COSIMA´18

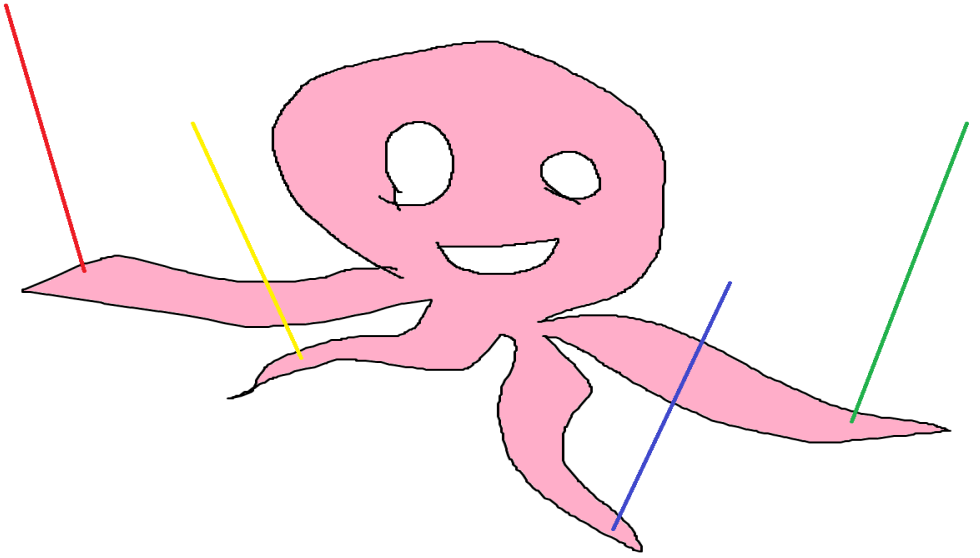
TOS Lehrstuhl für Technologie Optischer Systeme | RWTH AACHEN UNIVERSITY

Projektmappe für den COSIMA-Wettbewerb 2018

Der Gestikulaser

Die neue Technologie der Gestenerkennung

November 2018



Verfasst von

Christoph Behr
Cailing Fu
Nicole Grubert
Daniel Wolff

Inhaltsverzeichnis

1. Einleitung

Was wurde gemacht? Was war die Motivation? Inwiefern besitzt das Projekt eine Alltagsrelevanz?

Gestenerkennung ist immer wieder ein Thema, welches viel Aufmerksamkeit erregt. Und obwohl ein Mensch recht einfach verschiedene Gesten erkennen kann, ist es für den Computer eine große Herausforderung, zuverlässig die Gesten eines Menschen zu erkennen.
Mit dem Gestikulaser wollen wir ein neues System entwickeln, um Handgesten eines Menschen zu erkennen. Dabei soll dieses nicht mit einer Kamera arbeiten, wie die meisten heute verfügbaren Systeme, sondern die Hand des Nutzers soll mit Infrarot-LEDs beleuchtet und die Gesten sollen durch die erzeugten Reflektionsmuster erkannt werden. Eine auf diese Weise realisierte Gestenerkennung ist nicht nur Tageslicht unabhängig, sondern kann auch in vollkommener Dunkelheit betrieben werden. Mit Hilfe eines modularen Stecksystems aus mehreren Komponenten soll es möglich sein, ein individuelles Muster des Systems anzufertigen. Somit soll in Zukunft nicht nur die Handgesten-Steuerung möglich sein, sondern auch eine Ganzkörper-Gestensteuerung wie sie in einer Cave verwendet werden könnte, ermöglicht werden.

Im Alltag kann der Gestikulaser eingesetzt werden, um verschiedenste Dinge, wie ein ferngesteuertes Auto, eine Smart Home Einrichtung oder eine Drohne zu steuern.

