



UNIVERSITY OF ST.GALLEN

School of Management, Economics,
Law, Social Sciences, International Affairs and Computer Science

Hands-on HCI
Click-y-Point
The Magic Finger

Submitted by:
Sebastian Oes 22-618-441
Leon Muscat 22-619-621
Max Beringer 22-613-152

Date of Submission:
18.11.2024

Inhaltsverzeichnis

1 Einleitung	3
2 Motivation	4
2.1 Warum ist das Projekt nützlich?	4
2.2 Welches Problem löst es?	4
2.3 Herangehensweise	5
3 Umsetzung	6
3.1 Projektentwicklung	6
3.2 Hardware und Software	8
3.3 Software Features	10
4 Reflektion des Projektes	11
4.1 Herausforderungen	11
4.2 Erfolge	11
4.3 Erkenntnisse	12
4.4 Mögliche Erweiterungen	12
5 Fazit	13
A Anhang	14
A.1 Github Repository	14
A.2 Produkt Video	14

Abbildungsverzeichnis

3.1	Gyroskop und Akzelerometer Daten für die Erkennung von Hangelenkbewegungen	7
3.2	Prototyp 1	9
3.3	Prototyp 1 Schaltplan	9
3.4	Prototyp 2	9
3.5	Prototyp 2 Schaltplan	9
3.6	Standard Ansicht	10
3.7	Menu	10
3.8	Mausempfindlichkeit	10
5.1	Click y Point - Logo	13

1 Einleitung

Jeder Student kennt dieses Problem: Man steht vor dem Publikum und erzählt voller Elan von seinem Projekt. Dabei bewegt man sich im Raum, aber dann möchte man etwas auf den Slides verdeutlichen, oder auch nur zur nächsten Slide wechseln. Jetzt muss man in einer unangenehmen Pause zurück zu seinem Laptop gehen. Deshalb entschieden wir, uns diesem Projekt anzunehmen. Die Idee war es, ein am Handgelenk getragenes Gerät zu entwerfen, das es dem Nutzer ermöglicht, seinen Computer durch Handbewegungen zu steuern und bei Präsentationen frei und interaktiv zu sprechen, ohne an ein herkömmliches Eingabegerät gebunden zu sein.

2 Motivation

2.1 Warum ist das Projekt nützlich?

Traditionelle Präsentationstechniken binden die Person oft an einen Computer oder ein Steuergerät. Unser Prototyp zielt darauf ab, diese Barriere aufzuheben, sodass der Sprecher sich frei bewegen kann, indem er den Mauszeiger durch Handbewegungen steuert und damit Präsentationen interaktiv und dynamisch gestaltet kann.

Dies ermöglicht ein besseres Präsentation Erlebnis und lässt die Zuschauer, ohne Unterbrechungen, Störungsfrei ins Thema eintauchen.

2.2 Welches Problem löst es?

Click y Point bietet eine hands-free Lösung zur Steuerung des Mauszeigers und zum Durchführen von Klicks, was in Präsentationssituationen oder anderen Anwendungen die Handhabung erleichtert und die Flexibilität erhöht.

Dabei lassen sich auch weitere Hotkeys auf Click y Point definieren und erlauben es dem Nutzer, die Hardware an seine individuellen Bedürfnisse anzupassen.

2.3 Herangehensweise

Schnell war klar, dass unser Vorhaben in drei Hauptprobleme aufgeteilt werden kann. Wir begannen also mit einem Brainstorming, und versuchten Lösungen für diese Probleme zu finden. Wir hatten drei Hauptprobleme die es zu Lösen galt:

1. Bewegungsverfolgung für die Hand
2. Zuverlässiges Klicken der Maus
3. Gesten Erkennung

Wir diskutierten welche Sensoren wir nutzen könnten und ob diese auch effizient wären und unsere Vision Unterstützen. So hatten wir beispielsweise die Idee, unsere Bewegungen mithilfe von einer Kamera und einem AI-Model zu trainieren und zu erkennen. Aber diese Idee stellte sich schnell als unrealistisch heraus, da man Bewegungen dann immer in Richtung der Kamera machen müsste, diese Kamera irgendwo dominant zu sehen wäre und somit unser Plan von einer eleganten, unauffälligen Lösung nicht umsetzbar wäre.

