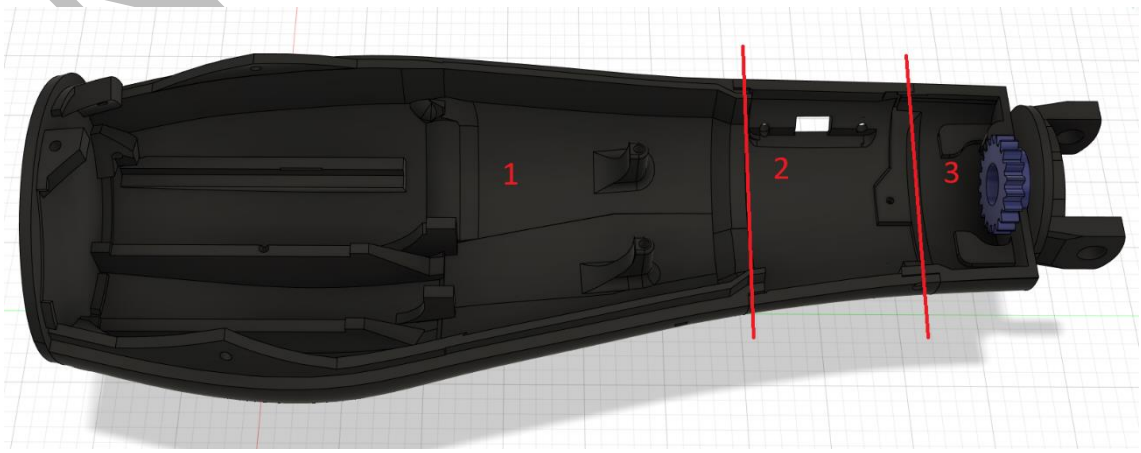

Abbildung 20 Konstruktionsansicht & perspektivisch

Als Grundlage für den Arm dient das in Kapitel 1.3 eingeführte Hand- und Arm Modell vom InMoov-Projekt. Allerdings nutze ich nur die Form und das Prinzip der Sehnen, die dem menschlichen Körper nachempfunden sind. Alle Modelle habe ich selbst nachkonstruiert und mich dazu von InMoov inspirieren lassen. Meine Neuauflage des Designs ist außerdem qualitativ hochwertiger, da ich mit einer hohen Vektorendichte gearbeitet habe. Die von InMoov bereitgestellten Dateien haben eine deutlich geringere Facettenanzahl, was die Qualität mindert. Mein Modell besitzt überall glatte und abgerundete Oberflächen. Somit konnte ich alle Modelle in hoher Qualität abspeichern und später drucken. Man erkennt in allen Dateien und den Drucken keine Facetten mit dem bloßen Auge, da sie dafür zu klein sind.

Begonnen habe ich mit dem Unterarm, der die Elektronik enthält. Zuerst habe ich das Design für die äußere Hülle angefangen. Dabei habe ich mich nur grob am Design von InMoov orientiert. Bzw. die Form eines Armes ist nicht Urheberechtlich geschützt, weshalb ich einen simplen Arm designte, in den die Elektronik passen wird. Hier nutze ich unteranderem Blender, als Modellierungsplattform.

Abbildung 20 zeigt einen Querschnitt des inneren Unterarms. Die Rückenplatte zur linken und die Befestigung für die Hand sowie das blaue Zahnrad zur rechten wurden später hinzugefügt. Somit besteht der Unterarm aus drei größeren Abschnitten. Hier graphisch getrennt durch einen roten Strich. Der Abschnitt eins enthält den Großteil der Elektronik und ist auch das größte Objekt. Auf ihm passt der Servoblock, der alles Servos enthält. Der Servoblock wird durch schrauben befestigt und sitzt dank Schienen gerade und fest im Unterarm. Dies ist wichtig, da er mechanischem Druck ausgesetzt seien wird. Im rechten Bereich von Objekt eins befinden sich zwei Stelzen mit Löchern für Schrauben. Hier wird das Battery Shield befestigt was noch zusätzlich auf zwei Platten rechts auf Objekt drei liegt. In Objekt zwei befinden sich die Batterien und ein Loch für den USB-Zugang. Objekte zwei und drei werden mit Schrauben befestigt, da man diese lösen muss, um Zugang zum Battery Shield zu erhalten. Objekte eins und zwei sind mit Zwei-Komponenten Harzkleber verklebt. Dieser ist sehr stark und kann den mechanischen Kräften standhalten.

Die Rückenplatte wird auch mit Schrauben befestigt. Somit besteht hier die Möglichkeit die Rückenplatte auszutauschen, um eine andere anzuschrauben, die z.B. befestigte Bänder besitzt, damit die Prothese sinngemäß getragen werden kann. Die Standard-Rückenplatte hat allerdings ein flaches Ende, damit die Prothese zu Präsentationszwecken hingestellt werden kann.

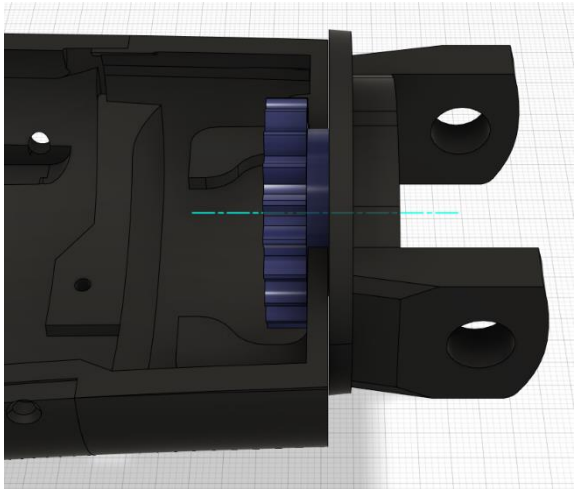

Abbildung 21 Konstruktionsansicht & perspektivisch

An Objekt drei befestigt findet man die Halterung für die Hand (rechts in schwarz) und ein Zahnrad in blau. Diese beiden Teile werden durch Objekt drei gesteckt und miteinander verschraubt. Die türkise Linie zeigt die Rotationsachse der Hand. Es ist wichtig, dass alle Objekte keine Lufträume zwischen einander haben, damit sie ineinander passen und es nicht zu Verklemmungen.

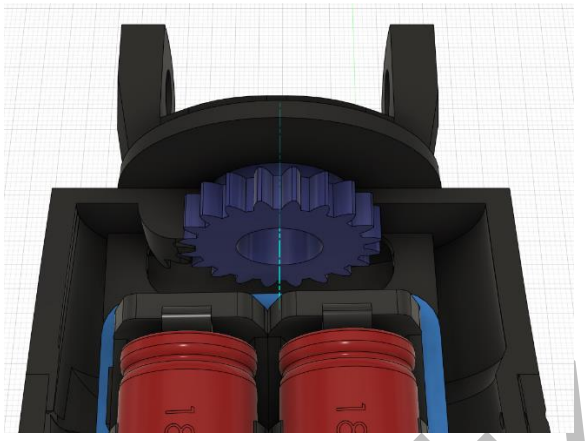

Abbildung 22 Konstruktionsansicht & perspektivisch

Das blaue Zahnrad ist mit einem weiteren starren Zahnrad an Objekt drei verzahnt. Es wurden Zahnräder als Arretierungsmittel genutzt, da somit die Möglichkeit besteht die Hand an der Rotationsachse zu drehen, wenn dies benötigt wird. Allerdings muss dazu Objekt zwei von drei und das blaue Zahnrad von der Handhalterung abgeschraubt werden. In Abbildung 22 kann man schon erkennen, wie das Battery Shield aufliegt.

Alle carbon-schwarzen Objekte wurden mit dem Carbon PLA, wie in Kapitel 1.9.7 beschrieben hergestellt.

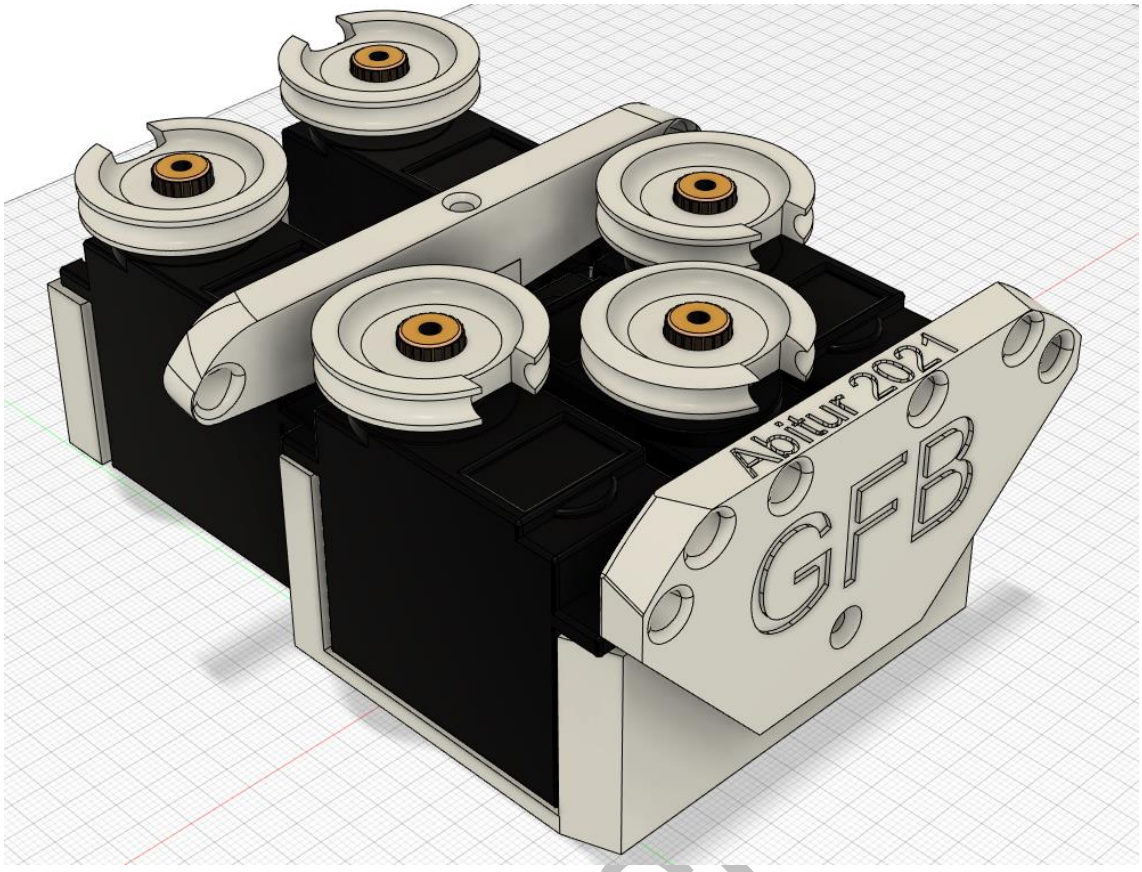

Abbildung 23 Konstruktionsansicht & perspektivisch

Abbildung 23 zeigt den Servoblock. Dieser enthält fünf Servomotoren, die mit dem Block verschraubt sind. Vorne und oben befinden sich zwei zusätzliche Objekte, die auch angeschraubt wurden. Sie dienen dazu die Sehnen, bzw. die Seile zu leiten, damit diese nicht verknoten und zu den Servomotoren geführt werden können. Diese Objekte bieten auch Möglichkeiten der Beschriftung. Auf den Servomotoren befinden sich Führungen für die Sehnen, die hier auch verschraubt werden. Leider ist es nicht wirklich möglich diese Sehnen in CAD nachzubauen, weshalb ich hier auf die Bilder des fertigen Projekts verweise. Alle weißen Objekte wurden mit weißem PLA hergestellt.

In Abbildung 24 erkennt man den Servoblock ohne die Servomotoren. Hier zeigt sich wie die Motoren aufliegen und die Löcher und Schlitze im Gehäuse die dazu da sind, die Kabel der Servomotoren zu leiten. Alles musste extrem passgenau designt und hergestellt werden, weshalb viele Prototypen hergestellt wurden.