Gestensteuerung

- zunehmend wichtigerer Bestandteil der Mensch-Computer-Interaktion - Idee: Entwurf einer neuartigen Gestensteuerung, die portabel ist und im Alltag bzw. in eingebetteten Systemen verwendet werden kann -

2. Stand der Technik

Mit dem zunehmenden Einfluss von Computern auf die Gesellschaft wurde auch die Interaktion von Mensch und Computer ein immer wichtigerer Bestandteil unseres Alltags. Innerhalb der letzten Jahrzehnte wurde auf diesem Gebiet stetig geforscht und die verfügbaren Technologien weiterentwickelt, mit dem Ziel, die Interaktion von Mensch und Computer so natürlich wie möglich zu gestalten. Diese Entwicklung erstreckte sich von der ursprünglichen Kommunikation mittels Maus und Tastatur, über Touchscreens und Virtual Reality Systeme, bis hin zu Sprachsteuerungen von elektronischen Endgeräten. Vor diesem Hintergrund hat auch das Interesse an Gestenerkennung als zusätzlichem natürlichen Kommunikationsmittel zugenommen.

Bei Gestenerkennungssystem kann man grundsätzliche zwischen zwei verschiedenen Arten von Gestenerkennung unterscheiden: Bei der *gerätebasierten Gestenerkennung* werden die Bewegungen eines Nutzers durch Beschleunigungs- oder Positionssensoren wahrgenommen und einer Geste zugeordnet. Bei dieser Art der Gestenerkennung muss der Nutzer die Sensorik in irgendeiner Form am Körper tragen, damit die Bewegungsmuster richtig aufgezeichnet werden können. Dabei kann z.B. ein Handschuh wie z.B. der CyberGlove II des Unternehmens CyberGlove Systems (siehe auch Abbildung 2.1) oder ein einfacher Controller wie bei der Spielekonsole Wii von Nintendo (siehe Abbildung 2.2) zum Einsatz kommen.

Kamerabasierte Gestenerkennungsverfahren nutzen externe Systeme, die die Bewegungen des Nutzers beobachten und Aufnahmen von dessen Bewegungsabläufen erstellen. Dabei werden meistens Kamerasysteme eingesetzt. Auf die aufgenommenen Bilddaten werden dann verschiedene Bildrekonstruktionsverfahren sowie Bildanalyseverfahren angewendet, um die Gestendaten zu extrahieren. Nachdem die Gestendaten extrahiert wurden, erfolgt ein Abgleich mit einer Datenbank, in der verschiedene Aufnahmen von verschiedenen Gesten gespeichert sind. Dieser Datenbankabgleich erlaubt dann letztendlich die Zuordnung der Geste. Ein Beispiel für

Abbildung 2.1.: Der CyberGLove II von CyberGlove Systems. Er enthält verschiedene Sensoren, unter Anderem Biegesensoren für die Finger, sowie Sensoren zur Detektion der Drehung des Handgelenks. Quelle: http://www.cyberglovesystems.com/cyberglove-ii/

ein System, das auf diese Weise arbeitet, ist die Kinect Erweiterung für die XBox-Spielekonsole von Microsoft (siehe auch Abbildung 2.3).

Beide Arten von Gestenerkennungssystemen werden aktuell eingesetzt. Allerdings haben auch beide Verfahren gewisse Nachteile: So erfordern gerätebasierte Gestenerkennungssysteme immer Sensorik am Körper des Nutzers, was sich in alltäglichen Situationen häufig als unkomfortabel erweist. Demgegenüber können kamerabasierte Gestenerkennungssysteme beispielsweise nur solche Gesten erkennen, die in der Datenbank vorhanden sind. Zudem sind Datenbankabfragen und Vergleiche mit den darin gespeicherten Daten häufig sehr rechenaufwändig, was sich als problematisch erweisen kann, falls eine Echtzeit Erkennung der Gesten gefordert ist. Eine Anbindung an die Datenbank erfordert zudem einen ständigen Internet Zugang oder alternativ große Mengen an Speicherplatz, um die für den Abgleich benötigten Gestendaten lokal zu speichern. Insbesondere für den Einsatz in eingebetteten Systemen, wie z.B. in der Automobil-Industrie sind diese beiden Gestenerkennungsverfahren nur bedingt geeignet: Gerätebasierte Gestenerkennungssysteme benötigen die externe Sensorik am Körper des Nutzers, die mit dem eingebetteten System kommunizieren und Daten austauschen muss. Dabei ist in eingebetteten System häufig die Kommunikation über externe Schnittstellen sehr zeitaufwändig. Eingebettete Systeme dürfen zudem oft nur einen geringen Energieverbrauch aufweisen. Das erschwert den Einsatz von den Kamerasystemen, die von kamerabasierten Erkennungsverfahren benötigt werden. Zudem muss die für die Bilderkennung bzw. -analyse eingesetzte

Abbildung 2.2.: Ein Controller der Nintendo Spielekonsole Wii. Die Erkennung der Bewegungen des Spielers erfolgt mit Hilfe von Positions- und Beschleunigungssensoren. Quelle: https://www.chip.de/artikel/Nintendo-Wii-Test-2_140216386.html

Software plattformunabhängig sein, damit sie auch im Rahmen des eingebetteten Systems eingesetzt werden kann.

