Wahlpflichtmodul-Labor Wintersemester 2023/24

Laborbericht

über das Projekt „Gleichgewichtstrainer“

im Modul „Internet of Things – Smart Sensor Systems”

Eingereicht durch:

Anh Quoc Nguyen
1397466

Betreuung von Prof. Dr. Bergbauer

Frankfurt University of Applied Sciences

Fachbereich 2: Informatik- & Ingenieurwissenschaften

Studiengang: Informatik (B.Sc.)

Inhaltsverzeichnis

Abbildungen

Tabellen

I. Einleitung

In diesem Projekt wird ein Fitnesstrainer entwickelt, der spezielle Funktionen und Spiele anbietet. Im Gegensatz zu normalen Fitnessuhren und Gadgets, die mehr für Bewegung und Aktiv-Leben fördern, fordert dieser Fitnesstrainer die Spieler dazu auf, komplett still zu halten, um ihre Körperspannung zu verbessern. Dieses Gadget bietet den Sportlern zwei Grundfunktionen: **„don't move"** und **„keep your balance".**

Bei der ersten Grundfunktion **„don't move"** ist der Spieler verpflichtet, möglichst still zu bleiben. Jede kleine Bewegung kostet einen Punktabzug von insgesamt zehn Punkten. Das Spiel ist beendet, wenn alle zehn Punkte abgezogen sind. Während des Spiels wird die Stoppuhr automatisch aktiviert und speichert am Ende jedes Spiels nur das beste Ergebnis.

Alternativ zur ersten Funktion kann der Spieler eine beliebige Position wählen und sein Gleichgewicht halten. Bei diesem Modus hat der Spieler die Freiheit, zwischen fünf Schwierigkeitsniveaus zu wählen. Jedes Niveau hat einen Toleranzwinkel für Auslenkung. Wenn der Spieler seine Position nicht halten kann und sich innerhalb des Toleranzbereichs bewegt (nähert sich dem maximale zugelassene Ablenkungswinkel), warnt der Gleichgewichtstrainer den Spieler durch ein Tonsignal. So kann der Spieler sofort zu seiner ursprünglichen Position zurückfinden. Je höher das Niveau, desto kleiner ist die zugelassene Ablenkung und desto kürzer der Zeitabschnitt, in dem der Spieler reagieren muss. Wenn er nicht rechtzeitig reagieren kann, verliert er sofort einen Punkt.

Niveau/ Toleranz	Winkel in grad	Zeit in Sekunden
Noob	Von 20° bis 60°	5s
Athletic	Von 20° bis 50°	4s
Champion	Von 15° bis 30°	3s
Legend	Von 10° bis 25°	2s
God-like	Von 5° bis 15°	1s

Tabelle 1: Toleranz-Winkel und Zeitabschnitt für die einzelnen Niveaus

Abbildung 1: Mögliche Positionen, mit den man üben kann

II. Theorie

Der „Gleichgewichtstrainer“ nutzt grundlegende physikalische Prinzipien und moderne Sensorik, um die Körperstabilität zu trainieren. Hierbei kommen folgende Komponenten und Technologien zum Einsatz:

Calliope mini Mikrocontroller

- Beschleunigungssensor: Erfasst die Bewegungen und Neigung des Geräts.
- Gyroskop: Misst die Drehbewegungen.
- Magnetometer: Bestimmt die Ausrichtung relativ zum Erdmagnetfeld.

Programmierung

- Microsoft MakeCode: Eine visuelle Programmierumgebung, die Blockprogrammierung sowie JavaScript unterstützt. Ermöglicht die einfache Implementierung der Spiele „Don’t move“ und „Keep your balance“.

Diese theoretischen Grundlagen bilden die Basis für die Entwicklung und Funktionalität des Fitnesstrainers, der sowohl für Bildungszwecke als auch für körperliches Training geeignet ist.