Nach einigem Diskutieren entschieden wir uns dann, verschiedene Sensoren auszuprobieren, um ihre ausgegebenen Daten zu bewerten und dann diese zu verwenden oder nach alternativen zu suchen.

3 Umsetzung

3.1 Projektentwicklung

Wie bereits erwähnt hatten wir drei Hauptprobleme, die es zu Lösen galt. Für das Erkennen der Bewegungen und die zuverlässige Verfolgung, war relativ schnell klar, dass wir ein Gyroskop, für die räumliche Lage und ein Akzelerometer für die Beschleunigung benötigen. Wir bauten also einen ersten kleinen Prototyp, um herauszufinden, was wir aus den Daten herauslesen können und wo gewisse Limitationen sind. Ebenfalls mussten wir überprüfen, wie genau die Daten sind und ob es überhaupt möglich ist, mit diesen Daten das Muster bestimmter Bewegungen ablesen zu können.

Der erste Prototyp basierte noch auf einem Sensor des Arduino Plug and Make Kits. Neben Hardwareelimitionen dieses Sensors Modul, stiessen wir auch schnell auf Probleme mit der Arduino Software Umgebung, so liess uns zum Beispiel die Mouse library von Arduino, nur eine sehr kleine maximale Geschwindigkeit des Mauszeigers definieren. Dies führte dazu, dass man die Maus nur sehr langsam über den Bildschirm bewegen konnte und die Steuerung mit der Hand sich somit unnatürlich anfühlte. Ebenfalls konnte man keine absoluten Mausdaten definieren, was dazu führte, dass wir keine Möglichkeit hatten den Mauszeiger zu zentrieren. Ebenfalls war schnell klar, dass wir mit Arduino alleine keine Möglichkeit hatten die Sensordaten kabellos an unsere Computer zu übertragen, da der Bluetooth Low Energy Chip des Arduinos kein Serial Support hat. Das bedeutet, ein Client-Programm auf dem zu kontrollierenden Gerät ist keine Option.

Wir begannen also die Suche nach einem besseren Sensor. Bei unserer Recherche stiessen wir auf den M5Stick 2 Plus, einem leistungsfähigen Board mit integriertem ESP32-Chip, Gyroskop, Akzelerometer, Akku, Display und mehreren Knöpfen.

Wir begannen also unseren Prototypen um dieses Board zu entwerfen.

Nach einem Testen und nach dem wir die Sensorwerte in einem Diagramm plotteten, stellten wir fest, dass wir Bewegungen viel besser am Gyroskop ablesen konnten, wie am Akzelerometer.