Genau an dieser Schnittstelle zu eingebetteten Systemen soll der Gestikulaser eingesetzt werden können. Durch die Verwendung von einfachen elektronischen Bauteilen in Kombination mit Mikrocontrollern benötigt der Gestikulaser nur wenig Energie. Die kompakte und modulare Bauweise, auf die in Referenz zu Nicoles Abschnitt über die Konstruktion einfügen genauer eingegangen wird, erlaubt den Einsatz in unterschiedlichen Größenmaßstäben, da der Gestikulaser bei Bedarf einfach erweitert werden kann. Da die für die Gestenerkennung benötigten mathematischen Modelle (siehe dazu auch Abschnitt 3.4) für jede Anwendung individuell offline antrainiert werden, benötigt das System im Live-Betrieb vergleichsweise wenig Speicherplatz, was es ebenfalls für den Einsatz in eingebetteten Systemen qualifiziert.

Abbildung 2.3.: Das kamerabasierte Gestenerkennungssystem Kinect von Microsoft. Es werden Aufnahmen des Nutzers gemacht, die nach einer Analyse mit einer Datenbank abgeglichen werden, um die Geste zu zuordnen. Quelle: https://blogs.microsoft.com/ai/kinect-for-windows-game-on-for-commercial-use/

3. Funktionsweise und Aufbau

Die Funktionsweise des Gestikulasers besteht aus der Detektion und Weiterverarbeitung von Lichtsignalen, welche mit einer Geste erzeugt werden.

Dabei wird infrarotes Licht von einer LED Quelle durch die Hand reflektiert und mit Hilfe von mehreren Photodioden detektiert. Die durch die Photodioden erhaltenen Daten werden dann in einer Software weiterverarbeitet, welche mit Hilfe von Machine Learning die tatsächliche Handgeste erkennt. Von dort aus kann dann jedes beliebige Endgerät angesteuert werden.

Der Gestikulaser selbst besteht aus einer Platte, der Photoplatte, welche aus verschiedenen Steckmodulen zusammen gesteckt werden kann. Die Steckmodule bestehen aus einzelnen kleinen Boxen, in welche die Elektronik integriert ist. In der Mitte befindet sich der Oktokommander, welcher durch weitere Detektormodule erweitert werden kann. Auf jedes der Module befinden sich vier Photodioden um das reflektierte Licht zu messen.

3.1. Oktokommander

Ein Arduino Micro steuert den Oktokommander mittels eines I2C Bus an. Der I2C Bus benötgt lediglich 2 Pins auf dem Arduino, SDA für die eigentliche Datenverbindung und SCL für den vom Arduino vorgegebenen Takt. Jeder Sensor wird über eine, im System, einzigartige Hardware Adresse (I2C-Adressen) angesteuert. Der Oktokommander erzeugt mittels eines I2C Expanders je 4 I2C Adressen die für die Zuordnung und Ansteuerung der einzelenen Verstärker benötigt werden. Der I2C multiplexer erhöht die Anzahl der verwendbaren I2C Expandermodulen, da diese intern max. 4 unterschiedliche I2C Adressen erziehen können. Durch das multiplexen der vom Oktokommander erzeugten Adressen können so bis zu 1024 Sensoren angesteuert werden ohne das verschiedene Expandermodule verwendet werden müssen. Um den Oktokommander mit den Detektormodulen erweitern zu können wur-