III. Projekt

1. Vorstellung der Hardware

1.1. Calliope mini Platine

Die Calliope mini Platine ist ein vielseitiges und benutzerfreundliches Microcontroller-Board, das speziell für den Einsatz im Bildungsbereich entwickelt wurde. Es eignet sich hervorragend, um Kindern und Jugendlichen die Grundlagen der Programmierung und Elektronik näherzubringen

1.2. Übersicht

OnBoard Hardware:

- Nordic nRF51822 Multi-protocol Bluetooth® 4.0 low energy/ 2.4GHz RF SoC
- 32-bit ARM Cortex M0 processor (16MHz)
- 16kB RAM 256 kB Flash
- Bluetooth Low Energy
- 5 x 5 LED-Matrix-Bildschirm
- Beschleunigungssensor, Gyroskop, Magnetometer (Bosch BMX055)
- MEMS-Mikrofon
- DC-Motortreiber (TI DRV8837)
- Piezo-Lautsprecher
- Programmierbare RGB-LED (WS2812b)
- 2 programmierbare Taster
- Serielle Schnittstelle (USB + konfigurierbare Anschlüsse)
- PWM-Ausgabe
- 4 Bananenstecker-/Krokodilklemmenanschlüsse
- 4 analoge Eingänge
- 8-11 Ein-/Ausgangsanschlüsse (je nach Softwarekonfiguration)
- SPI + I2C
- USB-Micro-B-Anschluss (Programmierung und Stromversorgung)
- JST-Batterieanschluss (3.3V)
- 4 Bananen-/Krokodilklemmenanschluss für 3.3V (Ausgang)
- 2 Grove-Steckverbinder (I2C + Seriell/Analog)

- NXP KL26 (USB und Stromversorgung)
- Flash-Programmspeicher (optional)

a. Eingabe und Ausgabe

- 5x5 LED-Matrix: Zeigt Bilder und Symbole, unterstützt Programmieren und visuelle Anzeige.
- Lautsprecher: Gibt Töne und Musik wieder.
- Knöpfe/ Tasten: Zwei programmierbare Tasten für Befehle oder Steuerung.
- Touch-Pins: Berührungssensitive Eingabegeräte.
- Logo Touch-Pin: Zusätzlicher Touch-Pin im Logo für weitere Eingaben.

b. Sensoren

- Lagesensor: Erkennt Ausrichtung und Bewegungen, ideal für Bewegungserkennung und Spiele.
- Lichtsensor: Misst Umgebungshelligkeit, reagiert auf Lichtveränderungen.
- Temperatursensor: Erfasst Umgebungstemperatur.
- Kompass: Bestimmt Himmelsrichtung, unterstützt Navigation.
- Funk: Drahtlose Kommunikation zwischen Calliope mini Geräten.
- Bluetooth: Drahtlose Verbindung zu anderen Geräten und Sensoren, ideal für IoT-Projekte.

c. System

- Grove: Schnittstellen für einfache Erweiterung mit Sensoren und Aktoren.
- Batterie Anschluss: Anschluss für externe Batterie.
- USB-Anschluss: Programmierung, Stromversorgung und Datenübertragung.
- Prozessor: Leistungsstarker ARM Cortex M0 für Programmausführung und Steuerung.
- Status LED: Zeigt Betriebszustand und hilft bei der Diagnose.
- Reset Taste: Setzt das Gerät zurück, startet Programme neu oder behebt Fehler.