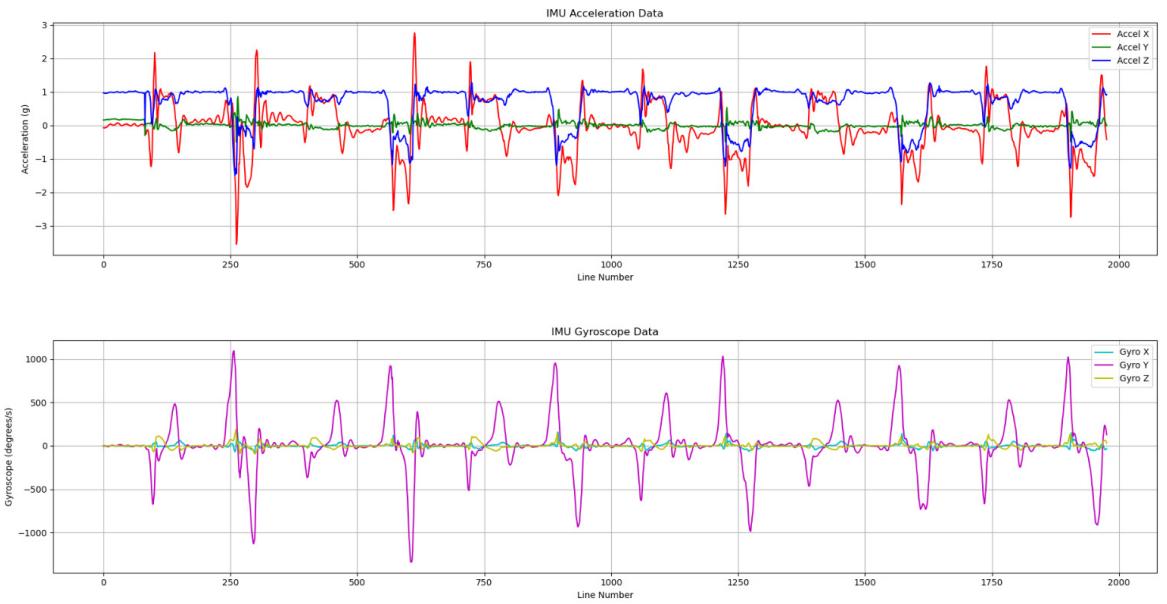


Abbildung 3.1: Gyroskop und Akzelerometer Daten für die Erkennung von Hangelenkbewegungen

Für die Umsetzung der Mausklicks hatten wir verschiedene Ideen. Zuerst wollten wir durch eine Pinch-Bewegung den Klick verursachen (ähnlich wie Apple Vision Pro nur mit Akzelerometerdaten), jedoch waren unsere Sensoren zu ungenau, um zuverlässig diese kleine Bewegung im Handgelenk zu identifizieren.

Dann kamen wir auf die Idee, zwei Kontaktpunkte an den Fingern zu haben, die durch Berührung wie ein Knopf funktionieren und so ein Signal verursachen. Dazu präparierten wir zuerst einen Handschuh mit Kontakten, um diese Idee auszuprobieren. Später stellten wir zwei kleine Ringe aus einem leitenden Material her, die wir dann mit zwei Kabel an unserem Gerät befestigten.

3.2 Hardware und Software

Sensoren und Steuerung: Der Gyroskopsensor steuert die Mausbewegungen; für die Kalibrierung der Nullposition wurde ein Knopf hinzugefügt. Später wurde die Kalibrierung auf einen eingebauten Knopf des M5Stick 2 Plus verlegt. Der Rotary Encoder zur Navigieren des Menüs und Sensitivitätsanpassung ermöglicht es, die Geschwindigkeit des Mauszeigers auf die individuellen Handbewegungen abzustimmen, sowie auch eine intuitive Navigation des eingebauten Menus.

Mausklicks: Der erste Prototyp nutzte einen Handschuh mit Drähten an Daumen und Zeigefinger, um durch Berührung Klicks zu simulieren. Später ersetzten wir diesen durch zwei Ringe, die mit einem Kabel verbunden sind und direkt an den Fingern getragen werden können.

Software: Um den M5Stick 2 Plus zu programmieren, muss man mit C++ arbeiten. Dies war eine weitere Herausforderung für uns, da keiner von uns viel Erfahrung mit C++ hatte. Wir haben verschiedenen Libraries benutzt, aber auch eigene Ideen umgesetzt, um zu unserem Ziel zu gelangen. Großes Problem war, dass die allermeisten Libraries schlecht dokumentiert oder nicht mehr funktionstüchtig sind. Dies war besonders mit Bluetooth HID (Human Interface Devices) Libraries der Fall. Daher haben wir uns entschieden eine bestehende so zu erweitern, dass sie unseren Anforderungen entspricht. Die Library die wir erweitert haben basiert auf dem Apache NimBLE Stack, welcher zwar noch neuer ist, wie etablierte Lösungen, aber dafür deutlich effizienter (sowohl für Speicher als auch Geschwindigkeit). Die Library hat allerdings nur Maus und Tastatur Support. Für die Kalibrierung müssen wir aber auch die Maus zentrieren. Daher haben wir die Bluetooth-Reports erweitert, sodass der ESP32 auch als Drawing Tablet auftritt. Kompiliert haben wir den Code dann wieder direkt in der Arduino, um ihn danach auch direkt auf die Hardware zu flashen.

