TRAININGSPROGRAMM 42
GEMEINSAM FÜR UNSEREN FRIEDEN

Von: Brua Karim, Georg Settels, Vu Minh Hoang Nguyen

Datum: 23.06.2025

INHALT

INHALT 2
1. Allgemeines 3
1.1 Projektbezug 3
1.2 Story 3
1.3 Aufgabe der Spielerinnen und Spieler 3
1.4 Minispiele 3
2. Benötigte Komponenten 4
2.1 IR-Pistole (x2) 4
2.2 IR-Receiver Board (Alle Daten pro Zielfläche) 4
2.3 pyBadge Zentrale 4
3. AufbauAnleitung 5
4. Installationsanleitung 8
5. Spielanleitung 9
6. Bekannte Probleme & Fixes 10
6.1 Das Interferenzproblem 10
6.2 Das Reset Problem 10

1. ALLGEMEINES

1.1 Projektbezug

Gitlab: https://git.rwth-aachen.de/hoang.nguyen/multimodal-madness-gruppe5

1.2 Story

Der neunte Sektor im äußeren Bereich des Imperiums steht unter Beschuss. Eine Gruppe von Systemfeindlichen hat sich zusammengeschlossen, um den Sektor in Anarchie und Chaos zu werfen.

Das Imperium hat daher neue Trainingsprogramme ins Leben gerufen. Auf der Suche nach Ruhm und Ehre hast du dich Trainingsprogramm 42 angeschlossen, um Frieden und Sicherheit in der Galaxis weiterhin zu gewährleisten.
Doch bis dahin ist es noch ein langer Weg, denn du bist nur einer unter Vielen. Zeige mithilfe des neuen Trainingssystem, dass du mehr bist. Steige über Andere und dich selbst hinaus und beweise, dass du der perfekte Kämpfer für unseren Frieden bist.

1.3 Aufgabe der Spielerinnen und Spieler

Die Spielerinnen und Spieler sind in einem Trainingsprogramm und müssen gegen Mitspieler und gegen ihre eigenen Grenzen antreten. Durch Minispiele sollen körperliche und geistige Fähigkeiten gestärkt werden, um die Kadetten auf einen kommenden galaktischen Krieg vorzubereiten. Dazu erhalten zwei Spieler IR-Pistolen, mit denen man auf IR-Receiver-Boards schießen kann. Die anzuschießenden Boards werden mit LEDs markiert.

1.4 Minispiele

- **StandOff(30-60 Sekunden)**

  Beim StandOff Spiel geht es darum, dass eine Zielfläche grün aufleuchtet und man Diese schneller abschießt als der Gegner. Dies gibt einen Punkt. Der erste Spieler mit 5 Punkten gewinnt das Spiel.

- **Tag-A-Mole (45-50 Sekunden)**

  Das Spiel beginnt mit einem Ziel, welches alle 2 Sekunden wechselt. Alle Sieben Sekunden wird diese Zeit verkürzt und das Ziel wechselt schneller. Der Spieler, der dieses zuerst abschießt, kriegt einen Punkt. Der Spieler mit mehr Punkten gewinnt nach 45 Sekunden das Spiel. Bei Gleichstand wird noch ein entscheidender Punkt gespielt.

2. BENÖTIGTE KOMPONENTEN

2.1 IR-Pistole (x2)

- IR-LED TSAL6200
- Arduino Nano 3.3V / 5V
- Button
- Piezo-Speaker
- Laser-Modul KY-008
- Vibrationsmotor
- USB-B zu USB 3.0 Kabel
- 3x M3 20mm Schrauben für Plastik
- 2x Transistoren 2N2222A
- Halbes Breadboard GND/VCC Rail
- Jumper Kabel
    - F-to-M : 5
    - F-to-F : 7
- Widerstände
    - 1x 1kΩ
    - 1x 100Ω
    - 1x 150Ω
    - 1x 10kΩ