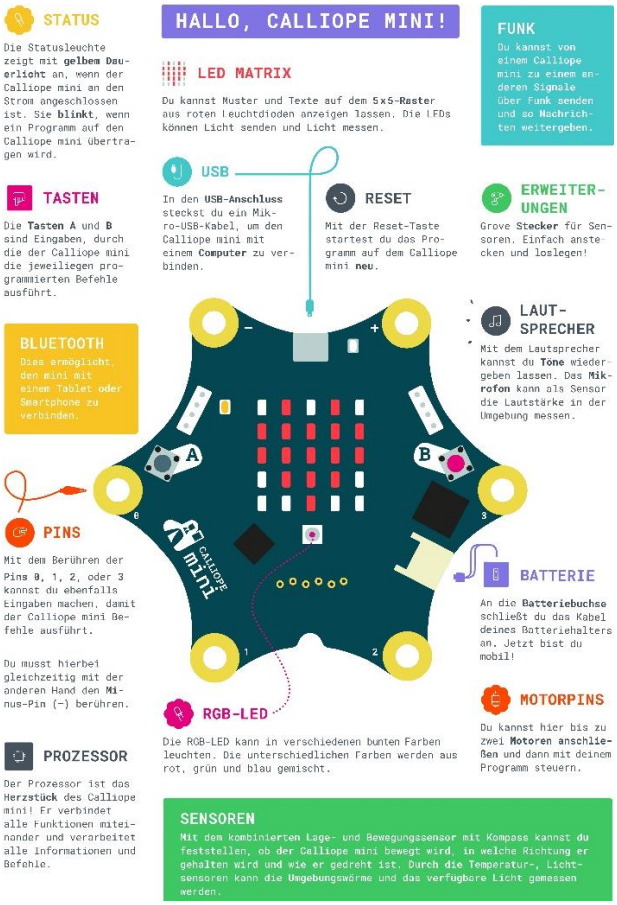

Abbildung 2: Allgemein Überblick der Calliope mini Platine

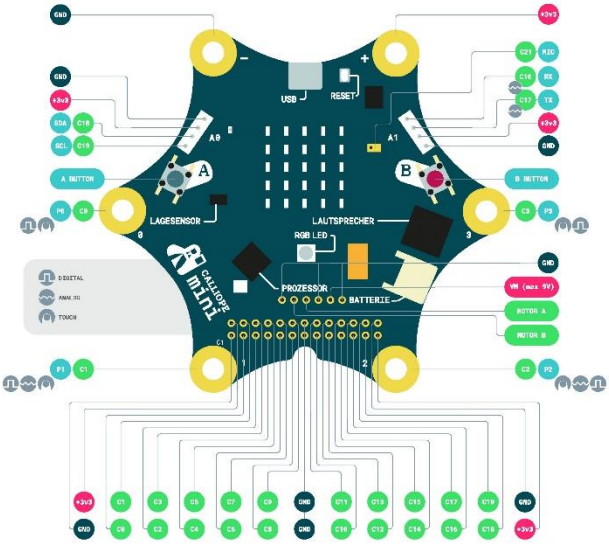

Abbildung 3: Pinbelegung der Calliope mini Platine

1.3. Programmierungsumgebung

MakeCode ist eine von Microsoft entwickelte Online-Entwicklungsumgebung, die es ermöglicht, Programme für Mikrocontroller wie den Calliope mini mithilfe einer grafischen Benutzeroberfläche zu erstellen. Sie ist besonders für Anfänger und den Bildungsbereich geeignet, da sie eine intuitive visuelle Programmierung bietet.

Der Calliope mini kann über PC, Android und iOS drahtlos programmiert werden. Nach dem Anschluss über USB oder Bluetooth wird der Calliope mini als USB-Datenträger verwaltet, auf dem Programme übertragen werden können. Diese Programme starten sofort nach dem Einschalten oder Neustart des Systems.

Unterstützte Entwicklungsumgebungen:

- Calliope mini-Editor: Einfache Webanwendung basierend auf Scratch.
- Microsoft MakeCode: Bietet eine visuelle Programmierung mit JavaScript, C++ und MicroPython.
- Open Roberta Lab (NEPO): Eine grafische Programmierumgebung.

Vorteile von MakeCode für den Calliope mini:

- Einfache Benutzeroberfläche: Drag-and-Drop-Programmierung.
- Integrierter Simulator: Testen von Programmen vor dem Hochladen.
- Vielfältige Programmiermöglichkeiten: Unterstützt Blockprogrammierung sowie JavaScript und Python.

Deswegen ist Makecode Blocks bzw. JavaScript für unseren Projekt ausgewählt. (Calliope, Makecode Calliope, n.d.)