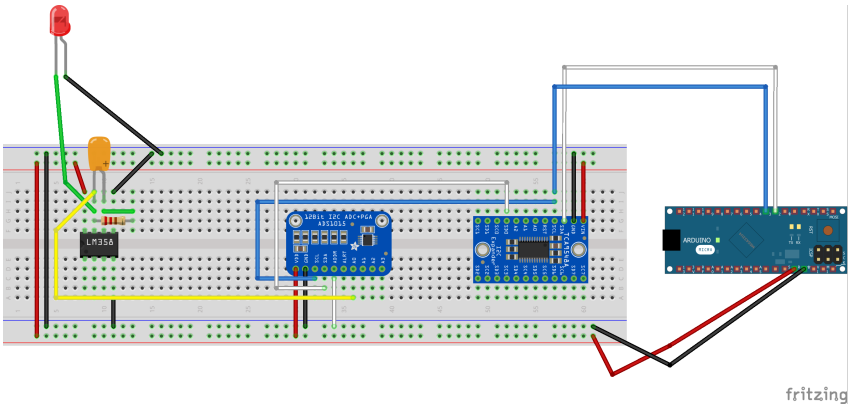


Abbildung 3.1.: Schematischer Aufbau des Oktokommanders bestehend aus einem Arduino einem I2C Multiplexer, einem I2C Expander und einem Detektormodul

den USB-2A Schnittstellen verwendet. Durch diese Schnittstellen können bis zu 7 Detektormodule, jeweils eines pro Seite angesteuert werden werden. Um eine größere Platte zu konstruieren können noch weitere Detektormodule an den bereits vorhandenen Detektormodule angeschlossen werden.

3.2. Transimpedanzverstärker

Die Verstärkerschaltung besteht aus zwei sogenannten Transimpedanz Verstärkern. Der Aufbau besteht aus einem Operationsverstärker vom Typ LM358 und je zwei Widerständen und Kondensatoren sowie zwei Infrarot Photodioden (siehe Abbildung 3.2). Die Photodiode fungiert in ihm als Konstantstromquelle, die einen kleinen

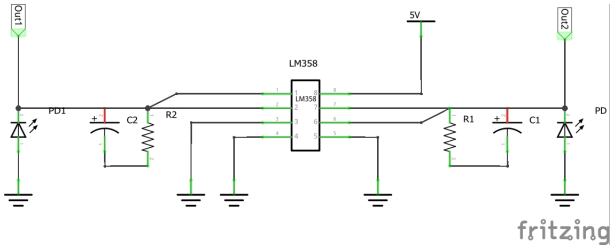


Abbildung 3.2.: Schematische Darstellung des Transimpedanzverstärkersystems im Detektormodul

Strom an das System abgibt. Mit Hilfe des Widerstandes R_1 und des Kondensators C_1 kann der Operationsverstärker (LM358) eine vom Mikrocontroller verarbeitbare hohe Spannung ausgeben. Wichtig ist hier die möglichst genaue Wahl der Kapazität C_1 ohne die das System anfangen würde zu Schwingen. Es ergibt sich:

$$C_1 = \sqrt{\frac{C_I}{R_1 \cdot GBP}}$$

Hierbei steht GBP für *Gain Bandwidth Product*. Sie ist die Frequenz, bei der keine Verstärkung mehr stattfindet. Sie ist somit die Obergrenze für einen praktikablen Einsatz des Systems Für den Operationsverstärker LM358 ist das GBP mit 1 MHz im Datenblatt angegeben. C_I ist die Summe der beiden Sperrschichtkapazitäten der Photodiode und des Operationsverstärkers. Typische Anwendungsgrößen für R_1 liegen bei etwa 1 MΩ. Der Arbeitsbereich des Verstärkersystems beschränkt sich durch aktive Bauelemente auf einen bestimmten Frequenzbereich. Dieser lässt sich wie folgt berechnen:

$$f_{3\,\mathrm{dB}} = \sqrt{\frac{GBP}{2\pi \cdot R_1 \cdot C_I}}$$

Für unser System liegt die Bandbreite bei etwa 73 kHz. Dies ist auch die Grenzfrequenz des Verstärkersystems. Die Grenzfrequenz ist die Frequenz bei der die Ver-

stärkung auf den $\frac{1}{\sqrt{2}}$ fachen Wert der maximalen Verstärkung gesunken ist und bei der die Hälfte der maximalen Leistung an einen, rein Ohm'schen, Verbraucher abgegeben wird.

3.3. Detektormodul

Das Detektormodul besteht lediglich aus vier Photodioden und einem USB-2A Eingang. Dadurch kann das Detektormodul an den Oktokommander angeschlossen werden. Zur Ansteuerung der Photodioden wurde auch hier ein i2c Expander verwendet.