Montage und Befestigung: Das gesamte Gerät wird mit einer 3D-gedruckten Halterung auf einem Uhrenband befestigt, das am Handgelenk getragen werden kann. Die ursprüngliche Befestigung mit einem Gummiband wurde durch eine Halterung ersetzt, an die ein Standard-Uhrenband angeschlossen werden kann.

Erster Prototyp:

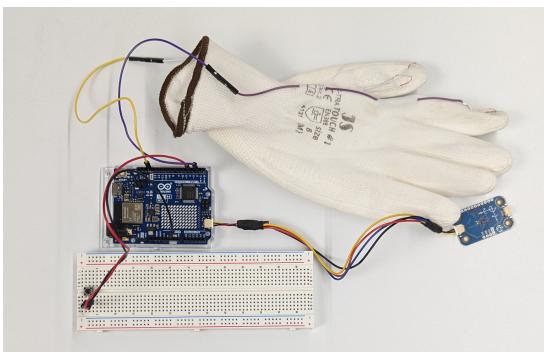


Abbildung 3.2: Prototyp 1

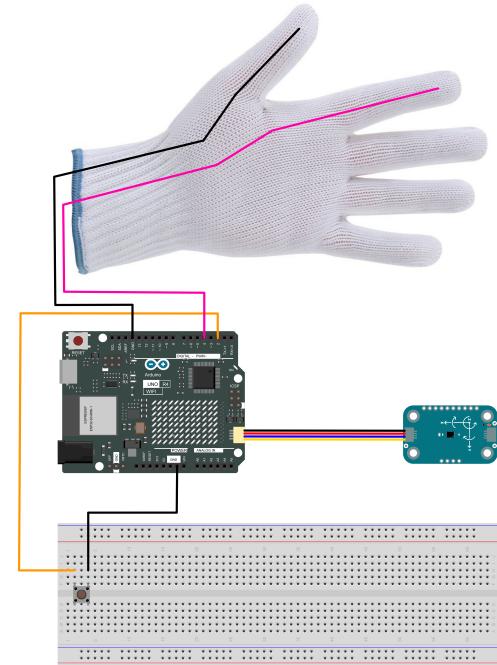


Abbildung 3.3: Prototyp 1 Schaltplan

Zweiter Prototyp:

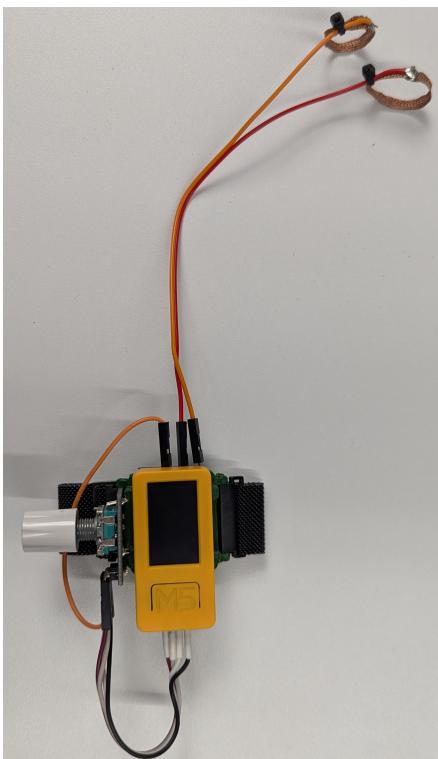


Abbildung 3.4: Prototyp 2

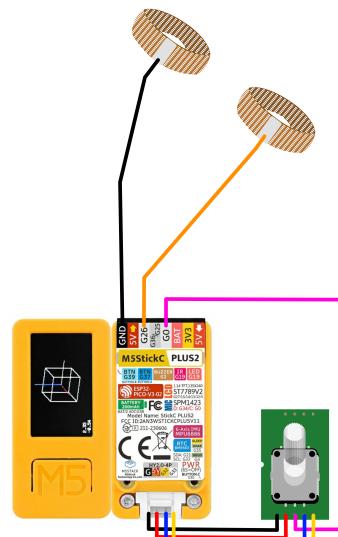


Abbildung 3.5: Prototyp 2 Schaltplan

3.3 Software Features

Neben unseren Hardware-Features haben wir auch einige Software-Features eingebaut. Um den ESP32 zu bedienen haben wir das verbauten Display des M5 Stick 2 Plus zusammen mit dem Rotory Encoder verwendet.