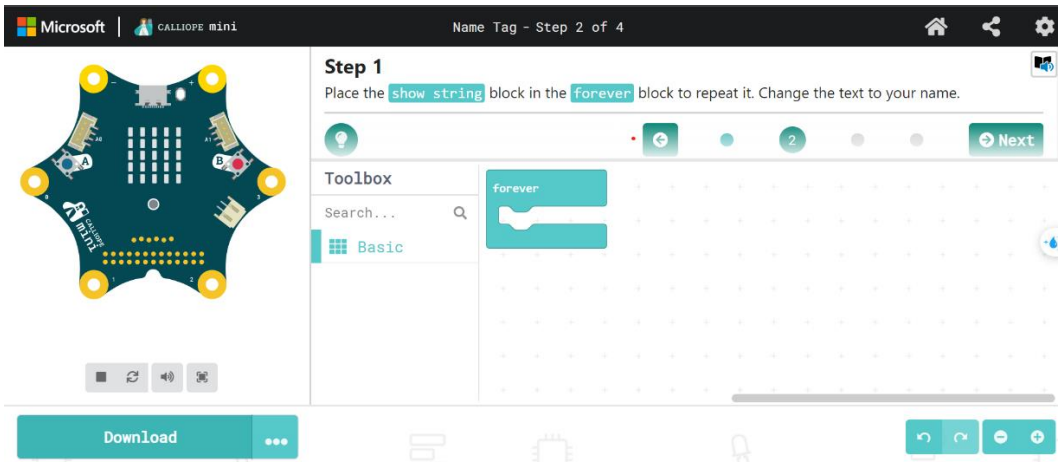

Abbildung 4: Die Schnittstelle des Makecode

1.4. Initialisierung des Programms im Simulator

Der Projekt wird mit dem Button „Neues Projekt“ über die Webseite von Makecode gestartet. (Calliope, Makecode Calliope, n.d.)

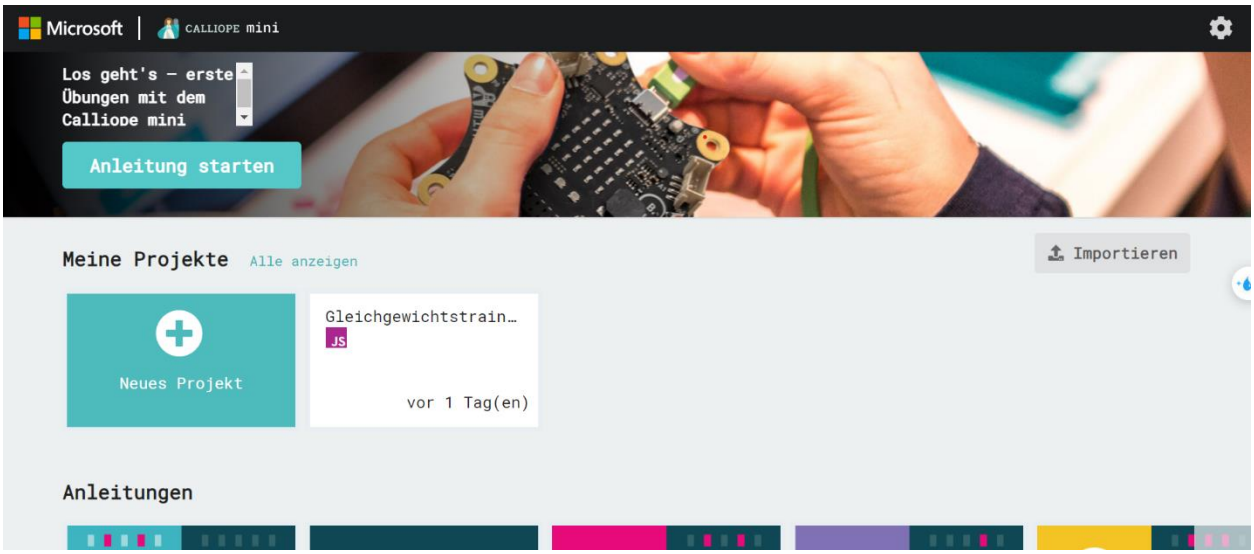

Abbildung 5: Der Startbildschirm des Makecodes

In diesem Projekt werden zwei Probleme in der Programmiersprache JavaScript gelöst.