2.2 IR-Receiver Board (Alle Daten pro Zielfläche)

- IR- Receiver TSOP4840
- Arduino MKR WIFI 1010
- 9-Volt Batterie
- 9-Volt Batterie Clip Anschluss
- Schalter
- RGB-LED
- Kleines oder zerschnittenes Breadboard
- 1m langer Schrumpfschlauch
- Jumper Kabel
    - M-to-M: 3
    - F-to-M : 8
    - 3 1m lange M-to-M Kabel
- Widerstände
    - 1x 150Ω

2.3 pyBadge Zentrale

- pyBadge
- Breadboard
- 1x 3,3V LiPo Akku
- 1x Button
- 1x Neopixel LED-Strip
- Jumper Kabel
    - M-to-M: 8

3. AUFBAUANLEITUNG

Gun (2x)

Alle Verbindungen (außer zu den Arduino PINs) sollten am besten gelötet werden, da die Verbindungen beim Spielen ungewollt getrennt werden können.

1. Drucke die gegeben 3D-Modelle
2. Wir arbeiten jetzt im Teil der Gun mit dem rechteckigen Loch im Gehäuse (Linkes Teil)
3. Lege ein 5V Rail und GND Rail in den großen Bereich der Gun.
4. Lege den Arduino Nano in den Handgriff Teil. Die Pins sollten in „Schussrichtung“ zeigen. Der USB-Anschluss soll nach außerhalb der Gun zeigen.
5. IR-LED (TSAL6100):

   Verbinde den längeren Anschlusspin (-) an GND, den kürzeren über einen 100Ω Widerstand an PIN 2 des Nanos.
6. Laser-Modul (KY-008):

   Verbinde „S“ über einen 150Ω Widerstand an PIN 3 des Nanos. Der rechte Pin des Lasers muss an GND. Der mittlere Pin bleibt unangeschlossen.
7. Button:

   Teile ein Kabel in zwei Wege und verbinde ein Ende an den Button. Verbinde einen Weg an PIN 4 des Arduinos, den anderen über einen 10kΩ Widerstand an GND.

   Auf derselben Seite des Buttons, verbinde den Pin an das 5V Rail.
8. Vibrationsmotor

   Verbinde…

   - das rote Kabel an das 5V Rail.
   - das blaue Kabel an den Collector des Transistors
   - parallel dazu eine Diode mit der Anode zum Collector und die Kathode zum 5V Rail
   - die Basis des Transistors über einen 1kΩ Widerstand an PIN 5 des Arduinos
   - den Emitter des Transistors an GND
9. Piezo-Lautsprecher

   Verbinde einen PIN des Lautsprechers an PIN 6 des Arduinos, das andere an GND.

Pybadge-Zentrale (x1)

Für Einfachheit und Modularität empfehlen wir hierfür ein Breadboard zu nutzen.

10. LED-Strip

    Verbinde…

    - GND des LED Strips an GND des Arduinos
    - 5V des LED Strips an den USB PIN des Arduinos
    - D_{in} (mittlerer PIN) des LED Strips an PIN 5 des Pybadges

11. Button

    Verbinde einen PIN des Buttons an GND, den anderen an PIN 9 des Pybadges.

12. Receiver Boards (Aufbau weiter unten)

    Erstelle zur Modularität ein „SDA-Rail“ auf dem Breadboard, verbinde den SDA PIN des Pybadges an dieses. Wiederhole dies für SCL.

    Jetzt können beliebig viele Receiver Boards an das SDA-, SCL- und GND-Rail verbunden werden.

Receiver Board (beliebig viele)

Wir empfehlen die Nutzung eines GND/5V Rail zur Einfachheit. Wir verwenden für jedes Receiver Board einen neuen Arduino.

13. Pybadge Verbindung

    Verbinde SDA, SCL und GND des Arduinos an die jeweiligen Rails des Pybadges. Dazu sollten die langen Kabel genutzt werden (optimal >1m)

14. IR-Receiver (TSOP4840)

    Wir betrachten den Receiver von hinten (flache Seite)

    Verbinde…

    - den linken PIN (V_S) des Receivers an das 5V Rail
    - den mittleren PIN (GND) an das GND Rail
    - den rechten PIN (OUT) an PIN 2 des Arduinos

15. RGB LED

    Wir betrachten die LED von der Seite, wo der längste Anschlusspin auf der linken Hälfte ist.

