

Ostbayerische Technische Hochschule Amberg-Weiden
Fakultät Elektrotechnik, Medien und Informatik

Studiengang Medieninformatik

Physical Computing

Projekt **Wire & Warriors**

von

Berkay **Yurdagül**

Manuel Pickl

Bearbeitungszeitraum: von 26. April 2023
bis 14. März 2024

1. Prüfer: Prof. Dipl.-Des. Martin Frey

2. Prüfer: Prof. Dr.-Ing. Ulrich Schäfer

3. Prüfer: Prof. Dr.-Ing. Gerald Pirkl

Inhaltsverzeichnis

1 Einführung und Idee	1
1.1 Spielkonzept eines Escape-Room	1
1.2 Spielkonzept vom Heißen Draht-Spiel	2
1.3 Spielkonzept von Wire & Warriors	2
1.3.1 Spielregeln	3
2 Konzeptionierung des Systems	4
2.1 Modellierung des 3D-Design	7
3 Hardware Umsetzung	11
3.1 Prototypen	12
3.2 Realisierung des Projekts	15
4 Software Umsetzung	18
4.1 Entwicklungsumgebung	18
4.2 Programmiersprache	18
4.3 PlatformIO	19
4.4 Komponenten	19
4.4.1 main	19
4.4.2 game	20
4.4.3 gameConstants	20
4.4.4 lights	21
4.4.5 logging	21
4.4.6 motor	22
4.4.7 pins	22
4.4.8 touch	23
4.5 Asynchronität der Software	23
4.5.1 Beispiel	23
5 Ausblick und Fazit	25
5.1 Ausblick	25
5.2 Fazit	26
Literaturverzeichnis	27

Abbildungsverzeichnis	27
Quellcodeverzeichnis	29

Symbole, Formelzeichen und Einheiten

Kapitel 1

Einführung und Idee

Das Spiel 'Wire & Warriors' ist ein innovative Spiel, dass die Intensität eines Escape Rooms mit der Geschicklichkeit eines 'Heißen Draht'-Spieles vereint. Dieses Spiel stellt eine neue Herangehensweise an das klassische Konzept dar, indem es Teamarbeit und Geschicklichkeit miteinander verbindet.

1.1 Spielkonzept eines Escape-Room

Ein Escape-Room basiert auf dem Prinzip der Live-Action-Rätselspiele, bei denen Spieler in einem speziell gestalteten Raum oder einer Umgebung eingeschlossen sind. In diesem Raum müssen verschiedene Rätsel gelöst werden, um zu entkommen. Im folgenden werden die Schlüsselkomponenten aufgelistet:

- **Thematische Umgebung:** Ein Escape-Room besitzt ein Thema und zusätzlich eine Geschichte. Das Thema sowie die Geschichte bilden den Rahmen für die Rästel und Aufgaben.
- **Rätsel und Aufgaben:** Die Spieler im Escape-Room müssen eine Reihe von Rätseln lösen, um zu entkommen. Die Rästel bzw. Aufgaben können aus Beobachtungsaufgaben, Geschicklichkeit oder auch Teamarbeit bestehen. Häufig sind die Rästel thematisch in die Umgebung eingebettet und tragen zur Gesamtgeschichte bei.
- **Zeitlimit:** Die Spieler haben eine festgelegte Zeit (oftmals eine Stunde), um alle Aufgaben zu lösen und aus dem Raum zu entkommen. Dadurch soll Druck erzeugt werden und die Spannung des Spiels erhöhen.
- **Teamarbeit:** Escape-Rooms sind oft auf Teamarbeit ausgelegt, um gemeinsam zu rästeln und zu entkommen.
- **Interaktive Elemente:** Escape-Rooms nutzen technologische und mechanische Vorrichtungen, um das Spielerlebnis zu bereichern.