3.4. Software

Die Software-Seite des Gestikulasers ist für die Zuordnung der Gesten zuständig. Die von den Detektormodulen gemessenen Reflektionsmuster werden an den Mikrocontroller im Oktokommander und von da aus an einen Computer weitergeleitet, wo sie verarbeitet werden. Bei der dazu eingesetzten Software wurde komplett auf open-source verfügbare Programme und Bibliotheken gesetzt. Der Code auf den Microcontrollern wurde mit Hilfe der `Arduino IDE` entwickelt. Die Verarbeitung am Computer erfolgt mit Hilfe von `Python` Skripten. Für den Machine Learning Teil wurde die `TensorFlow`™ Bibliothek verwendet. Die Software kann in zwei Teile unterteilt werden, die im Folgenden genauer erläutert werden.

Trainingsphase

Der erste Teil der Software kommt während der Trainingsphase zum Einsatz. Hier werden die von den Photodioden gemessenen Daten aus dem Microcontroller im Oktokommander zunächst ausgelesen und gemeinsam mit einem Label, das der gerade aufgenommenen Geste entspricht, in eine Datei geschrieben. Die auf diese Weise gesammelten Daten werden dann im nächsten Schritt verwendet, um ein mathematisches Modell zu erstellen, das im Live-Betrieb angewendet wird, um die vom Nutzer gemachte Geste, einer der bekannten zuordnen zu können. Angenommen es stehen Daten von N Photodioden zur Verfügung, dann können diese Daten in dem Vektor $x \in \mathbb{R}^N$ zusammengefasst werden. Für den Anfang soll ein Modell entwickelt werden, welches einem beliebigen gemessenen Datensatz x eine der m vordefinier-

ten Gesten $G_1, ..., G_m$ zuordnet. Dazu stellt der Nutzer vor Aufzeichnung der Daten in dem Python Skript ein, was für eine Geste G_j er als nächstes aufnehmen will. Nachdem für jede Geste ausreichend Daten aufgenommen wurden, kann als nächstes das mathematische Modell erstellt werden, um die Sensordaten zu verarbeiten. Ziel ist es, ein gegebenes Datum x_i, das im Live-Betrieb gemessen wird, eindeutig einer Geste G_j zuzuordnen. Damit Datum x_i der Geste G_j zugeordnet wird, muss die Bedingung

$$p(G_j|x_i) \geq p(G_k|x_i) \quad \forall k = 1, ..., m \quad k \neq j$$

erfüllt sein. Dabei ist $p(G_j|x_i)$ die bedingte Wahrscheinlichkeit für Geste G_j gegeben die Photomessdaten x_i. Diese Bedingung lässt sich leicht überprüfen, allerdings ist die bedingte Wahrscheinlichkeit $p(G|x)$ unbekannt. Aus diesem Grund wird ein neuronales Netz mit Hilfe der zuvor aufgezeichneten Daten $(x, G) \in \mathbb{R}^N \times \mathbb{R}$ trainiert, das die unbekannte bedingte Wahrscheinlichkeit annähert. Der Aufbau und das Training des neuronalen Netzes erfolgen mit Hilfe vordefinierter Methoden von `TensorFlow`$^{\text{TM}}$. Nach erfolgreich abgeschlossenem Training wird das neuronale Netz in eine Datei exportiert und kann nun im Live-Betrieb verwendet werden.

Live-Betrieb

Im Live-Betrieb wird das zuvor trainierte Modell nun verwendet, um die nun unbekannten Reflektionsmuster, die von den Photodioden gemessen und über den Mikrocontroller an den PC weitergeleitet wurden, einer bekannten Geste zuzuordnen. Nachdem die gemessenen Daten einer Geste zugeordnet wurden, wird der mit dieser Geste verknüpfte Steuerungsbefehl bestimmt und an das angeschlossene Endgerät weitergeleitet. Dieser Ablauf ist in Abbildung 3.3 dargestellt.