    Verbinde…

    - den 1. PIN (=Rot) an PIN 3 des Arduinos
    - den 2. PIN (=Gemeinsame Anode) über einen 150Ω Widerstand an 5V
    - den 3. PIN (=Grün) an PIN 5 des Arduinos
    - den 4. PIN (=Blau) an PIN 6

16. Powerswitch (Kippschalter + 9V Batterie + Batterieclip)

    Verbinde…

    - einen der PINs des Schalters an das VIN des Arduinos
    - den anderen PIN an die Plusseite eines Batterieclips
    - die Minusseite des Batterieclips an GND
    - die 9V Batterie an den Batterieclip

Die Verbindungen zur RGB LED und zum Batterieclip sollten angelötet werden, da diese oft zu Wackelkontakten neigen.

4. INSTALLATIONSANLEITUNG

17. Verbinde alle Teile nach der Aufbauanleitung
18. Lade alle hinterlegten Code Dateien herunter
19. Schalte den Strom des Receiver Boards aus und verbinde den Arduino über USB an einen PC. Ändere die Adresse unter „SLAVE_ADDRESS“ zur jeweiligen in der Box beschrifteten Adresse. Wiederhole diesen Schritt für jeden Arduino.
20. Verbinde den Pybadge per USB. Drücke zweimal schnell den Reset Knopf. Lade die Datei „adafruit-circuitpython-pybadge-de_DE-9.2.7.uf2“ in das Laufwerk des pyBadges. Nun sollte man auf dem Laufwerk Circuitpython haben.
21. Legt nun einen Ordner names sfx an und die fünf sfx Dateien in diesen Ordner hochladen.
22. Ändere im Code die Variable „SLAVE_COUNT“ zu der jeweiligen Anzahl an Receiver Boards. Schreibe in die Liste „slave_index“, die Adressen, die auf die Arduinos geschrieben wurden.
23. Verbinde die Gun per USB-B zu USB. Ändere im Code die Variable „PLAYER“ zu 1 (Orange Waffe) oder 2 (Blaue Waffe). Schreibe den Code auf die Waffe und wiederhole den Schritt für die andere Gun.
24. Drücke den Reset Knopf für jeden Receiver Arduino. Warte bis alle LEDs weiß sind, dann kann der Pybadge Reset Knopf gedrückt werden. Der Reset ist fertig, wenn alle LEDs lila sind.

Mögliche Fehlerquellen in der Installation:

- Wenn der Arduino in der Arduino DIE nicht erkannt wird, dann gibt es drei häufige Fehlerquellen. 1. Entweder muss der Pybadge noch mit Strom per USB oder Batterie versorgt werden 2. SDA, SCL müssen entfernt und der Arduino resettet werden. 3. Das benutzte USB-Kabel ist kein Datenkabel.
- Wenn einer der LEDs nach Pybadge Reset noch weiß ist, dann kann der Pybadge erneut resettet werden. Danach müsste die übrige LED weiß werden.
- Wenn die LED rot bleibt, dann muss die Batterie ausgetauscht werden

5. SPIELANLEITUNG

Nachdem ihr die Anleitungen ausgeführt habt, solltet ihr nun im Menü sein. Dies bedeutet, dass zwei LEDs Lila leuchten sollten. Falls dies nicht der Fall ist oder eine oder mehrere Zielflächen in einem tiefen Rot aufleuchten, solltet ihr nochmals den RESET-Button drücken (Zur Not auch ein paar Mal). Falls sich das Problem dadurch immer noch nicht löst, versucht erst einmal den Strom auszumachen und nochmals anzumachen. Falls das Problem dadurch nicht gelöst wird, müsst ihr direkt den Reset Knopf des Arduinos drücken oder die Batterie austauschen. **Sobald die Felder in gelb/orange und zwei Felder Lila leuchten, könnt ihr beginnen.**

Das Spiel selbst ist zu 2 Leuten spielbar. Jeder Spieler nimmt sich eine der beiden Waffen und stellt sich ca. 3-4 Meter von der Zielfläche hin. Im Folgenden ist Spieler1, der Spieler mit der orangenen Waffe und Spieler2, der mit der Blauen.