Abbildung 6: Wechseln in die textbasierte JavaScript-Ansicht

2. Vorstellung der Software

Im folgenden Abschnitt werden nacheinander die beiden Probleme „don't move" und „keep your balance" gelöst. Jede Lösung besteht aus zwei Teilen: Flussdiagramm und Codierung im Makecode.

2.1. Die erste Funktion „Don't move"

a. Flussdiagramm

Bevor zum Programmierteil übergegangen wird, ist es äußert wichtig, ein Flussdiagramm zur Lösung des Problems zu erstellen. Dies erleichtert nicht nur das Verständnis des Programms, sondern auch die Identifikation möglicher Fehlerquellen und die Strukturierung des Codes.

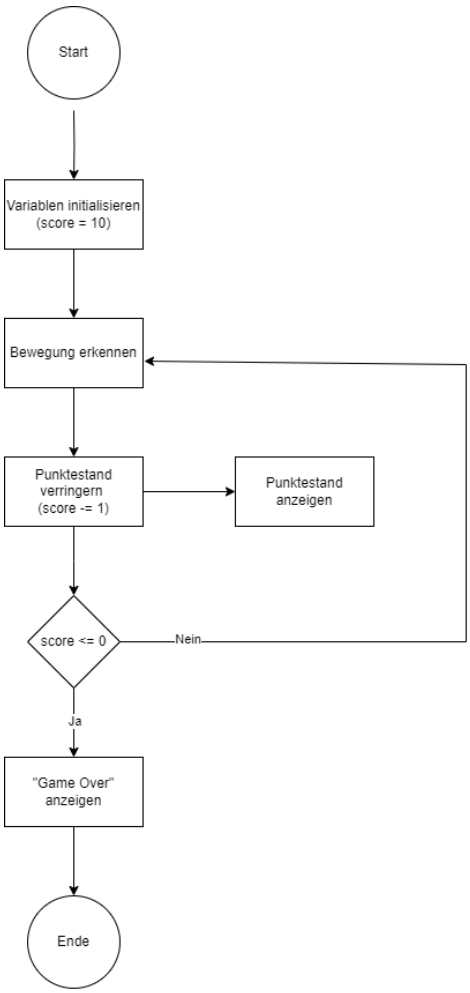

Abbildung 7: Flussdiagramm der Grundfunktion "Don't move"

b. Programmierung

Um diese Funktion zu programmieren, ist es sehr wichtig, die Dokumentation zu lesen. Einige integrierte Funktionen werden verwendet, die detailliert in der Dokumentation beschrieben sind. Dies hilft, ein besseres Verständnis der verfügbaren Werkzeuge und Methode zu erlangen und ermöglicht eine effektivere Implementierung der Lösung.

- Function basic.showNumber (Calliope, Zahl anzeigen, n.d.)
- Function input.onGesture (Calliope, Bei Geste, n.d.)
- Function basic.showString (Calliope, Text anzeigen, n.d.)
- Function input.runningTime (Calliope, Laufzeit, n.d.)

```
let elapsedTime = 0
let score = 10
let bestScore = 0
basic.showNumber(score)

input.onGesture(Gesture.Shake, function () {
    score -= 1
    basic.showNumber(score)
    if (score <= 0) {
        basic.showString("Game Over")
        elapsedTime = input.runningTime()
        basic.showString("Result: " + elapsedTime / 1000 + "s")
        if (elapsedTime < bestScore || bestScore == 0) {
            bestScore = elapsedTime
            basic.showString("Best Result: " + elapsedTime / 1000 + "s")
        }
    }
})
```

Abbildung 8: Calliope Mini JavaScript Programm "Don't move"

2.2. Die zweite Funktion „Keep your balance“

a. Flussdiagramm

Funktion „Keep your balance“ ist zwar schwieriger als die letzte Funktion, aber durch dieses folgenden Flussdiagramm wird Klarheit über die Lösung gewonnen.