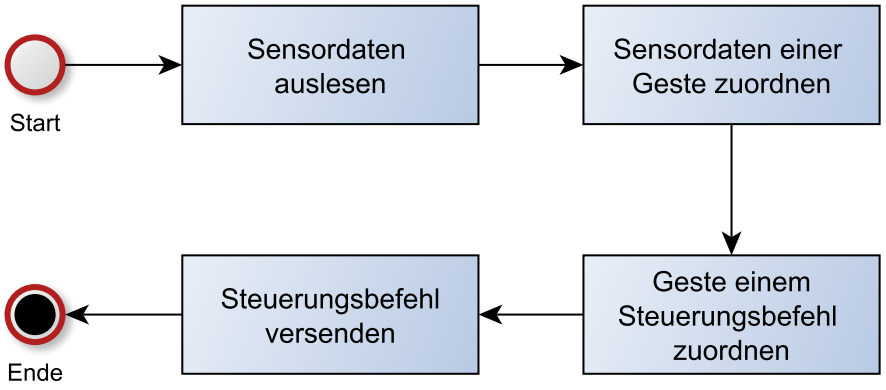


Abbildung 3.3.: Ablaufdiagramm der Software im Live-Betrieb: Die aus dem Microcontroller ausgelesenen Sensordaten werden zur Verarbeitung an das Python Skript übergeben, wo sie einer Geste und diese einem Steuerbefehl zugeordnet wird. Dieser Steuerungsbefehl wird schließlich an das anzusteuernde Endgerät weitergegeben.

4. Marketing und Kosten

Neben der Entwicklung unseres Produktes wurde im Laufe der Entwicklungszeit zusätzliche Öffentlichkeitsarbeiten geleistet. So wurde die Website `https://www.gestikulaser.de/` ins Leben gerufen sowie der Instagram Account `laserharptos` mit über 100 Abonnenten betreut. Im Zuge dessen wurde auch Sponsoren angeworben, welche Interesse an der Unterstützung unseres Projektes hatten.

Folgende Sponsoren haben uns bei diesem Projekt unterstützt:

Aconity3D GmbH

Fraunhofer ILT

TOS RWTH Aachen University

Würth Electronik GmbH & Co. KG

Die Produktionskosten unseres Gestikulasers können wie folgt aufgegliedert werden:

Oktokommander

Produktname	Kosten /Stk	Anzahl	Gesamtkosten	Kostenträger
Arduino Micro	19.99 €	1	19.99 €	TOS
I2C-Multiplexer TCA9548A	7.80 €	1	7.80 €	ILT
I2C ADS1015	11.20 €	1	11.20 €	ILT
Infrarot LED		4		Würth
LED Treiber		2		Würth
Infrarot Photodiode		4		ILT
OP-Verstärker	1.85 €	4	7.40 €	TOS
UBS 2 TypA -Mount		7		Würth
Passive Bauteile	———	—	1 €	TOS
			60 €	

Detektormodul

Produktname	Kosten /Stk	Anzahl	Gesamtkosten	Kostenträger
I2C ADS1015	11.20 €	1	11.20 €	ILT
Infrarot Photodiode		4		ILT
OP-Verstärker	1.85 €	4	7.40 €	TOS
UBS 2 TypA -Plug		1		Würth
UBS 2 TypA -Mount		1		Würth
Passive Bauteile	——	-	1 €	TOS
			20 €	

Dabei wurde ein Oktokommander und sieben Detektormodule aufgebaut. Die Gesamtkosten für die Produktion belaufen sich also auf 200€.

Darüber hinaus sind Kosten von BLA € für Unterkunft und Anreise entstanden sowie T-Shirts in Wert von BLA €. Die T-Shirts wurden von Aconity3D gespondert.

A. Ausblick

Erweiterung um Sensorhandschuh zur verbesserten Erkennung der Daten. Überarbeitung des Designs um die Module besser bzw. anders zusammen zu stecken.

An dieser Stelle möchten wir einen Ausblick geben, wie der von uns entwickelte Gestikulaser weiterentwickelt werden könnte.

Hier wäre zunächst einmal eine Verfeinerung der aktuell durchführbaren Gestenerkennung denkbar, die nicht nur die Stellung der Hand, sondern auch die Krümmung der einzelnen Finger berücksichtigt. Zu diesem Zweck wurde bereits ein Prototyp für einen Sensorhandschuh entwickelt.