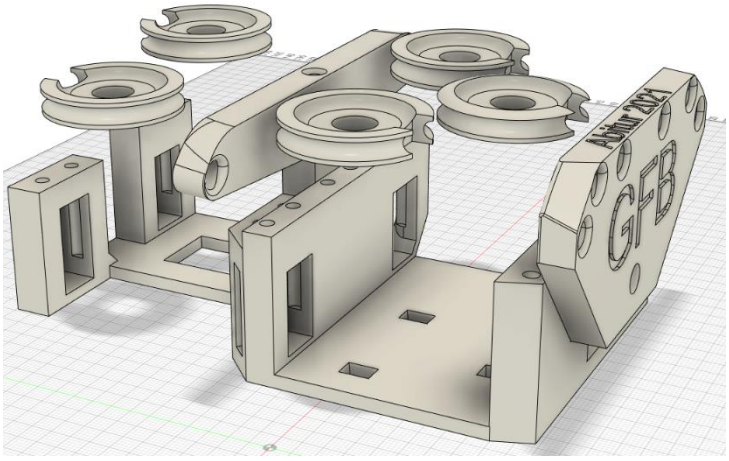

Abbildung 24 Konstruktionsansicht & perspektivisch

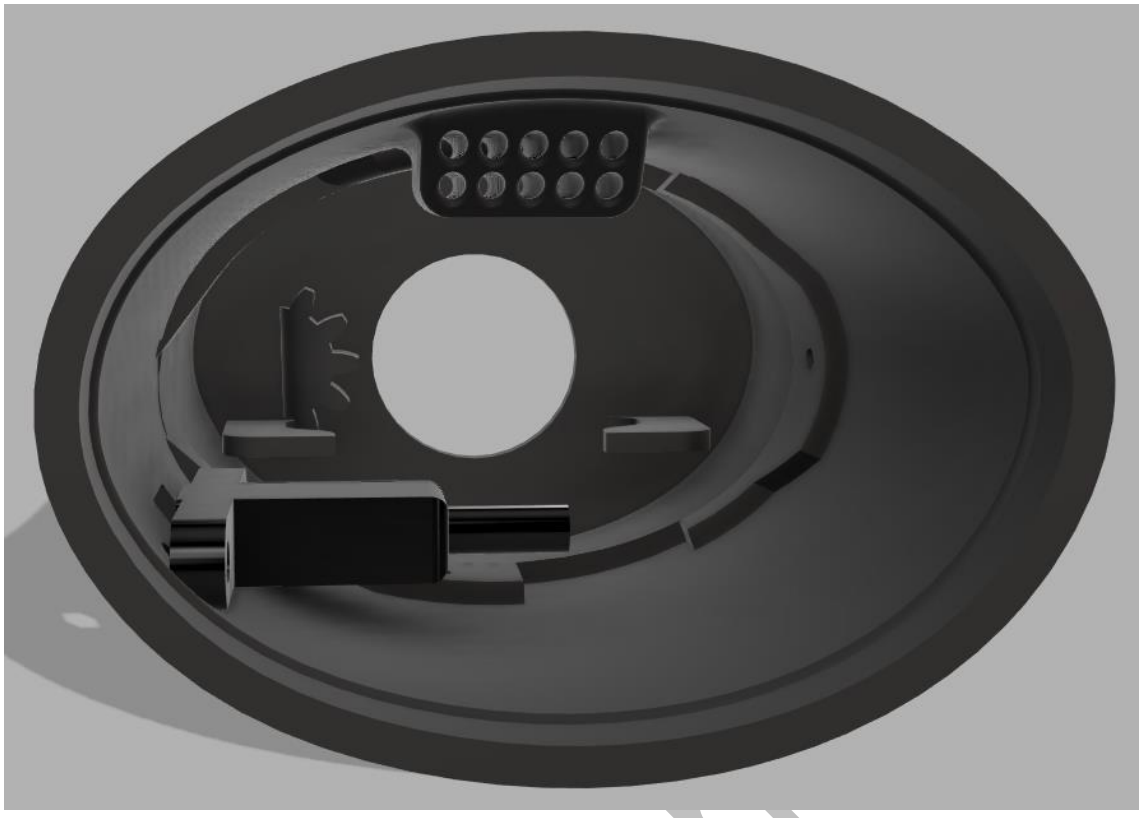

Abbildung 25 Konstruktionsansicht & perspektivisch (ohne kontrastierte Kanten)

Der Innenraum der Objekte zwei und drei aus Abbildung 20 werden in Abbildung 25 dargestellt. Der Querschnitt wurde entfernt. Im oberen Bereich befinden sich Führungsschienen für die Sehnen. Damit diese nicht im ständigen Kontakt mit den Akkus sind, werden die Sehnen über diese Zehn Röhrchen an Objekt zwei geführt. Unten links befindet sich die USB-Buchse, die hier grob modelliert wurde, um zu überprüfen, ob es zu Kollisionen kommt. Die Schwierigkeit hier war, dass der Vorsprung an Objekt zwei nicht zu weit nach unten und die USB-Buchse nicht zu weit nach oben ragt, damit die Akkus passen. Unter dem Battery Shield, bzw. unter der USB-Buchse ist Platz für Kabel.

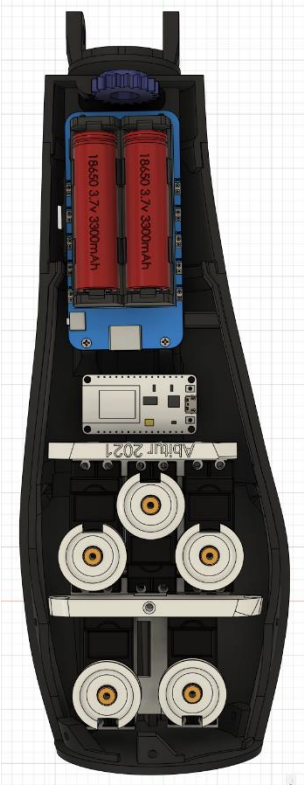

Abbildung 27 ist eine Vergrößerung des Knopfes an der Außenseite des Arms, der durch diesen bis zum Battery Shield reicht und so durch klicken die Elektronik aktiviert.

Abbildung 27 Konstruktionsansicht & perspektivisch

Abbildung 26 zeigt eine weitere Querschnittansicht des Arms, der die Elektronik zeigt

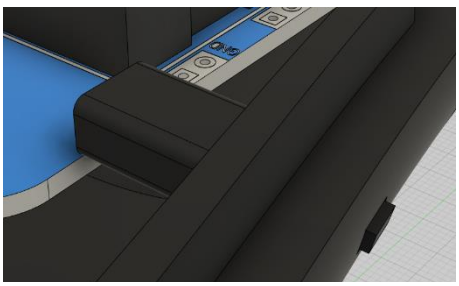

Abbildung 26 Konstruktionsansicht & perspektivisch

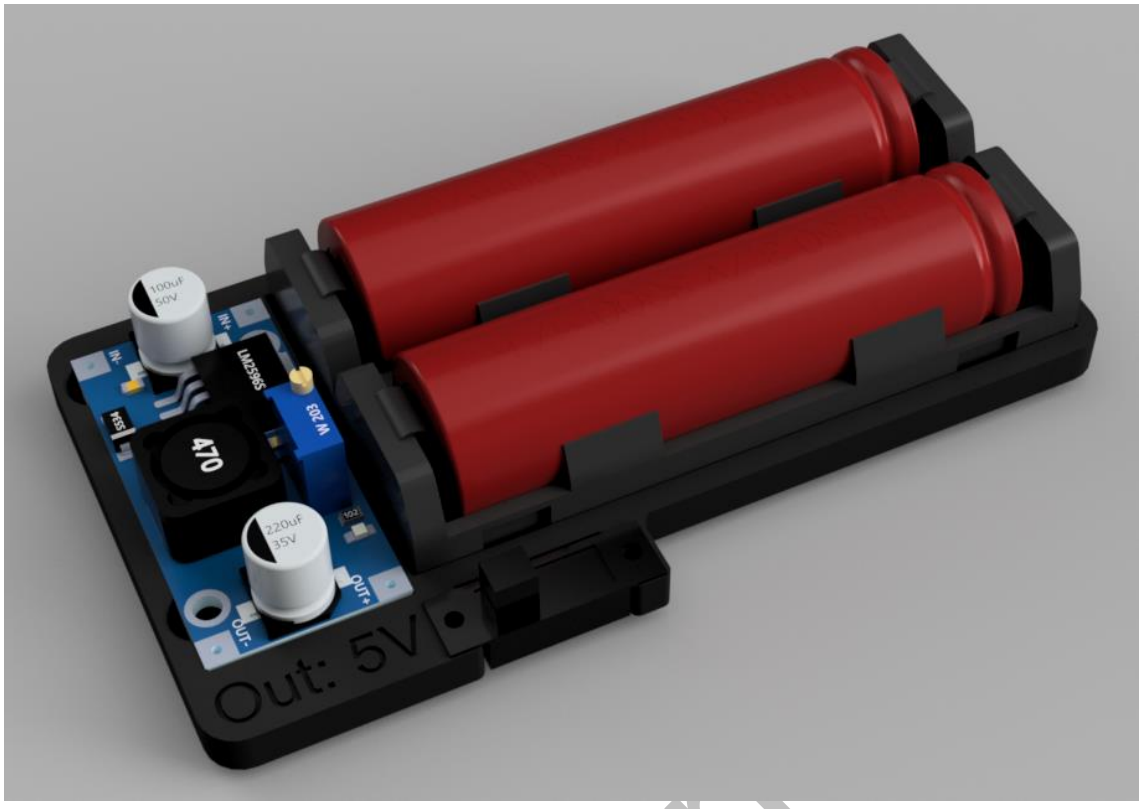

Abbildung 28 gerendert & perspektivisch

Der weitere Entwicklungsverlauf hat gezeigt, dass das ursprüngliche Battery Shield einen defekt aufweist. Außerdem hat es sich herausgestellt, dass der gemessene Strom des Battery Shields nicht dem entspricht, was der Hersteller angegeben hat (3A). Weshalb ich mich kurz vor Fertigstellung entschieden habe, die zwei Akkus (K. 1.9.3) in Kombination mit einem LM2596S Netzteil zu nutzen. Dieses ist klein genug, sodass alles in den vorher vorgesehen Platz passt.

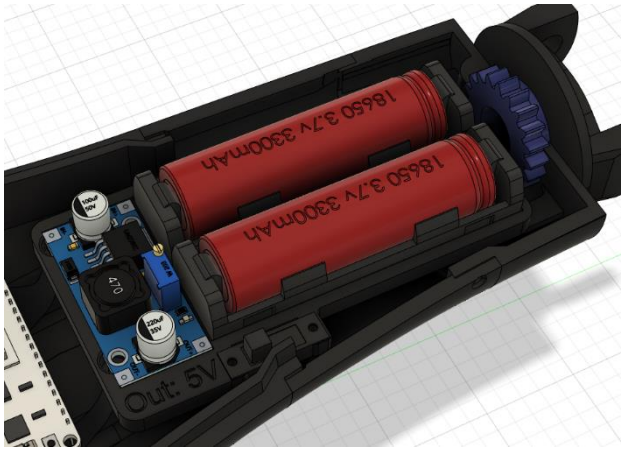

Abbildung 29 Konstruktionsansicht & perspektivisch

Dazu habe ich eine Halterung konstruiert, die in den für das Battery Shield vorhergesehenen Platz passt. Mit einem einfachen Schalter kann die Stromversorgung der Akkus unterbrochen werden und das LM2596S Netzteil erhält keine Spannung und die Prothese ist deaktiviert. Das Design für das Netzteil habe ich nicht selbst nachgebaut, sondern es ist von grabcad.com, einem Anbieter für zahlreiche kostenlose CAD-Modelle [38]. Das neue Akkuboard passt genau in den Platz des ehemaligen Battery Shield, was sehr wichtig ist, da das gesamte Modell mit dem eigentlichen Battery Shield designt und darauf ausgelegt war. Der Knopf, der durch die Schale geht, wurde somit auch entfernt. Leider muss somit die Prothese geöffnet werden, um sie anzuschalten. Wäre dieser defekt nicht, bzw. die unklare Beschreibung des Herstellers, würde das Projekt auch mit

dem ursprünglichen Battery Shield funktionieren. Für das Konzept mit dem neuen Akkuboard habe ich mich erst zum Ende des Projekts entschieden und ist eindeutig nur eine Übergangslösung, bzw. eine die dafür sorgt, dass das Projekt präsentationsfähig ist.