1.2 Spielkonzept vom Heißen Draht-Spiel

Das Heiße Draht-Spiel ist ein klassisches Geschicklichkeitsspiel, bei dem die Spieler einen Metallstab entlang eines gewundenen Metalldrahtes führen müssen, ohne diesen zu berühren. Die Hauptelemente dieses Spiels sind:

- **Grundkonzept:** Es wird einen Metallstab oder eine Schlaufe verwendet, um ihn entlang des geformten Drahtes zu bewegen. Berührt der Stab den Draht, gibt es in ein Signal (oft ein Geräusch oder Licht), das einen Fehler anzeigt.
- **Geschicklichkeit und Konzentration:** Das Spiel erfordert eine ruhige Hand und Präzision, um den Draht nicht zu berühren. Die Herausforderung besteht darin, den Stab gleichmäßig entlang des Drahtes zu führen.
- **Verschiedene Schwierigkeitsgrade:** Das Design des Drahtes kann variieren, um verschiedene Schwierigkeitsgrade zu bieten oder zusätzlich erhöht werden durch eine kleinere Schlaufen.
- **Wettbewerbs- und Zeitfaktor:** Es ist möglich die Spielzeit zu messen, um zu sehen, wie schnell ein Spieler das Ende erreicht hat. Dadurch ist es möglich den Wettbewerbsaspekt hinzuzufügen.
- **Einfachheit und Zugänglichkeit:** Eines der Hauptmerkmale des Heißen Draht-Spiels ist seine Einfachheit in Bezug auf die Regeln und die leichte Zugänglichkeit, was es zu einem beliebten Spiel für alle Altersgruppen macht.

1.3 Spielkonzept von Wire & Warriors

Das Spielkonzept von Wire & Warriors stellt eine Fusion der zwei Spielarten dar: des Escape-Rooms und des Heißen Draht-Spiels. Dadurch konnte ein innovatives Spielprinzip entwickelt werden, dass sowohl die kognitiven als auch die feinmotorischen Fähigkeiten der Spieler auf die Probe stellt. Die Hauptelemente des Spieles sind:

- **Integration von Escape-Room-Elementen:** Im Spiel Wire & Warriors müssen die Spieler dem Heißen Draht-Spiel entkommen. Dies ist nur möglich, indem sie durch die verschiedenen Levels fortschreiten.
- **Einbindung des Heißen Draht-Prinzips:** Im Kern des Spiels steht das Konzept des Heißen Draht-Spiels, bei dem Spieler einen Metallstab entlang eines verwickelten Drahtes führen müssen, ohne diesen zu berühren.
- **Innovative Levels und Herausforderungen:** Das Spiel erhöht den Schwierigkeitsgrad mit fortschreitenden Levels und einer Brücke die gemeinsam überwunden werden muss. Jedes Level bringt neue, komplexere Drahtkonfigurationen, was die Spieler dazu zwingt, ihre Geschicklichkeit zu verbessern.
- **Interaktive und immersive Erfahrung:** Technologische Elemente wie Licht- und Soundeffekte verstärken die Gesamterfahrung.
- **Anpassungsfähigkeit und Vielfalt:** Das Spiel ist so konzipiert, dass es variable Schwierigkeitsgrade gibt und dadurch die Schwierigkeit erhöht werden kann indem z.B. es eine kleinere Schlaufen zum spielen nutzen. Auch ist das gesamte Spiel modular gestaltet, sodass alle Spielemente jederzeit ausgetauscht werden können, um das Spielerlebnis anzupassen.

1.3.1 Spielregeln

Das Spiel beginnt, wenn beide Spieler ihre Schlaufen in der Startposition haben. Die Startposition ist erreichbar, indem beide Spieler ihre Schlaufe durch den Draht am Turm fädeln. Die Positionen sind gültig, wenn die Platten am Turm berührt werden und das grüne Licht am Turm aufleuchtet. Die erfolgreiche Positionierung wird durch einen Sound signalisiert, daraufhin können die Spieler mit dem Spiel beginnen.

Ziel des Spiels: Das Hauptziel ist es, die gegenüberliegende Seite zu erreichen, ohne dabei den Draht zu berühren. Sobald ein Spieler den gegenüberliegenden Turm erreicht, muss er die Platten am Turm erneut mit seiner Schlaufe berühren. Die erfolgreiche Positionierung wird erneut durch ein grünes Licht am Turm signalisiert.