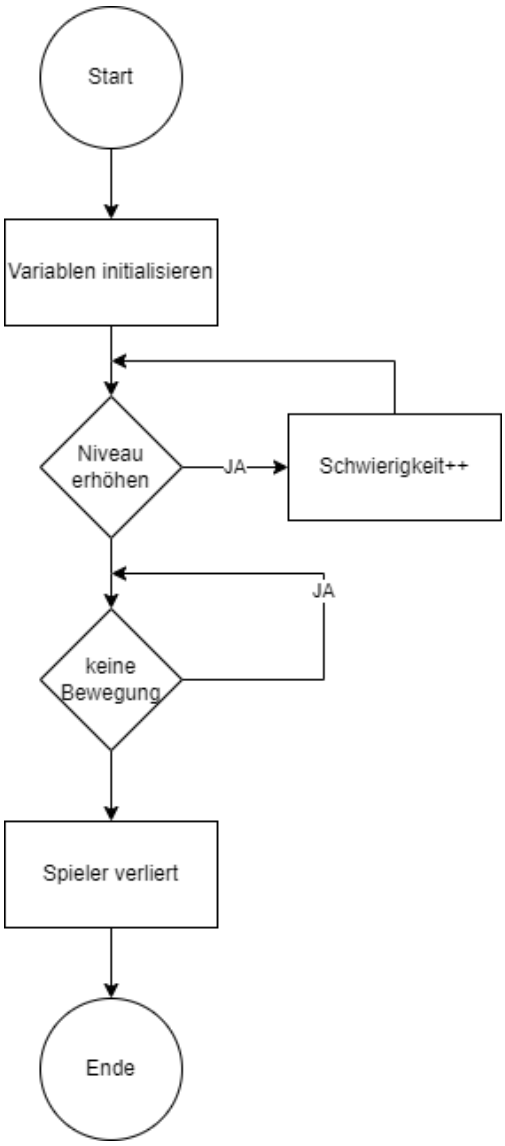

Abbildung 9: Flussdiagramm der Grundfunktion "Keep your balance"