Abbildung 30 Konstruktionsansicht & perspektivisch

Das Akkuboard ist so modelliert, dass alle Objekte genügend Platz haben, bzw. so viel Platz, dass sie eingeklickt werden können. Die Akkus erhalten durch die zwei großen Öffnungen genug Platz, um Wärme an die Umgebungsluft abzugeben. Sie werden festgeklebt und erhalten zusätzlichen Halt durch zwei Nuten am oberen Rand. Unter dem Netzteil sind Löcher im Board, durch die das Board einerseits an Objekt 2 angeschraubt werden kann und andererseits damit sich die Kabel des Netzteils unter dem Board befinden, sodass der Innenraum aufgeräumter ist. Eine kleine Senkung sorgt dafür, dass die Lötstellen unter dem Netzteil genügend Platz haben, damit das Board gerade im Rahmen liegt. Im unteren Bereich befindet sich noch Platz für einen Schalter, der angeschraubt wird. Der Schriftzug „Out: 5V“ signalisiert, dass das Netzteil auf eine Ausgangsspannung von 5 Volt eingestellt ist oder eingestellt werden muss.

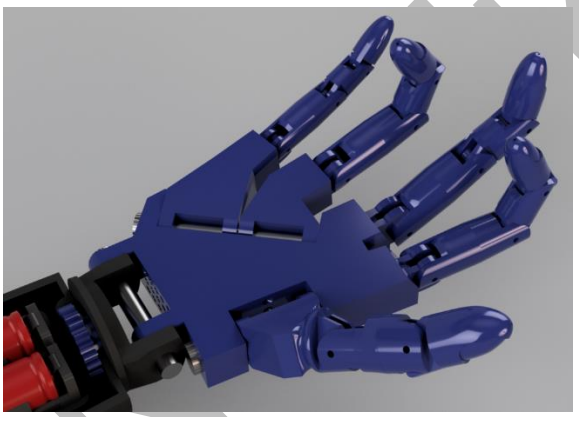
Abbildung 31 gerendert & perspektivisch

Am Modell der Hand habe ich gegenüber dem Modell von InMoov nur einige Abrundungen vorgenommen, um die Facettenanzahl, bzw. die spätere Druckqualität zu erhöhen. Sonst entspricht das Modell und das Prinzip dem von Inmoov.

Die größte, sichtbare Veränderung ist, dass der Schriftzug auf der Handrückenplatte von „InMoov“ durch „GFB“ ersetzt ist. Bei dem Bearbeitungsprozess wurde auch direkt die Facettenanzahl erhöht.

Abbildung 32 gerendert & perspektivisch

Somit ist die digitale Konstruktion der Prothese abgeschlossen. Es müssen 16 einzelne Objekte für den Unterarm und 37 einzelne Objekte für die Hand gedruckt werden, was insgesamt 53 Objekte sind, die alle einzeln gedruckt werden müssen und separate Druckeinstellungen, Filamente usw. benötigen. Außerdem befinden sich fünf Servomotoren, zwei Akkus, ein ESP32-Board und ein LM2596S Netzteil in der Prothese.

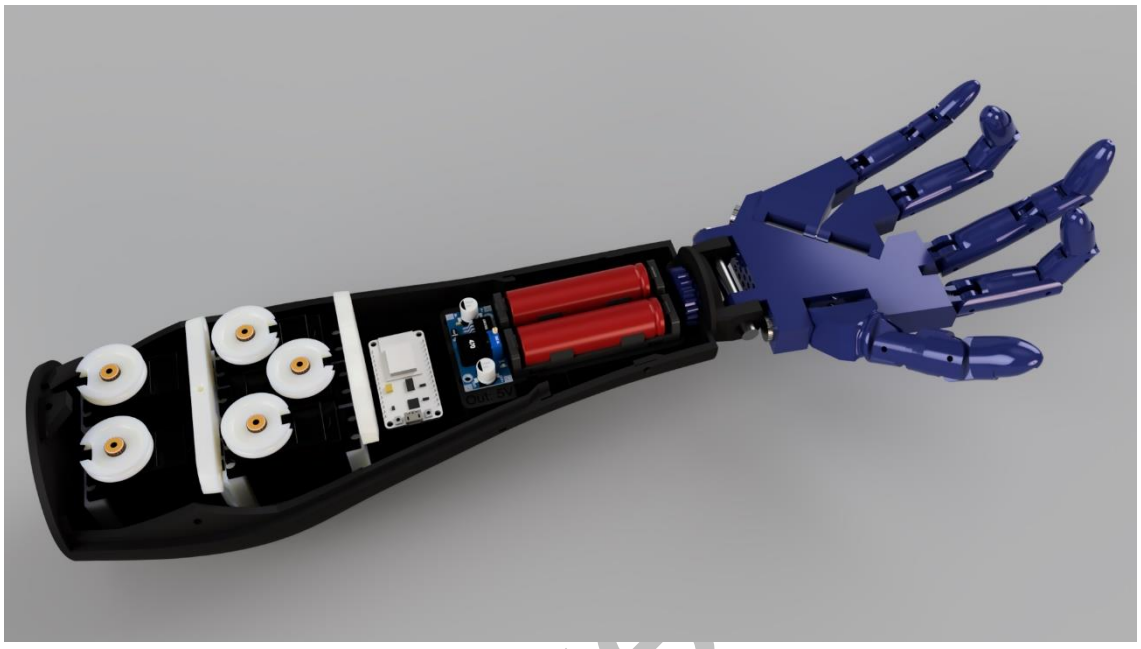

Abbildung 33 gerendert & perspektivisch

2.3 CAD Konstruktion des Handschuhs

Den Handschuh habe ich nicht wirklich mit CAD geplant, allerdings besitzt die Elektronik auch hier ein Gehäuse.

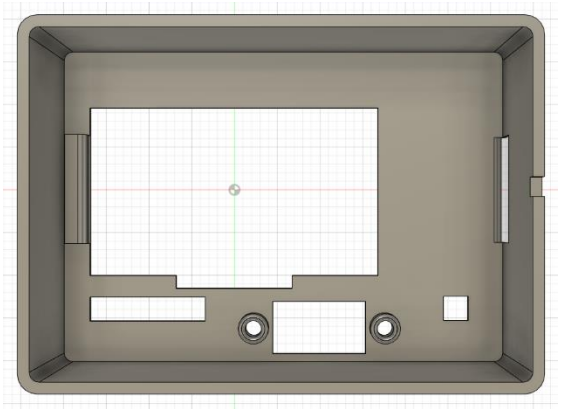

Abbildung 34 Konstruktionsansicht & perspektivisch (Innenansicht)

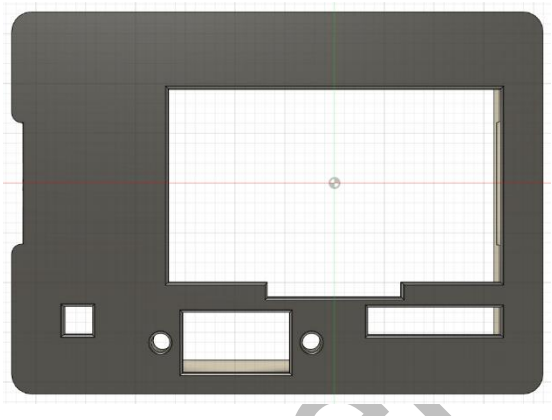

Abbildung 35 Konstruktionsansicht & perspektivisch (Außenansicht)

In Abbildung 34 befindet sich die Innenansicht des Gehäuses. Hier findet der Heltec ESP32 Platz. Eine Aussparung für das Display und das Displaykabel sind vorhanden. Links im Abbildung 34 erkennt man eine zusätzliche Erhöhung. Diese ist dafür da, dass das Board gerade im Gehäuse sitzt. Der längliche Schlitz zur linken und der kleine Kasten zur rechten sind für Jumperwire angedacht. Hier kann man die Beugungssensoren anschließen. Ich habe mich für Jumperwire anstatt einer festen Lötverbindung entschieden, damit es einfacher ist Teile zu wechseln oder falls man nur eine geringere Anzahl an Sensoren benutzen möchte. Mit dieser Lösung ist es außerdem auch einfacher zu experimentieren, wenn Sensoren auch untereinander getauscht werden müssen.

Eine größere Öffnung mit zwei Löchern zu den Seiten dienen für einen Schalter, der dort verschraubt wird und die Stromversorgung unterbricht, damit das Gerät ausgeschaltet werden kann. An der Seite des Gehäuses befindet sich außerdem noch eine Öffnung, um Zugriff zum USB-Ports des Boards zu erhalten, damit auch nach dem Zusammenbau die Programmierung geändert werden kann. Eine kleine Aussparung am Ende des Gehäuses rechts in Abb. 34, bzw. an der oberen Kante in Abb. 36 dient dazu, die Stromkabel des Akkus nach außen zu führen, da dieser von außen an den Handschuh befestigt wird. Ein Deckel verschließt das Gehäuse. Er wird später mit dem Handschuh und dem Gehäuse mit Sekundenkleber verklebt. Ich nutze Sekundenkleber, da dieser nicht so stark ist, wie der andere Zweikomponenten Harzkleber und man das Gehäuse somit im Notfall einfacher wieder öffnen kann. Das Gehäuse hat eine Dicke von 0,8mm an den Seiten und oben und unten von 1mm.

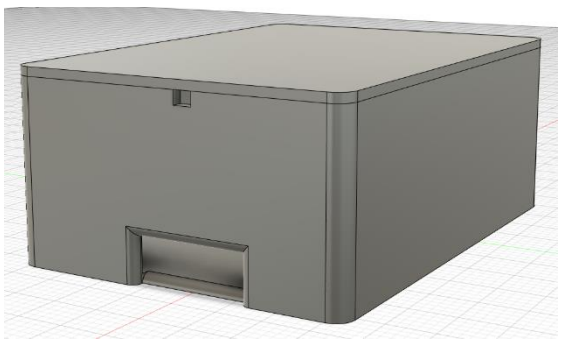

Abbildung 36 Konstruktionsansicht & perspektivisch

2.4 Elektronik

Zur Planung der Elektronik nutze ich das Programm Fritzing. Es ermöglicht aus einer Bibliothek mit zahlreichen elektronischen Komponenten auszuwählen und Schaltungen zu erstellen. [39]

2.4.1 Schaltplan des Handschuhs

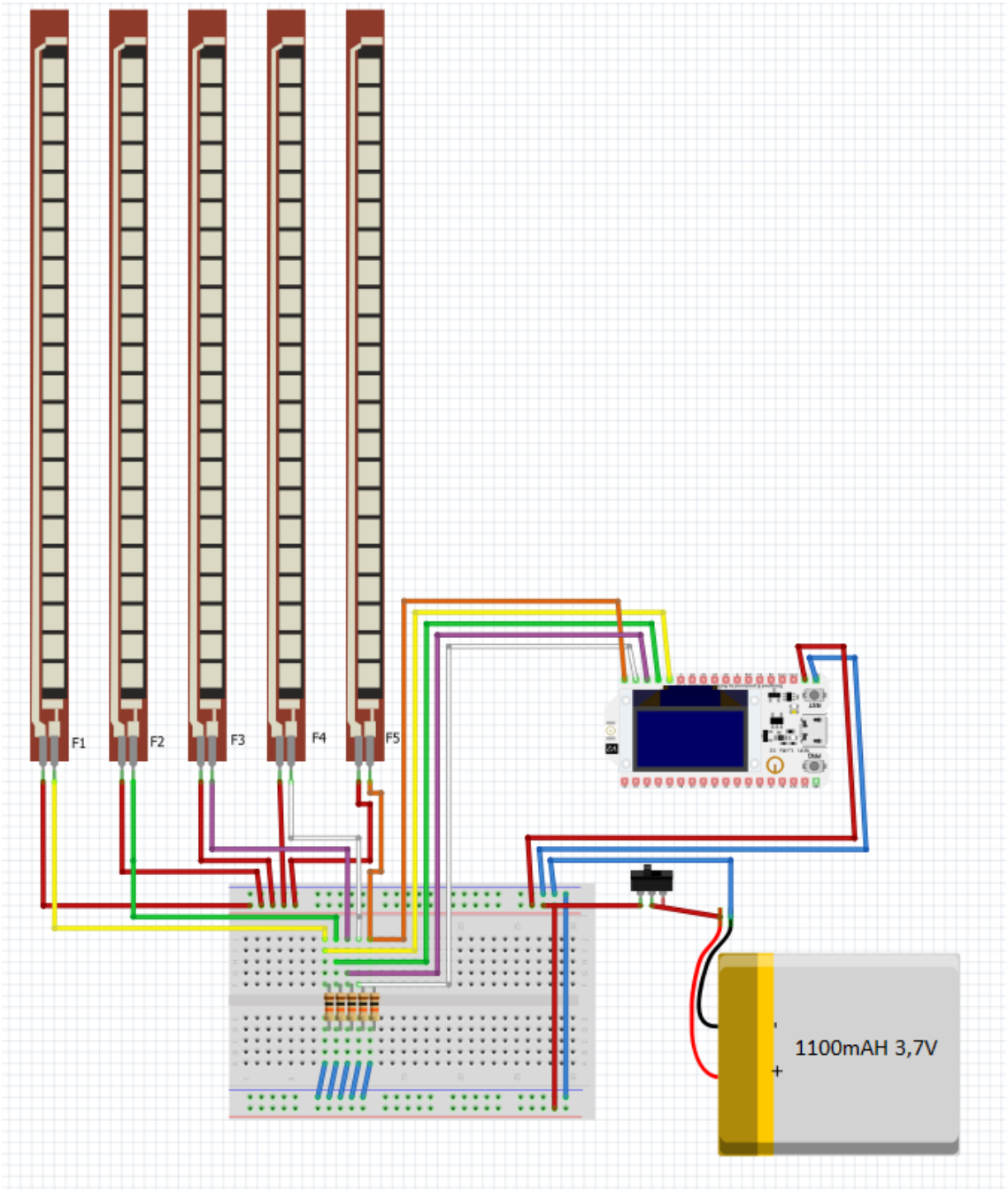

Abbildung 37 Schaltplan des Handschuhs

Da der Schaltplan relativ selbsterklärend ist, erläutere ich ihn nur grob. Die fünf Beugungssensoren sind jeweils mit einem $10k\Omega$ Widerstand verbunden. Der

Akku ist mit dem Pluspol über einen Schalter verbunden, der das System anschaltet. Die Beugungssensoren F1 bis F5 sind wie folgt mit dem ESP32 verbunden.