Spieler1 kann nun das zu spielende Minispiel auswählen. Wollt Ihr ein StandOff spielen, muss er mit seiner Waffe auf das rechte Feld schießen, welches Lila leuchtet. Wenn Ihr das Tag-A-Mole Game spielen wollt, schießt auf das linke lila Feld.

Beide Spiele funktionieren im Kern gleich. Nach dem der Start Sound endet, beginnt das Spiel. Es leuchtet immer ein Feld grün auf. Dieses Feld muss getroffen werden. Der Spieler, der zuerst trifft, kriegt einen Punkt. Auch hört ihr einen Sound und seht welcher Spieler dieses Feld getroffen hat.
Unterschiede:

StandOff: Beim StandOff Spiel leuchtet eine LED auf und geht auch erst aus, wenn ein Spieler trifft. Das Spiel endet sobald ein Spieler fünf Punkte hat. Dieser Spieler hat in dem Moment gewonnen. Punkte werden direkt auf dem LED-Strip über der Box angezeigt.

Tag-A-Mole: Beim Tag-A-Mole leuchtet eine Zielfläche nur kurze Zeit auf und wechselt danach. Die Zeitspanne wie lange eine Zielfläche leuchtet, wird dabei kürzer. Wer ein Feld trifft, kriegt einen Punkt. Das Spiel endet immer nach 45 Sekunden. Der Spieler mit mehr Punkten gewinnt nach dieser Zeit. Punkte werden hierbei als Prozente auf dem LED-Strip angezeigt.

Jeder Spieler hat zudem fünf Schüsse bevor das Magazin nachgeladen werden muss. Dies passiert automatisch und ihr könnt einen kurzen Moment nicht schießen. Also nicht durchfeuern!

Viel Spaß beim Spielen

6. BEKANNTE PROBLEME & FIXES

6.1 Das Interferenzproblem

Wir benutzen IR-LEDs, welche eine Bitsequenz über das Protokoll NEC entsenden, welche daraufhin von einem IR-Receiver gelesen wird. Wir wollten die Spieler damit unterscheiden, dass beide jeweils unterschiedliche Sequenzen schicken. Dabei ist uns aber erst sehr spät aufgefallen, dass Interferenzen entstehen, wenn beide Spieler zur gleichen Zeit schießen, da NEC ungefähr 100ms das Signal entsendet und sich beide Wellen somit gegenseitig beeinflussen und die Bitsequenz abändern.
Der Fix war am Ende, dass wir eine Waffe nur sehr kurz schießen lassen, sodass diese definitiv richtig sendet, wenn sie zuerst schießt und die zweite Waffe weiterhin normal schießen lassen, sodass bei einer Abänderung der Bitsequenz klar ist, dass diese Waffe zuerst geschossen wurde.

6.2 Das Reset Problem

Bei einem Fehler sowie beim Start des Spiels musste man alle Arduino MKR sowie den pyBadge einmal resetten. Auch konnte man das Spiel nicht mittendrin abbrechen und ins Menü zurückkehren. Die Lösung war ein Interrupt, welcher auf Knopfdruck aktiviert wurde und zentral vom pyBadge aus die Arduinos resettet hat. Dieser hatte aber als Problem, dass manchmal die Arduinos nicht resettet wurden, sondern vom Datenbus gekappt wurden und man alles nochmal resetten musste.
Jeder unserer versuchten Fixe hat das Problem nicht komplett gelöst bekommen. Die beste Möglichkeit war einfach alle einzelnd zu resetten. Sobald das Spiel nämlich einmal funktioniert, tritt der Fehler auch nicht mehr oder nur sehr selten auf.