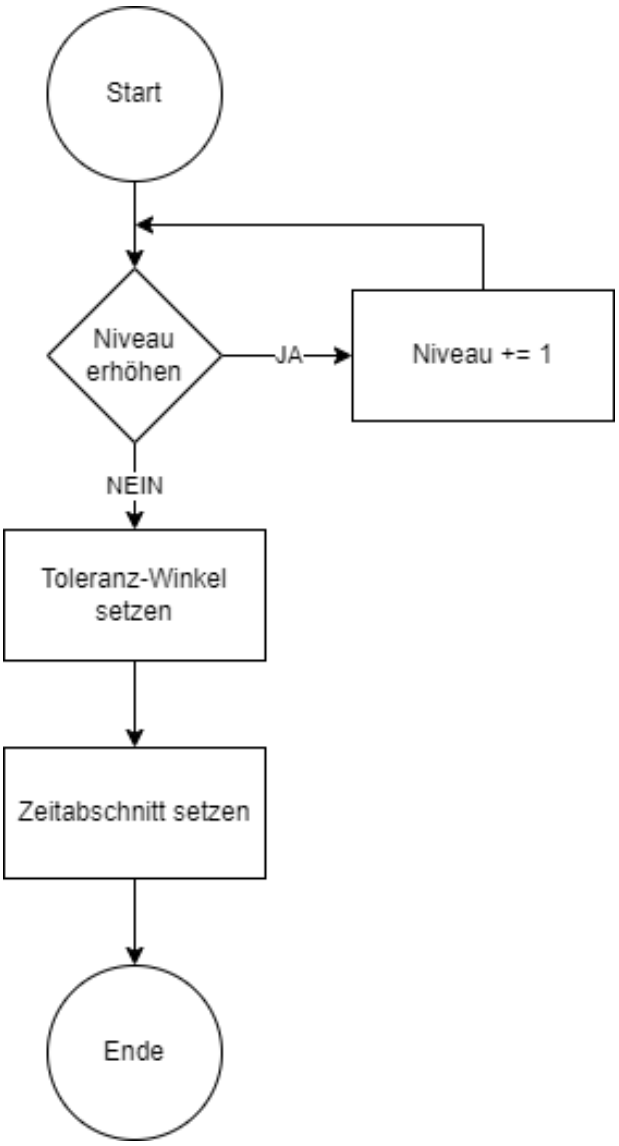

Abbildung 10: Flussdiagramm der kleinen Prozess "Niveau setzen"

b. Programmierung

Anhand der zwei Flussdiagramme wird es möglich, die Codierung zu implementieren. Im folgenden Teil werden alle detaillierten Probleme der beiden Flussdiagramme nacheinander gelöst.

Am Anfang werden alle notwendig benutzende globale Variable initialisiert.

```
let startGame = false
let pitch = 0
let rotation = 0
let level = 1
let minToleranceAngle = 0
let maxToleranceAngle = 0
let warningTime = 0
let warningLevel = 0
```

Abbildung 11: Programm der Funktion "Keep your balance" - Teil 1

Eine Funktion wird erstellt, die es ermöglicht, einen Toleranzwinkel für die Auslenkung sowie einen Zeitabschnitt abhängig vom eingestellten Niveau festzulegen.

```
// Function to set parameters based on the level
function setLevel (level: number) {
    if (level == 1) {
        maxToleranceAngle = 60
        minToleranceAngle = 20
        warningTime = 5
    } else if (level == 2) {
        maxToleranceAngle = 50
        minToleranceAngle = 20
        warningTime = 4
    } else if (level == 3) {
        maxToleranceAngle = 30
        minToleranceAngle = 15
        warningTime = 3
    } else if (level == 4) {
        maxToleranceAngle = 25
        minToleranceAngle = 10
        warningTime = 2
    } else if (level == 5) {
        maxToleranceAngle = 15
        minToleranceAngle = 5
        warningTime = 1
    }
}
```

Abbildung 12: Programm der Funktion "Keep your balance" - Teil 2

Jetzt wird das Spiel programmiert. Damit der Spieler jederzeit wieder die Schwierigkeit des Spiels auswählen kann, wird die Taste „A“ konfiguriert. Darüber hinaus wird die Taste „B“ so eingestellt, dass das Spiel gestartet werden kann.

```
// Event to increase the level when button A is pressed
input.onButtonPressed(Button.A, function () {
    if (level < 5) {
        level += 1
    } else {
        level = 1
    }
    basic.showNumber(level)
})

// Event to start the game when button B is pressed
input.onButtonPressed(Button.B, function () {
    setLevel(level)
    startGame = true
})
```

Abbildung 13: Programm der Funktion "Keep your balance" - Teil 3

Danach muss der Hauptteil des Spiels programmiert werden. Dieser fungiert als Prüfer, der überprüft, ob der Spieler Fehler macht. Wenn der Spieler im vorgegebenen Zeitabschnitt nicht still bleiben kann, wird auf dem Bildschirm der Satz „Lose“ angezeigt.

```
basic.showNumber(level)
// Main loop to start the game
basic.forever(function () {
    warningLevel = 0
    // Loop to run the game
    if (startGame) {
        pitch = Math.abs(input.rotation(Rotation.Pitch))
        rotation = Math.abs(input.rotation(Rotation.Roll))
        basic.showIcon(IconNames.Yes)
        // Check if the tilt exceeds the tolerance angle
        if (pitch > minToleranceAngle || rotation > minToleranceAngle) {
            basic.showIcon(IconNames.No)
            music.playTone(262, 1000)
            warningLevel = 1
        }
        basic.pause(warningTime)
        if (pitch > minToleranceAngle || rotation > minToleranceAngle) {
            warningLevel = 2
        }
        if (warningLevel == 2){
            basic.showString("LOSE")
            // End the game
            startGame = false
        }
    }
})
```