Finger	F1	F2	F3	F4	F5
Pin	36	37	38	39	34

Sensor F1 entspricht dabei Finger 1, bzw. dem Daumen und F5 Finger 5, bzw. dem kleinen Finger.

2.4.2 Schaltplan der Prothese

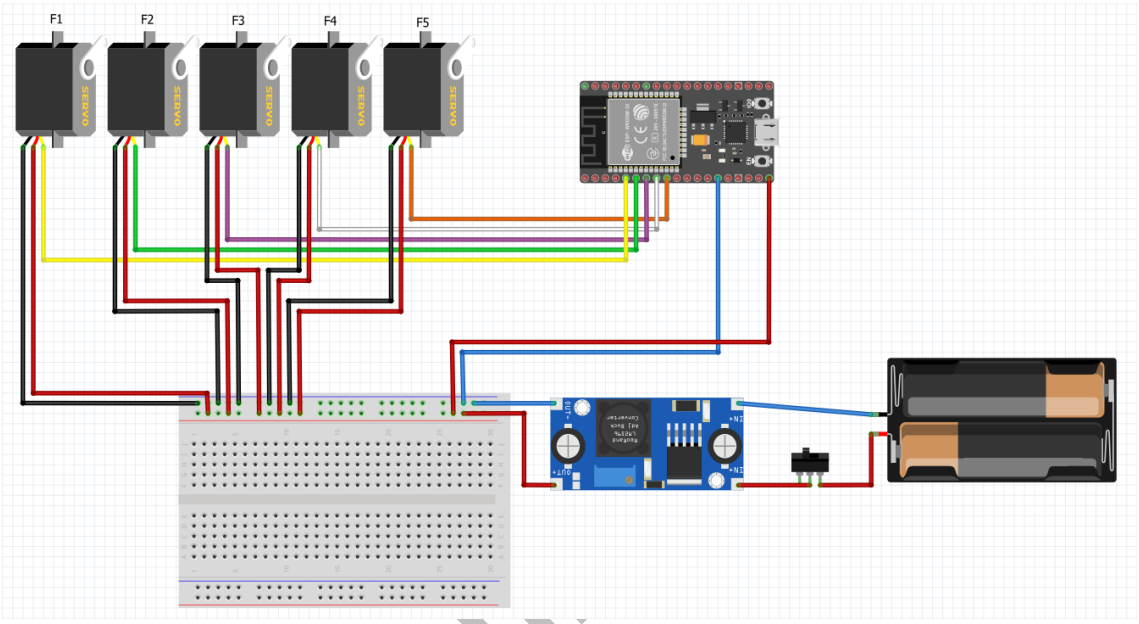

Abbildung 38 Schaltplan der Prothese

Der Schaltplan der Prothese zeigt den neuen veränderten Aufbau mit dem neuen Akkuboard und nicht den mit dem Battery Shield. Die Stromversorgung übernehmen die beiden Akkus, die an das 5V Netzteil mit einem Schalter angeschlossen sind. Das Netzteil versorgt die Prothese mit Strom. Die Servos sind hier schon den Finger F1 bis F5 zugeordnet. Sie bekommen die Spannung auch direkt vom Netzteil und nicht erst über den ESP32, um diesen zu entlasten. Die farbigen Signalkabel sind mit dem ESP32 verbunden. Die Farben sind dieselben wie bei dem Handschuh. Auch im fertigen Projekt werden die Jumperwire des Handschuhs diese Farben haben.

Die Servomotoren F1 bis F5 sind wie folgt mit dem ESP32 verbunden.

Motor	F1	F2	F3	F4	F5
Pin	12	14	27	25	26

Motor F1 entspricht dabei Finger 1, bzw. dem Daumen und F5 Finger 5, bzw. dem kleinen Finger.

Die ESP32 Boards haben unterschiedliche Pinbelegungen (siehe Abb. 6 und Abb. 14). Die aktuelle Pinbelegung ist nach einigem Experimenten die optimale.

3 Physische Konstruktion

Die gesamte Planung der Prothese und des Handschuhs ist nun abgeschlossen und die physische Konstruktion kann beginnen.

3.1 3D-Druck der Modelle

Insgesamt wurden drei Filamente benutzt. Das schon erläuterte Carbon PLA, ein weißes PLA und ein blaues ABS. Die Farben der Objekte im CAD-Modell entsprechen den später verwendeten Filamenten mit den entsprechenden Farben.

Zur Herstellung aller 3D Druck Modelle wurde der Artillery Sidewinder X1 3D Drucker genutzt. Der Drucker ist seit circa einem Jahr in Benutzung und wurde hauptsächlich für das Drucken der Prothesen Teile genutzt. Mit einer UVP von 489,99€ [40] ist er einer der teureren 3D-Drucker, die Ich besitze. Allerdings verfügt er über einen viel größeren Bauraum von 300x300x400 mm als herkömmliche 3D-Drucker. [40] Viele weitere Vorteile, wie der leise Betriebslautstärke und der schnell erhitzbaren 230 V Heizmatte unter dem Printbed. Das ursprüngliche Printbed wurde durch eine 6mm Aluminiumplatte ersetzt, die dafür sorgt, dass das Printbed gleichmäßig erhitzt. Auf dem Aluminium-Printbed wurde zusätzlich eine Pertinax-Platte hinzugefügt. Diese Hartpapierplatte besitzt die Eigenschaft, dass die verschiedensten Filamente hier halt finden und nach Abkühlung einfach abgenommen werden können. Die Platte ist mit Klammern an der Aluminiumplatte befestigt.

Zuerst begann der Druck mit der blauen Hand. Das hier nur leicht bearbeitet Modell wurde in das Slicer-Programm Cura geladen. Dieses Programm wandelt die Vektoren der stl-Datei in Punkte in einem kartesischen Koordinatensystem um und bestimmt, welche Vorgänge der Drucker durchführen soll. Im Cura kann nun eingestellt werden, mit welchen Vorgaben gedruckt werden soll. Welche Einstellungen die optimalen für ein Modell sind, kann nur mit Erfahrung und vielen Testdrucken herausgefunden werden. Die endgültigen Vorgaben für alle Bauteile in blauem ABS sind folgende:

Schichthöhe	0,6mm
Füllung	40%
Dicke der ersten Schicht	0,1mm
Linienbreite	0,25mm
Haftung	An
Wanddicke	0,8mm
Drucktemperatur	234° C
Printbedtemperatur	85° C
Druckgeschwindigkeit (generell)	40 mm/s
Wandgeschwindigkeit	22 mm/s

Alle genannten Werte sind Veränderungen gegenüber den Standardeinstellungen von Cura für die Filamentkategorie „Generic ABS“. Die Gesamtdruckzeit für die Bauteile in blauem ABS beträgt 102 Stunden. Die lange Druckzeit ist begründet mit den feinen Einstellungen, die vorgenommen worden sind. Die Schichthöhe von 0,6mm und die Linien- bzw. Nozzleöffnungsbreite von 0,25mm sind die kleinstmöglichen Einstellungen, die in der höchstmöglichen Qualität enden. Der Druckvorgang erstreckte sich über zwei Wochen, denn viele Objekte mussten mehrmals gedruckt werden, sodass sie keine Fehler mehr aufweisen. Nachts wurde auch gedruckt. Somit war der Drucker nur für die Hand circa 250 Stunden in Gebrauch.

Die Fehldrucke wurden teils aufbewahrt und teils entsorgt, bzw. recycelt. Ein Fehldruck kann z.B. passieren, wenn während eines Druckes plötzlich ein Fenster im Raum geöffnet wird. Die Lufttemperatur sinkt und Verformungen am Modell können entstehen. Deshalb wird normalerweise geraten, dass wenn man mit ABS druckt, der Drucker über ein komplettes Gehäuse verfügt, in dem die Lufttemperatur immer konstant ist und nicht durch einen Windzug beeinflusst werden kann.

Wurden alle Einstellungen getroffen wird eine gcode-Datei generiert, die auf USB-Stick geladen und in den Drucker gesteckt wird.

Der Druck beginnt. Es ist nun sehr wichtig, dass die ersten zwei bis drei Schichten perfekt gedruckt werden. Dies garantiert schon fast, dass das Modell erfolgreich bis zum Ende druckt. An den ersten Schichten kann erkannt werden, ob Fehler aufgetreten sind und, ob andere Einstellungen getroffen werden müssen. Ist z.B. das Printbed zu weit von der Nozzle entfernt, dann findet das frisch erhitzte Plastik keinen Halt. Nur mit einem geschulten Auge können hier kleine Abweichungen erkannt werden, die dazu führen, dass das sich Modell nach einigen Stunden der Herstellung von der Platte löst.

Allerdings gibt es auch Fehlerquellen, die man nicht anhand der ersten Schichten ausfindig machen kann, weshalb der Druck ab und zu kontrolliert werden muss. Hierzu eignet sich z.B. eine Netzwerkkamera. Die Kamera „blink mini“ eignet sich dazu besonders gut, da sie mit ihren 40€ ein perfektes Preis- / Leistungsverhältnis besitzt [41]. Über eine Smartphone-App kann hier die Kamera ständig überwacht werden. In Kombination mit einer Wlan-Steckdose am 3D-Drucker kann dieser im Notfall auch aus der Ferne ausgeschaltet werden.

Der Druck der weißen PLA-Modelle folgte. Alle weißen Modelle können auf Abbildung 24 eingesehen werden. Folgende Einstellungen wurden getroffen:

Schichthöhe	1 mm
Füllung	60%
Dicke der ersten Schicht	0,1mm
Linienbreite	0,4mm
Haftung	aus
Wanddicke	0,8mm

Drucktemperatur	215° C
Printbedtemperatur	60° C
Druckgeschwindigkeit (generell)	50 mm/s
Wandgeschwindigkeit	30 mm/s

Alle genannten Werte sind Veränderungen gegenüber den Standardeinstellungen von Cura für die Filamentkategorie „Generic PLA“.

Da diese Teile mechanischer Kraft ausgesetzt sind, benötigen sie mehr Füllung, um an Stabilität zu gewinnen. Die Schichthöhe ist hier 1mm. Diese könnte sogar noch größer sein, da es im inneren der Prothese nicht nötig ist, feine Strukturen zu drucken. Wenn es jedoch möglich ist, feiner zu drucken, dann ist dies durchaus sinnvoll. Der Druck lief über eine Nacht und dauerte circa. 11 Stunden. Die erhöhte Druckgeschwindigkeit und die niedrigere Linienbreite machte diese schnelle Fertigungszeit möglich. Die weiße Farbe habe ich gewählt, um Kontrast im inneren der Prothese zu erhalten. Nicht nur ist es einfacher bei der CAD- und physischen Konstruktion, sondern die Farbe bringt einen sinnvollen Kontrast bei der Präsentation, um erkennbar zu machen, welche Objekte zusammenhängen.