Unser Ziel war es hierbei ein intuitives UI zu erstellen, das dem User alle wichtigen Information anzeigt und das Präsentationserlebnis positiv beeinflussen kann.

Auch hier stiessen wir wieder auf diverse Probleme, so benötigt zum Beispiel das Updaten des Displays sehr viel Akku und Rechungszeit, wodurch die gesamte Laufzeit beeinflusst wurde. Um dies zu minimieren, haben wir das Display in verschiedenen Sektoren aufgeteilt, um dann nur die betroffenen Sektoren upzudaten. Dies macht den Bildschirm deutlich effizienter. Eine weitere Schwierigkeit lag darin, dass die Akkudaten des M5 Stick 2 Plus sehr ungenau waren und grosse Schwankungen aufwiesen. Wir konnten gegen diese Schwankungen entgegenwirken, in dem wir ein Moving Average verwendet haben, um diese Auszugleichen.



Abbildung 3.6: Standard Ansicht

Auf der Standard Ansicht des Displays, kann man den aktuellen Modus sehen, indem sich Click y Point gerade befindet, sowie auch die Uhrzeit, Akkustand und den Bluetooth Verbindungs Status.

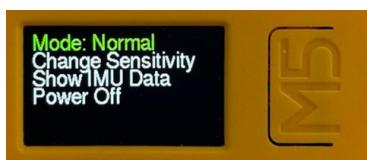


Abbildung 3.7: Menu

Durch das Klicken auf den Rotary Encoder, gelangt man in ein Menu. In diesem Menu kann man vom Normalen Modus in den Präsentationsmodus wechseln. Der Präsentationsmodus deaktiviert die Maussteuerung, damit man nicht ungewollte Klicks macht, wenn man beim Präsentieren gestikuliert.

Ebenfalls kann man sich die Daten des Gyroscops ansehen, die Maus Empfindlichkeit einstellen und das Gerät ausschalten.



Abbildung 3.8: Mausempfindlichkeit

Diese Funktion ermöglicht es dem Nutzer, die Empfindlichkeit der Maussteuerung anzupassen und diese somit auf die individuellen Bedürfnisse einzustellen.

4 Reflektion des Projektes

4.1 Herausforderungen

Im Laufe des Projektes stiessen wir immer wieder auf kleinere Hürden, wie zum Beispiel die bereits beschriebenen Limitationen unseres ersten Prototyps. Viele unserer grösseren Probleme hatten jedoch ihren Ursprung auf der Software Seite des Projektes. So war eines unserer grossen Probleme die Library des M5Stick Plus 2, M5Unified, die einen Bootloop verursachte. Da diese Library nur schlecht dokumentiert war, konnten wir lange Zeit den Fehler auch nicht finden und haben viele verzweifelte Stunden versucht dieses Problem zu beheben. Es hat sich rausgestellt, dass M5Unified FastLED nutzt. Vor 2 Monaten gab es allerdings ein Breaking Change von FastLED gab, der anscheinend noch nicht von M5Unified berücksichtigt wurde. Die Lösung war dann auch ziemlich einfach, da wir einfach nur eine ältere Version der Library benutzen mussten.

4.2 Erfolge

Wir hatten natürlich auch viele Erfolge, so war es jedes mal, wenn ein Stück Code funktionierte wieder eine grosse Freude. Grundsätzlich galt: Je länger man sich an einem Problem die Zähne ausgebissen hat, desto besser war das Erfolgsgefühl, wenn es denn endlich funktionierte.