Abbildung 14: Programm der Funktion "Keep your balance" - Teil 4

IV. Messergebnisse

Nach der Implementierung und dem erfolgreichen Testen der Funktionen „Don’t move“ und „Keep your balance“ auf der MakeCode-Plattform wurden alle Anforderungen vollständig erfüllt. Die Ergebnisse der Tests sind wie folgt:

„Don’t move“ Funktion:

- Alle Bewegungen wurden präzise erkannt und führten wie vorgesehen zu Punktabzügen.
- Die Stoppuhr funktionierte einwandfrei und speicherte die besten Ergebnisse korrekt.

„Keep your balance“ Funktion:

- Die Auswahl der fünf Schwierigkeitsstufen funktionierte problemlos.
- Die Toleranzwinkel und Reaktionszeiten wurden exakt eingehalten:
    - Noob: Toleranzwinkel von 20-60° und Reaktionszeit von 5 Sekunden
    - Athletic: Toleranzwinkel von 20-50° und Reaktionszeit von 4 Sekunden
    - Champion: Toleranzwinkel von 15-30° und Reaktionszeit von 3 Sekunden
    - Legend: Toleranzwinkel von 10-25° und Reaktionszeit von 2 Sekunden
    - God-like: Toleranzwinkel von 5-15° und Reaktionszeit von 1 Sekunden

V. Auswertung

Insgesamt zeigen die Ergebnisse, dass der „Gleichgewichtstrainer“ effektiv als Trainingsgerät zur Verbesserung der Körperspannung und Balance eingesetzt werden kann. Die erfolgreiche Umsetzung und die präzisen Ergebnisse auf der MakeCode-Plattform belegen die Zuverlässigkeit und Funktionalität des entwickelten Systems.

VI. Zusammenfassung

Dieses Projekt demonstriert erfolgreich die Entwicklung eines innovativen Fitnesstrainers, der durch die Nutzung eines Calliope mini Mikrocontroller-Boards und der Programmierung in JavaScript realisiert wurde.

Die beiden Hauptfunktionen „Don’t move“ und „Keep your balance“ wurden detailliert beschrieben und implementiert, um den Spielern eine effektive Möglichkeit zu bieten, ihre Körperspannung und ihr Gleichgewicht zu trainieren.

Die Verwendung des Calliope mini und der MakeCode-Umgebung ermöglichte eine benutzerfreundliche und vielseitige Lösung, die sowohl für Bildungszwecke als auch für Fitnessanwendungen geeignet ist.

VII. Literaturverzeichnis

Bergbauer. (24. 06 2024). *Bergbauer: Internet of Things -Smart Sensor Systems- - semesterübergreifend.* Von campuas.frankfurt-university.de: https://campuas.frankfurt-university.de/course/view.php?id=1130 abgerufen

Calliope. (kein Datum). *Allgemein der Calliope Mini Platine*. Von https://calliope-mini.github.io/v10/ abgerufen

Calliope. (kein Datum). *Bei Geste*. Von https://makecode.calliope.cc/reference/input/on-gesture abgerufen

Calliope. (kein Datum). *Eingabe.* Von https://makecode.calliope.cc/reference/input/ abgerufen

Calliope. (kein Datum). *Grundlagen.* Von https://makecode.calliope.cc/reference/basic abgerufen

Calliope. (kein Datum). *Laufzeit*. Von https://makecode.calliope.cc/reference/input/running-time abgerufen

Calliope. (kein Datum). *Makecode Calliope*. Von https://makecode.calliope.cc/ abgerufen

Calliope. (n.d.). *TECHNICAL DETAILS OF THE CALLIOPE MINI*. Retrieved from https://calliope.cc/en/calliope-mini/tech-facts

Calliope. (kein Datum). *Text anzeigen*. Von https://makecode.calliope.cc/reference/basic/show-string abgerufen

Calliope. (kein Datum). *Zahl anzeigen*. Von https://makecode.calliope.cc/reference/basic/show-number abgerufen