Alle carbon-schwarzen Teile werden später sichtbar sein, weshalb sie detaillierter gedruckt werden müssen. Alle bis auf die Handhalterung sind nur wenig mechanischen Kräften ausgesetzt, weshalb für diese folgende Einstellungen gelten:

Schichthöhe	0,6mm
Füllung	30%
Dicke der ersten Schicht	0,1mm
Linienbreite	0,4mm
Haftung	An
Wanddicke	0,8mm
Drucktemperatur	225° C
Printbedtemperatur	70° C
Druckgeschwindigkeit (generell)	40 mm/s
Wandgeschwindigkeit	20 mm/s

Alle genannten Werte sind Veränderungen gegenüber den Standardeinstellungen von Cura für die Filamentkategorie „Generic PLA“. Für den Druck mit Carbon PLA musste der Printhead umgebaut werden und eine Edelstahl Nozzle eingebaut werden (siehe K. 1.9.8). Die Modelle sind so konstruiert, dass sie keine Stützstrukturen benötigen. Die Einstellungen für den Druck der Handhalterung ändern sich nur in der Füllung. Diese beträgt 100%, da dieses Teil großer mechanischer Kraft ausgesetzt ist.Das neue Akkuboard, die Stangen für die Hand, sowie das Gehäuse der Elektronik des Handschuhs wurden auch mit dem Carbon PLA gedruckt. Hier wurden dieselben Einstellungen wie oben genannt genutzt. Im Gegensatz zum Servoblock, habe ich mich beim Akkuboard für carbon-schwarz anstatt weiß entschieden, da das schwarze Design besser zum blauen Netzteil und den grünen Akkus passt. Die Druckzeit hier beläuft sich auf über 150 Stunden inklusive Fehldrucke.

3.2 Zusammenbau der Prothese

Nachdem alle Objekte ausgedruckt sind, müssen diese mit Messer und Feile entgratet werden. Alle unerwünschten Fehler werden behoben. Fäden, die entstehen, wenn der Druckkopf sich von einer Stelle wegbewegt, müssen mit einem Feuerzeug abgeschmolzen werden. Die Löcher für die Schrauben müssen mit der entsprechenden Schraube vorgebohrt werden.

Zuerst wurde der Servoblock zusammengebaut. Die Servos werden eingesetzt und die Kabel in die vorgesehenen Schlitze geführt.

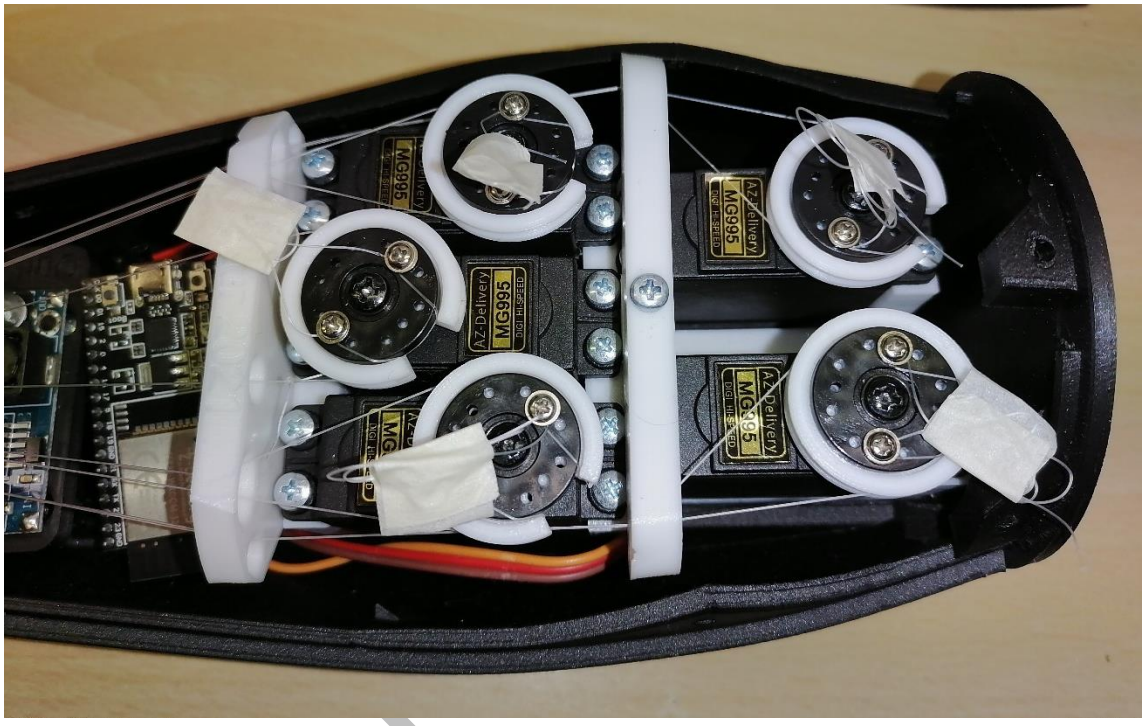

Abbildung 39

Zum festschrauben der Objekte wurden im Inneren M 3x10mm verzinkte Kreuzschlitzschrauben nach DIN 7985 genutzt. Nachdem der Servoblock auf die Prothese, bzw. auf Objekt eins festgeschraubt ist, können auch die Servos festgeschraubt werden. Danach wird die vordere weiße Platte und der mittlere Bügel festgeschraubt. Wie man hier schon erkennen kann, sind die Sehnen, die erst als letztes eingespannt werden, an Scheiben befestigt, die in den Servomotor geschraubt sind. Die Sehneneinspannung erfolgt erst als letztes, nachdem die Hand zusammengebaut ist.

Die Bauteile der Hand werden miteinander mit 2-Komponenten Epoxidharzkleber von UHU verklebt. Ein Fingerknöchel besteht dabei aus zwei Objekten. 0,3mm dicke Nylonfäden werden durch die Finger und die Handplatte geschoben. Dabei müssen beide Enden gleich lang sein. Die Sehnen sind circa 15cm länger als die Prothese, damit Spielraum für das spätere einspannen vorhanden ist. Die schwarzen Stangen, die den Daumen, Ringfinger und kleinen Finger an der Handplatte befestigen wurden vorher mit 100er Schleifpapier bearbeitet, damit es nicht zu starker Reibung kommt und die Bauteile nicht verkanten. Die

Fingerknöchel sind verbunden durch kleine Filamentestücke, die durch die Löcher zwischen den Knöcheln geschoben werden. An den Enden werden sie mit den Knöcheln verlötet.

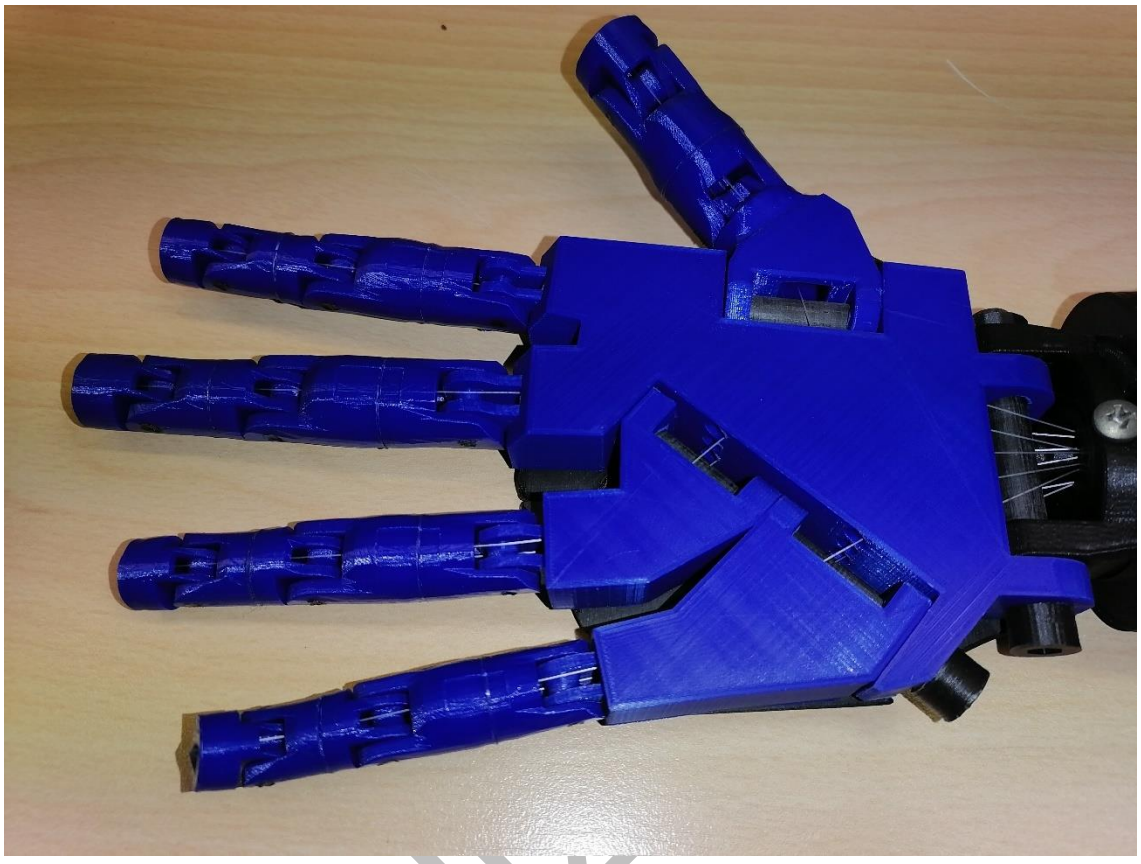

Abbildung 40

Die Nylonfäden werden an den Fingerspitzen mit dem Finger mit dem gleichen Kleber, wie für die Knöchel verklebt. In Abbildung 40 fehlen nun nur noch die Fingerkuppen, die als aller letztes angebracht werden, sodass notfalls noch die Möglichkeit besteht, Sehnen auszuwechseln.

Die Sehnen werden weiter zu den Zehn Röhrchen geführt und mit einer Pinzette hindurch geschoben. Das Layout der Röhrchen ist:

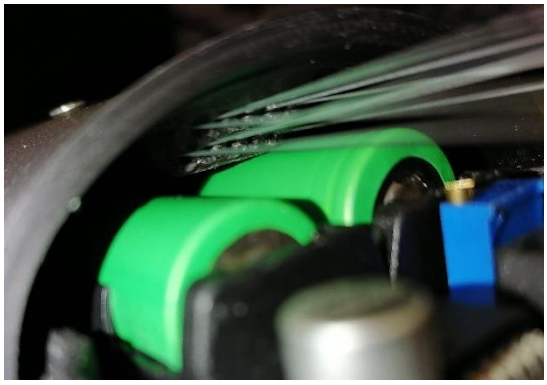

Abbildung 41

5 zu	4 zu	3 zu	2 zu	1 zu
5 auf	4 auf	3 auf	2 auf	1 auf

Die Nummer steht für den jeweiligen Finder. „zu“ bedeutet, dass der Finger sich schließt und in einen geschlossenen Zustand geht, wenn die entsprechende Sehne gezogen wird. „auf“ ist dementsprechend die Sehne, die bei Zug den Finger in einen offenen Zustand versetzt.

Abbildung 42

Als nächsten Schritt wird Objekt eins mit zwei mit dem Epoxidharzkleber verklebt. Das Akkushield wird nach dem Schaltplan aus Abbildung 38 verkabelt und verlötet und durch die Öffnung zwischen Objekt zwei und drei in Objekt zwei geschoben. Objekte zwei und drei werden mit drei PZ1 3x12mm verzinkten Spanplattenschrauben mit Senkkopf verschraubt. Das Akkushield wird mit den M3 Schrauben an Objekt eins geschraubt.