Auch auf der Hardware Seite hatten wir Erfolgsmomente, so freuten wir uns jedes mal, wenn wir eine neue Iteration unserer Uhrenhalterung 3D-gedruckt haben und diese dann perfekt für unsere Komponenten passte. Dieser Prozess konnte besonders ärgerlich sein, wenn die Komponenten nicht gepasst haben, da man dann die Halterung neu Zeichnen musste und erneut ausdrucken musste, was wiederum eine gewisse Zeit in Anspruch nahm, vor allem weil unsere Gruppe nicht die Einzige war, die den 3D Drucker genutzt haben.

4.3 Erkenntnisse

Ein wichtiges Learning war, dass verschiedene Betriebssysteme das Gerät unterschiedlich interpretierten: Auf Linux (und damit Android) funktionierte alles wie geplant, während auf macOS das Klicken und auf Windows die Mauszeigersteuerung nicht korrekt funktionierte. Außerdem haben manche ältere Windows Laptops keine Unterstützung für BLE (Bluetooth Low Energy), weswegen das Gerät nicht sichtbar ist.

4.4 Mögliche Erweiterungen

Denkbare Erweiterungen umfassen eine verbesserte Kompatibilität für alle Betriebssysteme, oder zusätzliche Interaktionen, wie z.B. eine erweiterte Motion-Control für Zoom, Scroll und Drehbewegungen.

5 Fazit

Der Prototyp bietet eine innovative Möglichkeit zur kabellosen Steuerung eines Computers durch Handbewegungen. Die Herausforderungen während der Entwicklung waren lehrreich und führten zu einem wertvollen Verständnis über Hardware- und Software-Kompatibilität. Trotz der verschiedenen Herausforderungen hatten wir viel Spass beim Bearbeiten des Projektes und sind im Grossen und Ganzen sehr zufrieden mit unserem Prototyp.

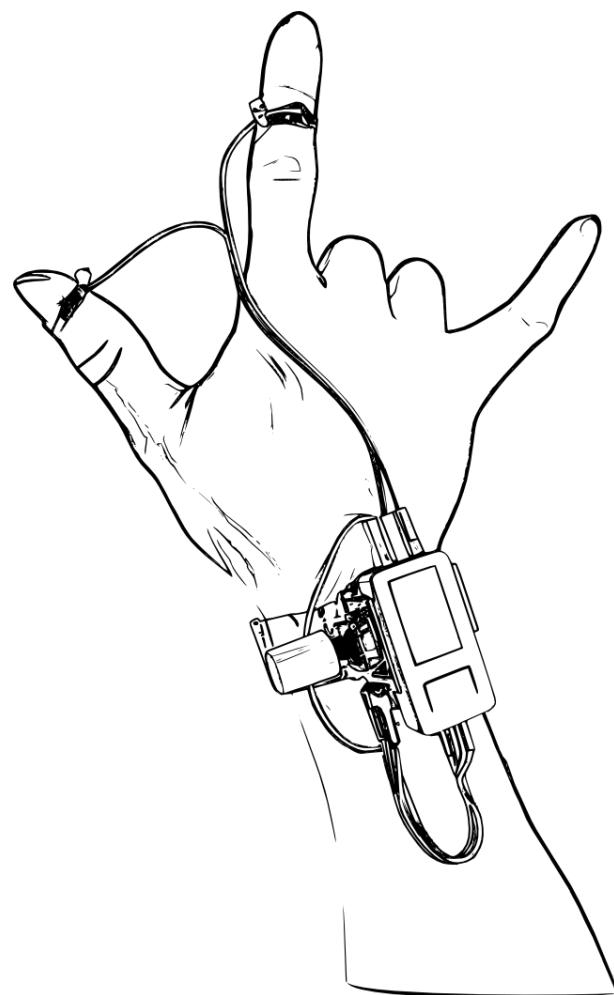


Abbildung 5.1: Click y Point - Logo

A Anhang

A.1 Github Repository

Unser gesamter Code kann in folgendem Repository gefunden werden:

Unser Code: [GitHub - Main Branch](#)

A.2 Produkt Video

Unser Produkt Video mit Pitch, detaillierter Erklärung und Behind the Scenes kann hier gefunden werden:

Unser Video: [Youtube - Produkt Video](#)