Dieser könnte während der Trainingsphase vom Nutzer dazu verwendet werden, feinere Gesten aufzunehmen, wobei neben den von den Detektormodulen detektierten Photoströmen zusätzlich die Daten der auf dem Sensorhandschuh befindlichen Module gespeichert werden. Aktuell kommen hier Biegesensoren für jeden Finger, sowie ein Gyroskop und ein Beschleunigungssensor zum Einsatz. Diese verfeinerten Gestendaten stellen nun natürlich eine neue Herausforderung an das Modell, das zur Auswertung der Photoströme im Live-Betrieb eingesetzt wird: Statt wie vorher den eingehenden Photoströmen nur das Klassenlabel einer Geste zuzuordnen, muss es nun in der Lage sein, auf Basis der gemessenen Photoströme die wahrscheinliche Lage der Hand, sowie die Krümmung der Finger vorauszusagen. Dieses Problem ist mathematisch deutlich schwieriger zu lösen, was nicht zuletzt an der deutlich höheren Anzahl an Parametern liegt, die bestimmt werden müssen, um eine ausreichende Aussagekraft zu gewährleisten. Aufgrund der höheren Anzahl an Parametern werden zudem auch mehr Daten benötigt, um das Modell zu trainieren.

Ebenfalls denkbar wäre eine Erweiterung auf dynamische Gesten, bei denen der Nutzer die Hand bewegt. In diesem Fall müssen die Daten der Photodioden im Live-Betrieb in Form Zeitreihe aufgenommen werden und das neuronale Netzmodell muss diese Zeitreihe mit einer Geste verknüpfen. Auch hier ergibt sich ein mathematisch deutlich komplexeres Problem, das ein aufwändigeres Training und viele Trainings-

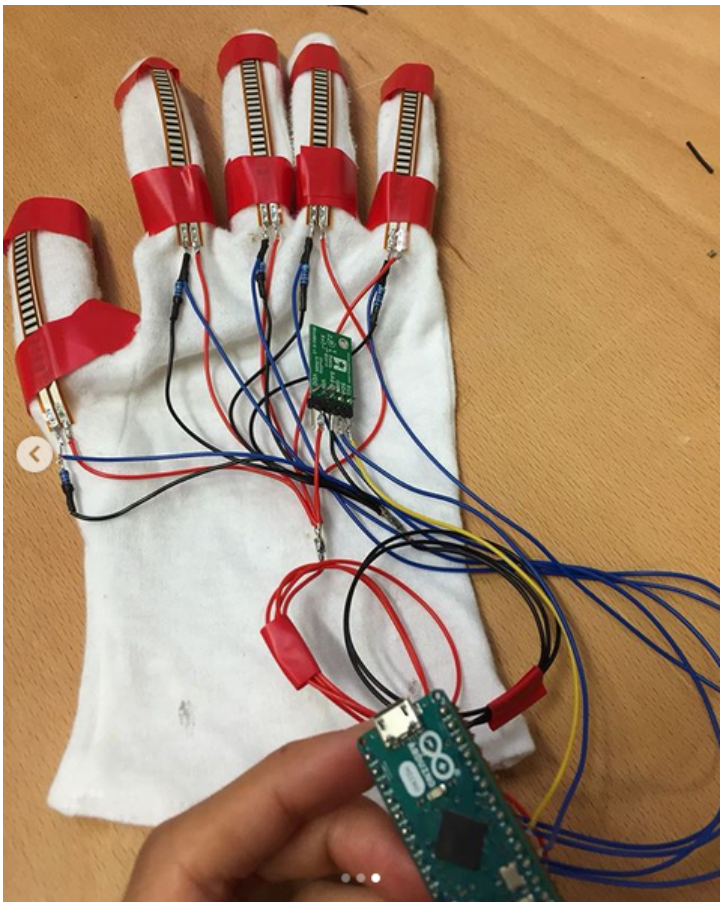


Abbildung A.1.: Aktueller Prototyp des Sensorhandschuhs. Es sind Biegesensoren für alle Finger, ein Gyroskop und ein Beschleunigungssensor im Einsatz. Bei der Verarbeitung der Sensordaten kommt ein Arduino Micro zum Einsatz.

daten erfordert. Die Erweiterung um dynamische Gesten könnte ebenfalls mit der Verwendung des Sensorhandschuhs kombiniert werden.