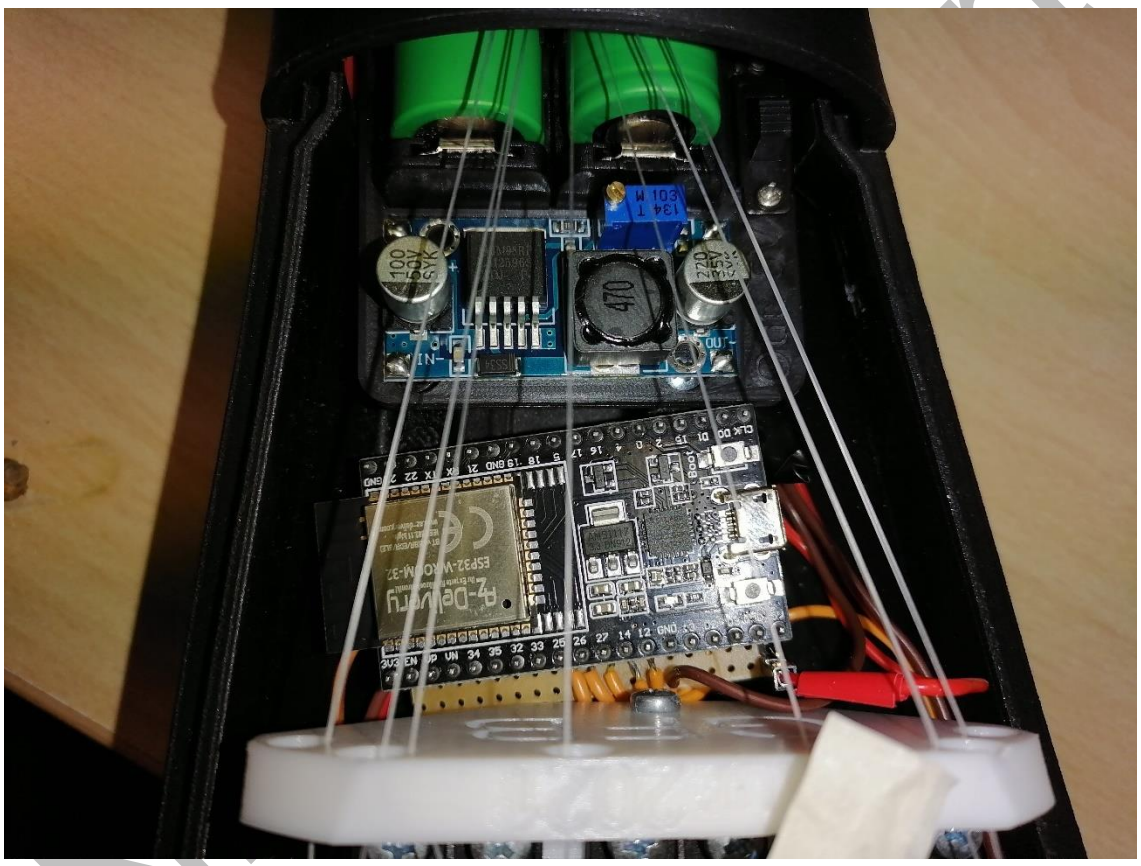

Abbildung 43

Der ESP32 wird gemäß Schaltplan aus Abbildung 38 auf eine Leiterplatine verlötet. Der ESP32 sitzt dabei auf Stiftleisten. Dies ermöglicht, dass der ESP32 ausgewechselt werden kann ohne dass er ausgelötet werden muss. Die Sehnen werden durch die vorgesehenen Löcher der Vorderplatte und des Bügels geführt. Das einspannen passiert erst in Kapitel 4.3. Damit die Sehnen nicht verheddern und der Überblick behalten werden kann, sind sie mit Kreppband miteinander verklebt.

3.3 Zusammenbau des Handschuhs

Die Elektronik wird gemäß dem Schaltplan aus Abbildung 37 auf einer Leiterplatine festgelötet. Der ESP32 ist auf Stiftleisten eingesteckt.

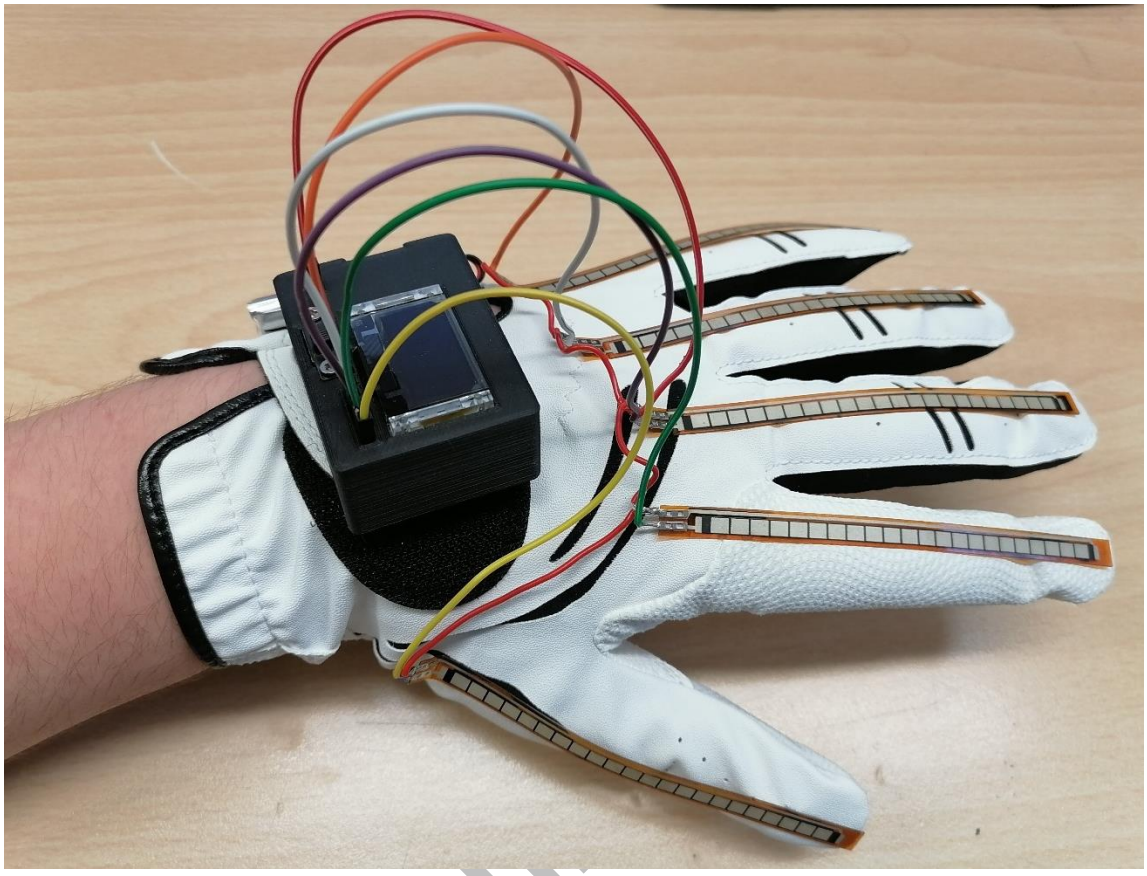

Abbildung 44

Ein Jumperwire ist jeweils an einen Beugungssensor gelötet. Am anderen Pol der Sensoren liegt ein Kabel, das alle Sensoren verbindet. Die Sensoren sind an Jumperwire befestigt, damit sie einfacher ausgetauscht werden können und das Umstecken an der Elektronikbox vereinfacht. Ein festgeschraubter Schalter stoppt die Stromversorgung des Boards. Der 1100mAh Akku ist an der Seite auf einem Klebestreifen befestigt, dies ermöglicht das einfache Tauschen. Auch die Beugungssensoren selbst sind über Klebestreifen an den Fingern befestigt. Das Gehäuse der Elektronik ist mit Sekundenkleber am Handschuh befestigt. Das Akkulademanagement des ESP32 Boards ermöglicht es, dass der Akku einfach über den Micro-USB-Port am Board geladen werden kann. Dazu befindet sich an der Seite eine Öffnung zum USB-Port.

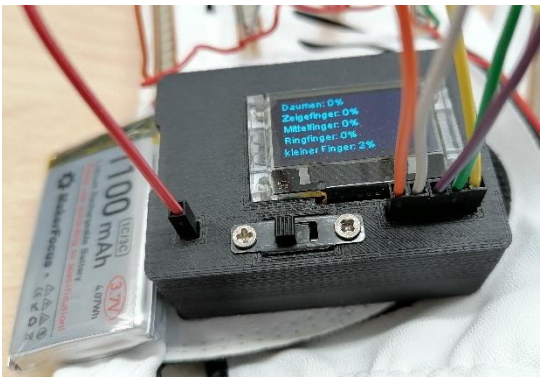

Abbildung 45

4 Programmierung

4.1 Arduino IDE Setup

Bevor die Programmierung begonnen werden kann, muss die Arduino IDE zuerst installiert und konfiguriert werden. Die Software kann unter www.Arduino.cc heruntergeladen werden [42]. Die genutzte Version ist die 1.8.13 für Windows 10.

Damit die beiden ESP32 Boards genutzt werden können, müssen zusätzliche Boardverwalter URLs angegeben werden. Nach der Installation kann unter Datei > Voreinstellungen > Zusätzliche Boardverwalter-URLs die folgenden Links hinzugefügt werden.

https://dl.espressif.com/dl/package_esp32_index.json

https://resource.heltec.cn/download/package_heltec_esp32_index.json

Diese verlinkten json-Dateien verknüpfen die Arduino IDE mit den speziellen Eigenschaften des ESP32 Boards. Die Dateien werden von den Herstellern veröffentlicht und können im Datenblatt eingesehen werden.

Anschließend müssen Bibliotheken installiert werden. Bibliotheken werden genutzt, um bestimmte Funktionen von Boards nutzen zu können. Möchte man z.B. Wifi-Funktion eines ESP32 Boards nutzen, muss die entsprechende Bibliothek installiert sein.

Bibliotheken werden installiert über: Werkzeuge > Bibliotheken verwalten. Es öffnet sich ein Fenster in dem Bibliotheken gesucht werden können. Folgende Bibliotheken wurden gesucht und mit der neusten Version installiert. Bibliotheken sind Rückwärtskompatibel. Selbst eine Version, die in einigen Jahren erscheint, bleibt kompatibel mit der Programmierung von heute.

- ESP_NOW
  - Zur Verwendung von ESP_NOW einem Übertragungsprotokoll für Daten über WiFi
- WiFi
  - Zur Verwendung von WiFi Funktionen
- ESP32SERVO
  - Zur Verwendung von Servomotoren mit dem ESP32
- WIRE
  - Unterstützt WiFi Bibliothek zur Kommunikation zwischen Arduinos
- ARDUINO
  - Standard Arduino Bibliothek
- HELTEC
  - Zur Verwendung des Displays des ESP32 von Heltec auf dem Handschuh

Die Arduino IDE ist jetzt bereit.

4.2 Programmierung des Handschuhs

Trotz, dass die Prothese zuerst fertiggestellt war, begann ich als erster mit der Programmierung des Handschuhs. Ursprünglich war es geplant die Verbindung mit Bluetooth herzustellen. Allerdings ist das Bluetooth-Protokoll im Vergleich zu WiFi viel langsamer. Außerdem wird für eine vernünftige Bluteooth-Verbindung ein weiterer externer Chip benötigt. Somit ist eine WiFi-Verbindung hier platzsparender und sinnvoller.

Für die Verbindung müssen die MAC-Adressen der ESP32 Chips festgestellt werden. Dafür wird auf beide Chips folgender Code geladen. Kommentare stehen nach zwei Schrägstrichen.

```
1  //Einfügen der WiFi Bibliothek
2  #include "WiFi.h"
3
4  void setup(){
5    //serielle Verbindung zum Computer wird gestartet
6    Serial.begin(115200);
7    //WiFi wird gestartet
8    WiFi.mode(WIFI_MODE_STA);
9    //Die MAC-Adresse wird über den seriellen Monitor des
10 Computers
11   //ausgegeben
12   Serial.println(WiFi.macAddress());
13 }
14 void loop(){
15 }
```

Die MAC-Adresse wird ausgegeben. Das Ergebnis lautet:

ESP32 MAC-Adresse Prothese: `3C:61:05:3F:B9:B8`

ESP32 MAC-Adresse Handschuh: `C4:4F:33:76:E6:8D`

Es folgt die fertige Programierung des Handschuhs. Es ist zu bemerken, dass die Entwicklung dieses Codes über Wochen andauerte und Zwischenversionen nicht extra genannt werden.

```
1  //ESP32 MAC-Adresse Prothese: 3C:61:05:3F:B9:B8
2  //ESP32 MAC-Adresse Handschuh: C4:4F:33:76:E6:8D
3
4  //Bibliotheken werden hinzugefügt
5  #include "Arduino.h"
6  #include "heltec.h"
7  #include <esp_now.h>
8  #include <WiFi.h>
9  #include <Wire.h>
10
11 //ESP32 MAC-Adresse Prothese: 3C:61:05:3F:B9:B8
```

```
12  uint8_t broadcastAddress[] = {0x3C, 0x61, 0x05, 0x3F, 0xB9, 0xB8};
13
14  //Pins für die Beugungssensoren werden definiert
15  const int F1_Pin = 36;  //Daumen
16  const int F2_Pin = 37;  //Zeigefinger
17  const int F3_Pin = 38;  //Mittelfinger
18  const int F4_Pin = 39;  //Ringfinger
19  const int F5_Pin = 34;  //kleiner Finger
20
21  // 5-Dimensionales Array um später konvertierte Daten zu speichern
22  int converted_flex[5];
23
24  // Variable, um abzuspeichern, ob das Senden erfolgreich war
25  String success;
26
27  //Struktur für zu sendende Daten (Muss gleich mit Empfänger sein)
28  typedef struct struct_message {
29    int send_F1;
30    int send_F2;
31    int send_F3;
32    int send_F4;
33    int send_F5;
34  } struct_message;
35
36  // Erstellt ein struct_message Objekt, um ausgehende Daten zu
37  //speichern
38  struct_message datatosend;
39
40  // Callback-Funktion, wenn Daten gesendet werden
41  void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status)
42  {
43    Serial.println();
44    if (status ==0){
45      success = "Delivery Success :)";
46    }
47    else{
48      success = "Delivery Fail :(";
49    }
50  }
51
52  void setup() {
53    //Serial Monitor wird gestartet (Nur für Debugging)
54    Serial.begin(115200);
55
56    //WiFi wird im Modus WIFI_STA gestartet
57    WiFi.mode(WIFI_STA);
58
59    //Display wird initialisiert
60    Heltec.begin(true /*DisplayEnable Enable*/, false /*LoRa Disable*/,
61  true /*Serial Enable*/);
62    Heltec.display->setContrast(255);
63    Heltec.display->clear();
64
65    Heltec.display->drawString(0, 0, "Handschuh gestartet \n Verbindung
66  wird \n hergestellt.");
```

```
67      Heltec.display->display();
68
69      //ESP-NOW wird initialisiert
70      if (esp_now_init() != ESP_OK) {
71        Heltec.display->drawString(0, 0, "Die Verbindung mit \n der
72    Prothese konnte \n nicht hergestellt \n werden!");
73        Heltec.display->display();
74        return;
75      }
76
77      //Registrierung einer Callback-Funktion, die aufgerufen wird, wenn
78      // Daten gesendet werden
79      esp_now_register_send_cb(OnDataSent);
80
81      //Peer wird registiert
82      esp_now_peer_info_t peerInfo;
83      memcpy(peerInfo.peer_addr, broadcastAddress, 6);
84      peerInfo.channel = 0;
85      peerInfo.encrypt = false;
86
87      //Peer wird hinzugefügt
88      if (esp_now_add_peer(&peerInfo) != ESP_OK){
89        Serial.println("Failed to add peer");
90        return;
91      }
92
93      Heltec.display->drawString(0, 0, "Handschuh initialisiert \n und
94      Verbindung \n hergestellt.");
95      Heltec.display->display();
96      //Pins für Beugungssensoren werden initialisiert
97        pinMode(F1_Pin, INPUT);
98        pinMode(F2_Pin, INPUT);
99        pinMode(F3_Pin, INPUT);
100       pinMode(F4_Pin, INPUT);
101       pinMode(F5_Pin, INPUT);
102
103   }
104
105   void loop() {
106     Heltec.display->clear();
107
108     //Daten der Beugungssensoren werden gelesen und in flex_read_NUMMER
109     // gespeichert
110     int flex_read_1 = analogRead(F1_Pin);
111     int flex_read_2 = analogRead(F2_Pin);
112     int flex_read_3 = analogRead(F3_Pin);
113     int flex_read_4 = analogRead(F4_Pin);
114     int flex_read_5 = analogRead(F5_Pin);
115
116     //Rohdaten des Bewegungssensors werden in grobe Werte von 0 bis 100
117     // umgewandelt
118     converted_flex[0] = 100 - ((flex_read_1 - 1050) / 5.5);
119     converted_flex[1] = 100 - ((flex_read_2 - 1000) / 6);
120     converted_flex[2] = 100 - ((flex_read_3 - 800) / 8.5);
121     converted_flex[3] = 100 - ((flex_read_4 - 1100) / 6.5);
```

```
122     converted_flex[4] = 100 - ((flex_read_5 - 800) / 9);
123
124     //Da Werte unter 0 und über 100 enstehen können, werden sie
125     // bereinigt
126
127     for (int i=0; i <= 5; i++){
128       if(converted_flex[i] < 0){
129         converted_flex[i] = 0;
130       }
131       if(converted_flex[i] > 100){
132         converted_flex[i] = 100;
133       }
134     }
135
136     //Zu sendende Daten werden in datatosend verpackt
137     datatosend.send_F1 = converted_flex[0];
138     datatosend.send_F2 = converted_flex[1];
139     datatosend.send_F3 = converted_flex[2];
140     datatosend.send_F4 = converted_flex[3];
141     datatosend.send_F5 = converted_flex[4];
142
143     //Sende Daten
144     esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *)
145     &datatosend, sizeof(datatosend));
146
147     //Für Debugging
148     if (result == ESP_OK) {
149       Serial.println("Sent with success");
150     }
151     else {
152       Serial.println("Error sending the data");
153     }
154     //Ausgabe der Werte am Display
155     Heltec.display->drawString(0, 0, "Daumen: " +
156     String(converted_flex[0])+ "%\n" + "Zeigefinger: " +
157     String(converted_flex[1])+ "%\n" + "Mittelfinger: " +
158     String(converted_flex[2])+ "%\n" +"Ringfinger: " +
159     String(converted_flex[3])+ "%\n" +"kleiner Finger: " +
160     String(converted_flex[4])+ "%");
161     Heltec.display->display();
162
163     //Daten werden im Intervall von 50 ms gelesen, umgewandelt und
164     // gesendet. Das Display wird somit auch jeden 50ms aktualisiert.
165     delay(50);
166   }
```

4.3 Programmierung der Prothese

Die Programmierung der Prothese ist simpler, da die meisten Rechenoperationen auf dem ESP32 des Handschuhs durchgeführt werden. Es fällt auf, dass viele Abschnitte ähnlich sind, dies liegt daran, dass die Verbindung über dasselbe Protokoll geschieht und somit auch selbe Codeabschnitte benötigt werden.

Um herauszufinden, welche Werte für die Servomotoren der Finger benötigt werden muss vor Beginn ein Kalibrierungscode erstellt und angewendet werden:

```
1  #include <ESP32Servo.h>
2
3  Servo servo1;
4  int minUs = 500;
5  int maxUs = 2500;
6
7  int servo1Pin = 12;
8  int servo2Pin = 14;
9  int servo3Pin = 27;
10 int servo4Pin = 25;
11 int servo5Pin = 26;
12
13 ESP32PWM pwm;
14 void setup() {
15   ESP32PWM::allocateTimer(0);
16   Serial.begin(115200);
17   servo1.setPeriodHertz(50);
18 }
19
20 void loop() {
21   servo1.attach(servo1Pin, minUs, maxUs);
22   servo1.write(0);
23   delay(1000);
24   servo1.write(60);
25   delay(3000);
26 }
```

Dieser Code wird ausgeführt. In diesem Zustand wird Servo 1 zuerst auf Position 0 und nach 1000ms auf Position 60 gestellt. Nach 3000ms wiederholt sich der Vorgang. Hier muss mit dem zweiten Positionswert in Zeile 24 experimentiert werden. In Position 0 können nun die Sehnen mit Schrauben eingespannt werden. Dreht sich nun der Servo auf Position 60 bewegt sich der Finger. Reicht die Beugung nicht muss mit dem Wert weiter experimentiert werden.Ist die Kalibrierung abgeschlossen erhält man folgende Winkel der Servomotoren für die jeweiligen Finger und Zustände:

	Finger 1	Finger 2	Finger 3	Finger 4	Finger 5
Winkel (auf)	60	90	90	100	80
Winkel (zu)	0	0	0	0	0

Die Programmierung der Prothese beginnt:

```
1  //ESP32 MAC-Adresse Prothese (Empfänger): 3C:61:05:3F:B9:B8
2  //ESP32 MAC-Adresse Handschuh (Sender): C4:4F:33:76:E6:8D
3  //Einfügen der Bibliotheken
4  #include <esp_now.h>
5  #include <WiFi.h>
6  #include <ESP32Servo.h>
7  #include <Wire.h>
8
9  //Erstellen von Servo Objekten. Servo1 für den Daumen usw.
10 Servo servo1;
11 Servo servo2;
12 Servo servo3;
13 Servo servo4;
14 Servo servo5;
15
16 //minimale und maximale Pulsweitenlänge der Servos //(Datenblatt)
17 int minUs = 500;
18 int maxUs = 2500;
19
20 //Pins der Servos werden festgelegt
21 int servo1Pin = 12;
22 int servo2Pin = 14;
23 int servo3Pin = 27;
24 int servo4Pin = 25;
25 int servo5Pin = 26;
26
27 //ESP32 wird in pwm Zustand versetzt
28 // PWM (Pulsweitenmodulation) wird für die Ansteuerung der //
29 Servomotoren benötigt.
30 ESP32PWM pwm;
31
32 //ESP32 MAC-Adresse Handschuh: C4:4F:33:76:E6:8D
33 uint8_t broadcastAddress[] = {0xC4, 0x4F, 0x33, 0x76, 0xE6, 0x8D};
34
35 // 5-Dimensionales Array um einkommende Daten zu Speichern
36 int converted_flex[5];
37
38
39 //Struktur für einkommende Daten (Muss gleich mit Sender sein)
40 typedef struct struct_message {
41   int send_F1;
42   int send_F2;
43   int send_F3;
44   int send_F4;
45   int send_F5;
46 } struct_message;
47
48 // Erstellt ein struct_message Objekt um einkommende Daten zu
49 speichern
50 struct_message incomingReadings;
```

```
51
52  // Callback-Funktion wenn Daten empfangen werden
53  void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int
54  len) {
55    memcpy(&incomingReadings, incomingData, sizeof(incomingReadings));
56  //Eingehende Daten warden mit prozentual mit den Werten der // Servos
57  im geöffneten Zustand verrechnet und im
58  // converted_flex Array gespeichert
59    converted_flex[0] = incomingReadings.send_F1*0.6;
60    converted_flex[1] = incomingReadings.send_F2*0.9;
61    converted_flex[2] = incomingReadings.send_F3*0.9;
62    converted_flex[3] = incomingReadings.send_F4*1.0;
63    converted_flex[4] = incomingReadings.send_F5*0.8;
64
65  }
66
67  void setup() {
68    //Array auf Ausgangsposition 0 stellen
69    converted_flex[0] = 0;
70    converted_flex[1] = 0;
71    converted_flex[2] = 0;
72    converted_flex[3] = 0;
73    converted_flex[4] = 0;
74
75    //Alle Timer werden zugewiesen, um bestmögliche
76    // Synchronisation zu gewährleisten
77    ESP32PWM::allocateTimer(0);
78    ESP32PWM::allocateTimer(1);
79    ESP32PWM::allocateTimer(2);
80    ESP32PWM::allocateTimer(3);
81
82    //Frequenz des Servos wird definiert (50hz)
83    servo1.setPeriodHertz(50);
84    servo2.setPeriodHertz(50);
85    servo3.setPeriodHertz(50);
86    servo4.setPeriodHertz(50);
87    servo5.setPeriodHertz(50);
88
89    //Servos werden eingebunden
90    servo1.attach(servo1Pin, minUs, maxUs);
91    servo2.attach(servo2Pin, minUs, maxUs);
92    servo3.attach(servo3Pin, minUs, maxUs);
93    servo4.attach(servo4Pin, minUs, maxUs);
94    servo5.attach(servo5Pin, minUs, maxUs);
95
96    //Serial Monitor wird gestartet (Nur für Debugging)
97    Serial.begin(115200);
98
99    //WiFi wird im Modus WIFI_STA gestartet
100   WiFi.mode(WIFI_STA);
101
102   //ESP-NOW wird initialisiert
103   if (esp_now_init() != ESP_OK) {
104     Serial.println("Fehler beim initialisieren von ESP-NOW");
105     return;
```

```
106     }
107
108     //Peer wird registiert
109     esp_now_peer_info_t peerInfo;
110     memcpy(peerInfo.peer_addr, broadcastAddress, 6);
111     peerInfo.channel = 0;
112     peerInfo.encrypt = false;
113
114     //Peer wird hinzugefügt
115     if (esp_now_add_peer(&peerInfo) != ESP_OK){
116       Serial.println("Failed to add peer");
117       return;
118     }
119
120     //Registrierung einer Callback-Funktion, die aufgerufen
121     // wird wenn Daten empfangen werden
122     esp_now_register_recv_cb(OnDataRecv);
123   }
124
125   void loop() {
126
127   //Ausführen der Methode updatefingers()
128   updatefingers();
129     //Daten werden im Intervall von 50 ms empfangen und die Servos
130     // werden auf den neuen Wert aktualisiert.
131
132   delay(50);
133   }
134
135   void updatefingers(){
136     //Servos werden auf den übertragenen Wert gesetzt
137     servo1.write(converted_flex[0]);
138     servo2.write(converted_flex[1]);
139     servo3.write(converted_flex[2]);
140     servo4.write(converted_flex[3]);
141     servo5.write(converted_flex[4]);
142
143   //Servowerte warden in über die serielle Schnittstelle an
144   // den Computer übertragen (Für Debugging)
145     Serial.println("Daumen: " + String(converted_flex[0])+ "%\n" +
146   "Zeigefinger: " + String(converted_flex[1])+ "%\n" + "Mittelfinger: "
147   + String(converted_flex[2])+ "%\n" +"Ringfinger: " +
148   String(converted_flex[3])+ "%\n" +"kleiner Finger: " +
149   String(converted_flex[4])+ "%");
150     Serial.println();
151   }
```

4.4 Berechnung der Fingerposition

Im Folgenden werde ich die Verarbeitung der Sensorwerte bis hin zu der genauen Beugung der Prothesenfinger übersichtlicher anhand eines Beispielfingers erläutern, da dies in der Programmierung natürlicherweise unübersichtlicher ist, damit der Computer es am effizientesten versteht.

Die Werte des Beugungssensors werden durch (hier beispielhaft des Daumens) `int flex_read_1 = analogRead(F1_Pin);` eingelesen und durch den seriellen Monitor ausgegeben. Man versetzt den Finger in einen offenen und graden Zustand und der Wert wird notiert. Man erhält:

	Finger 1	Finger 2	Finger 3	Finger 4	Finger 5
Gestreckt	1600	1600	1650	1750	1700
Geschlossen	1050	1000	800	1100	800

Als nächstes wird folgende Rechnung durchgeführt:

```
converted_flex[0] = 100 - ((flex_read_1 - 1050) / 5.5);
```

Somit wird der Wert converted_flex[n] allgemein so berechnet:

$$converted\_flex[n] = 100 - \frac{gelesener\ Sensorwert_n - Finger\ geschlossen_n}{(Finger\ offen_n - Finger\ geschlossen_n) * 0{,}01}$$

Man erhält eine Zahl. 100 entspricht dabei einem voll geschlossen Finger und 0 einem voll geöffneten. Da es möglich ist, dass Werte über 100 oder unter 0 erreicht werden, sorgt eine for-Schleife in Zeile 127 bis133 des Handschuhcodes für eine Bereinigung. Egal wie stark der Finger geknickt wird, ein Wert von über 100 oder unter 0 kann nicht erreicht werden.

Wenn converted_flex[n] gesendet und empfangen wird, dann multipliziert der ESP32 der Prothese den Wert mit dem Winkel des Servos mal 0,01, wenn der Prothesenfinger geschlossen ist. Die Prothese berechnet also folgendes:

$$converted\_flex[n] = einkomender\ Wert * Servowinkel_{max.\ von\ n} * 0{,}01$$

Nun laufen beide, Sender und Empfänger synchron und die Fingerposition ist identisch.

Die Prothese ist nun voll funktionsfähig und kann genutzt werden.

5 Anwendungsmöglichkeiten

Die Prothese ist speziell entwickelt für Personen, denen der rechte Unterarm fehlt. Ein Ellbogengelenk muss allerdings vorhanden sein. Menschen mit solchen Behinderungen haben im Alltag viele Schwierigkeiten. Das Greifen ist nur auf eine Hand beschränkt. Es gibt zwar viele verschiedene Prothesen, allerdings sind diese mit enormen Kosten verbunden. Klar – Prothesen von großen Herstellern wie bebionic sind viel leichter, ausgereifter, stärker und effizienter als mein Modell, jedoch zeigt meine Arbeit, dass solche Handicaps bald der Vergangenheit angehören können. Meine Prothese kann jedermann von zuhause nachbauen und Modifikation vornehmen. Jedoch zeigt mein Projekt wahrscheinlich nur das Konzept hinter einer Prothese. Für eine alltägliche Benutzung bedarf es hier noch an starker Weiterentwicklung.

Auch ist der Handschuh als Konzept einer Robotersteuerung in der Industrie funktionsfähig. In der Industrie bedarf es oft direkter menschlicher Kontrolle über Roboter. Es kann auch vorkommen, dass menschliche Hände für eine Arbeit benötigt werden, aber die Arbeit für echte Hände viel zu gefährlich wäre, wie z.B. bei einer Bombenentschärfung. Hier kann der Entschärfer aus der Ferne über eine Kameraverbindung eine Bombe mit seinen Händen entschärfen, obwohl er nicht anwesend ist. Die Fingerwerte werden drahtlos zu einer Roboterhand übertragen, die sich an der Gefahrenstelle befindet.

6 Fazit

Die Konzeption und Herstellung der Handprothese und der Handschuhsteuerung war erfolgreich und die fertigen Produkte sind nun nutzbar. Jedoch ist das Produkt für eine serienreife Produktion noch viel zu unterentwickelt. Das Projekt zeigt aber, dass es möglich ist eine solch komplexe Prothese mit drahtloser Steuerung selbst herzustellen. Leider ist mit jeder Entwicklung verbunden, dass man nicht sofort den besten Weg zum Ziel findet. Viele verschiedene Lösungswege und Ansätze mussten verworfen werden, weshalb sich die gesamte Entwicklung auf circa 3 Jahre streckte. Die meiste Arbeit geschah jedoch im Jahr 2020 bis April 2021.

Das Projekt verbindet Mechanik, Elektronik, Design und Informatik und umfasst somit viele Themengebiete, die in der Zukunft noch große Rollen spielen werden.

Ich danke Herrn **Marc Lachmann** und Herrn **Christian Becker-Andermahr** für die jahrelange Unterstützung. Auch **Till Wegener** möchte ich meinen Dank aussprechen, der als Co-Leiter des schuleigenen FabLabs immer für Anregungen offen war.

Ich bedanke mich außerdem bei **Gael Langevin**, dem Erfinder des InMoov Projektes und dem Sponsor des Projekts **Reha Media - Elektronische Hilfsmittel für Behinderte Elektronik.**

7 Quellenverzeichnis

Das Quellenverzeichnis mit klickbaren Links kann bis mindestens dem 31.12.2021 unter www.philipplemke.com/bll gefunden werden.

[1] https://www.amazon.de/gp/bestsellers/digital-text/611274031/ref=zg_bs (Stand: 7.4.2021)
[2] https://www.youtube.com/watch?v=fY-TsxkxYwA (Stand: 7.4.2021)
[3] http://inmoov.fr/ (Stand 10.8.2020)
[4] https://de.wikipedia.org/wiki/Arduino_(Plattform) (Stand 7.4.2021)
[5] Von Clic17 - Eigenes Werk, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=34226017 (Stand 7.4.2021)
[6] https://de.wikipedia.org/wiki/InMoov (Stand 10.8.2020)
[7] Eigenes Werk nach CC BY-SA 4.0 von Philipp Lemke vom 15.10.2018
[8] http://inmoov.fr/project/ (Stand 7.4.2021)
[9] https://creativecommons.org/licenses/by-nc/3.0/deed.de (Stand 7.4.2021)
[10] https://www.amazon.de/3D-Druck-Handbuch-f%C3%BCr-junge-Einsteiger-ebook/dp/B08848681V (Stand 8.4.2021)
[11] „3D-Druck Handbuch für junge Einsteiger: 3D-Druck mit Ultimaker Cura für SchülerInnen und LehrerInnen“ von Philipp Lemke von 2020 [10]
[12] https://www.az-delivery.de/products/az-delivery-servo-mg995?variant=12236815138912 (Stand 10.4.2021)
[13] Battery Shield, Modell digital nachgebaut und gerendert. Software Fusion360
[14] https://www.amazon.de/gp/product/B0822Q4VS4 (Stand 10.4.2021)
[15] VTC6 Akkus, Modell digital nachgebaut und gerendert. Software Fusion360
[16] https://www.amazon.de/gp/product/B087R9XXM8 (Stand 10.4.2021)
[17] Datenblatt: https://cdn.shopify.com/s/files/1/1509/1638/files/ESP-32_DevKit_C_V4_Datenblatt_AZ-Delivery_Vertriebs_GmbH_24ec770f-c65e-4bd3-92c9-cd64b4d070b8.pdf?v=1615364587 (Stand 10.4.2021)
[18] https://www.az-delivery.de/products/esp-32-dev-kit-c-v4 (Stand 10.4.2021)
[19] https://www.amazon.de/gp/product/B085WJCRX8 (Stand 10.4.2021)
[20] https://www.amazon.de/gp/product/B0754MNQPN (Stand 10.4.2021)
[21] https://www.amazon.de/gp/product/B07HVND5DL (Stand 10.10.2020)
[22] https://de.wikipedia.org/wiki/Kohlenstofffaser (Stand 10.4.2021)
[23] https://www.amazon.de/gp/product/B086RVJ36S (Stand 10.4.2021)
[24] Eigenes Werk nach CC BY-SA 4.0 von Philipp Lemke vom 10.04.2021
[25] https://www.sparkfun.com/products/8606 (Stand 10.4.2021)
[26] https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/FLEXSENSORREVA1.pdf (Stand 10.4.2021)
[27] https://heltec.org/project/wifi-kit-32/ (Stand 11.4.2021)
[28] https://www.amazon.de/gp/product/B076P8GRWV (Stand 11.4.2021)
[29] Datenblatt http://resource.heltec.cn/download/WiFi_Kit_32/WIFI_Kit_32_pinoutDiagram_V2.pdf (Stand 11.4.2021)

[30] https://www.amazon.de/gp/product/B07YWLCTLK (Stand 14.4.2021)
[31] https://www.amazon.de/gp/product/B087LTZW61 (Stand 14.4.2021)
[32] https://www.amazon.de/gp/product/B07MC4G4S8 (Stand 14.4.2021)
[33] https://www.amazon.de/gp/product/B07NDBPYBR (Stand 14.4.2021)
[34] https://www.amazon.de/gp/product/B07BHRGJLJ (Stand 14.4.2021)
[35] https://www.amazon.de/gp/product/B083DC38TZ (Stand 14.4.2021)
[36] https://www.amazon.de/gp/product/B07TT69PPV (Stand 14.4.2021)
[37] https://de.wikipedia.org/wiki/CAD (Stand 14.4.2021)
[38] https://grabcad.com/library/lm2596-dc-to-dc-buck-converter-module-1 (Stand 14.4.2021)
[39] https://fritzing.org/ (Stand 14.4.2021)
[40] https://www.amazon.de/Artillery-Sidewinder-Neuestes-vormontiert-300x300x400mm/dp/B089NFVBGG (Stand 14.4.2021)
[41] https://www.amazon.de/dp/B07X37DT9M (Stand 15.4.2021)
[42] https://www.arduino.cc/en/software (Stand 15.4.2021)

Grafiken wurden mit den kostenlosen Programmen Fusion360, Fritzing und Cura Slicer erstellt.

8 Eigenständigkeitserklärung

Hiermit bestätige ich, Philipp Lemke, geboren am 16.11.2002, des Abiturjahrgangs 2021 am Gymnasium in den Filder Benden in Moers, dass ich die vorliegende Arbeit zum Thema „Konzeption und Herstellung einer elektronischen Hand- und Unterarmprothese, die durch 3D-Druckverfahren hergestellt und drahtlos mit einem externen Handschuh mit Fingerbeugungssensoren gesteuert wird auf Basis des ESP32 Mikrocontrollers“ selbständig verfasst und keine anderen als die angegebenen Hilfsmittel, bzw. Quellen benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, bzw. Zitate, wurden unter Angabe der Quelle kenntlich gemacht.

P. Lemke — Moers, 22.04.2021

Philipp Lemke — Datum, Ort