Level und Herausforderung

- **Level 1:** Im ersten Level bewegt sich nur die zentrale Brücke des Spiels, während die Drähte auf der linken und rechten Seite statisch bleiben.
- **Level 2:** Im zweiten Level beginnen alle Drähte sich zu bewegen, allerdings in einer langsameren Geschwindigkeit.
- **Level 3:** Im dritten Level erhöht sich die Geschwindigkeit der Drähte an den beiden Türmen.

Lebenspunkte und Herzsystem: Das Spiel verfügt über ein Herzsystem, dass insgesamt aus sechs Herzen besteht. Jedes Mal, wenn ein Spieler einen Draht berührt, verliert das Team ein Herz. Die Herzen sind auf dem Spielbrett zusehen und werden durch rote LEDs signalisiert. Verliert man ein Herz, erlischt die entsprechende LED. In jedem neuen Level werden die Herzen zurückgesetzt auf sechs Herzen.

Fehler und Konsequenzen: Wenn die Schlaufe mit einem Draht in Berührung bleibt, zieht das Spiel kontinuierlich Herzen ab. Sind alle Herzen verloren, muss das Spiel neu gestartet werden.

Spielabschluss: Ist das Spiel erfolgreich abgeschlossen worden, erhalten die Spieler am Ende einen geheimen Code als Belohnung.

Kapitel 2

Konzeptionierung des Systems

Im Rahmen der Konzeptionierung des Systems wurden verschiedene Ideen entwickelt, um das Konzept vom Escape-Room und die Geschicklichkeit des Heißen Draht-Spiels zu kombinieren. Im folgenden werden sechs Konzept vorgestellt, die entwickelt und evaluiert worden sind.

Erste Konzept: Die Spieler des System konnten in verschiedene Rollen schlüpfen. Ein Spieler konnte die Schlaufe nutzen, um das Heiße Draht-Spiel zu spielen und der andere Spieler konnte das gesamte System um die x-Achse sowie die Drähte um die y-Achse rotieren lassen. Zusätzlich hat das System eine Stoppuhr, um die Zeiten zu messen, wie schnell das Spiel durchgespielt werden konnte. Dies sollte für den Wettbewerbsaspekt sorgen. In der Mitte des Systems wurde ein Chip angebracht, um das gesamte System zu steuern. Die Schwierigkeit für den Spiel mit der Schlaufe konnte durch die Größe der Schlaufe angepasst werden. Dieses Konzept wurde jedoch verworfen da sie das Wesen eines Escape Rooms nicht widerspiegeln konnte. Der Entwurf des Systems sah wie folgt aus:

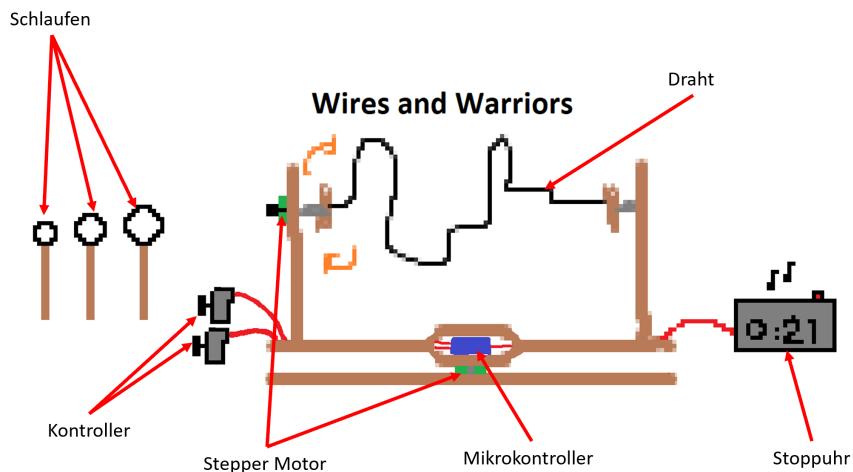


Abbildung 2.1: Visualisierung des ersten Konzepts

Zweite Konzept: Dieses System sollte drei separate Drähte nutzen: einer rechts, einer links und einer in der Mitte als Brücke. Die äußeren Drähte sollten sich um die Y-Achse und das Mittelstück um die X-Achse drehen können. Das Ziel ist es gewesen, die gegenüberliegende Seite zu erreichen. Die Herausforderung bestand darin, dass die Spieler sich koordinieren mussten, um das Spiel zu beenden. Dieses Konzept war nicht ausgereift, jedoch bietet sie ein vielversprechendes Grundkonzept für die weitere Entwicklung. Der Entwurf des Systems sah wie folgt aus:

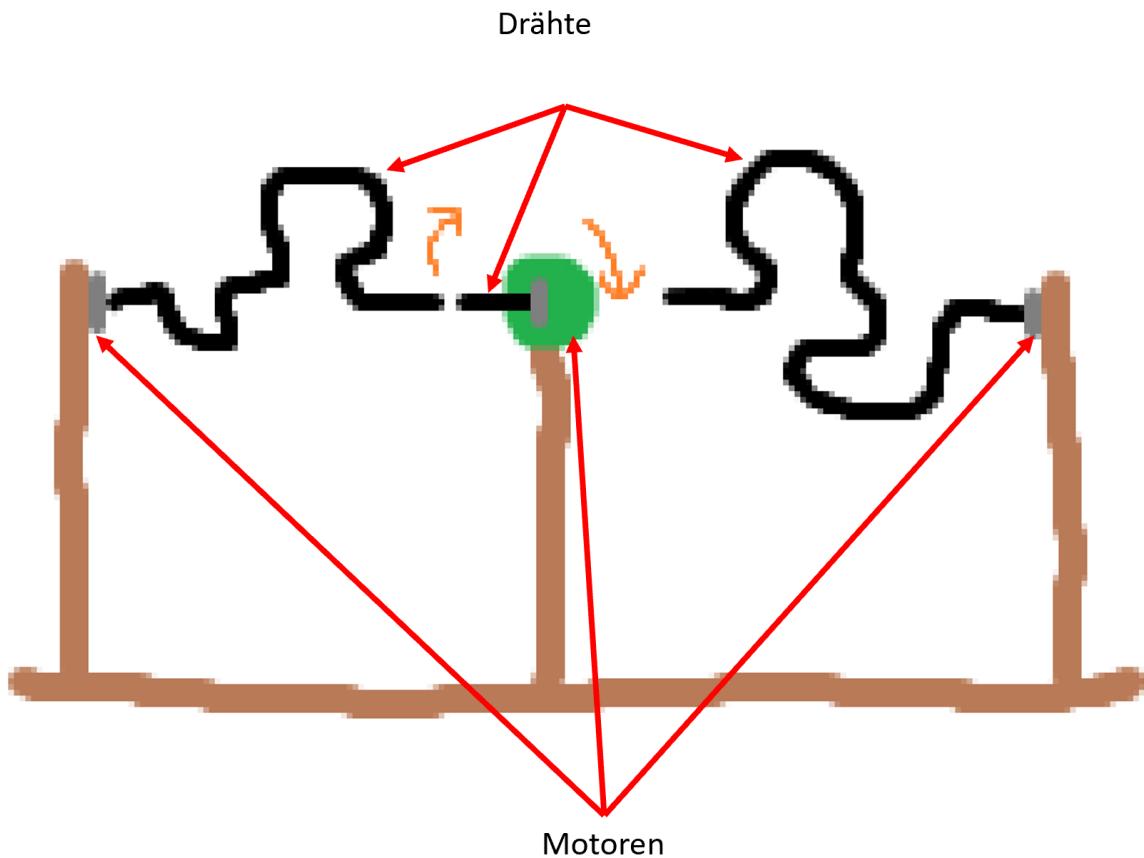


Abbildung 2.2: Visualisierung des zweiten Konzepts

Dritte Konzept: Das dritte Konzept sollte zwei Drähte besitzen. Diese wurden von den Seiten bis zur Mitte gespannt und am Mittelturm befestigt werden. Beide Spieler mussten, jeweils von den äußeren Seiten in die Mitte gelangen. Es zusätzlich möglich gewesen das ein dritter Spieler das System kontrollieren konnte über Kontroller. Diese Kontroller haben es ihm ermöglicht, die zwei Drähte rotieren zu lassen. Dieses Konzept entsprach nicht dem Charakter eines Escape-Rooms und wurde verworfen. Der Entwurf des Systems sah wie folgt aus:

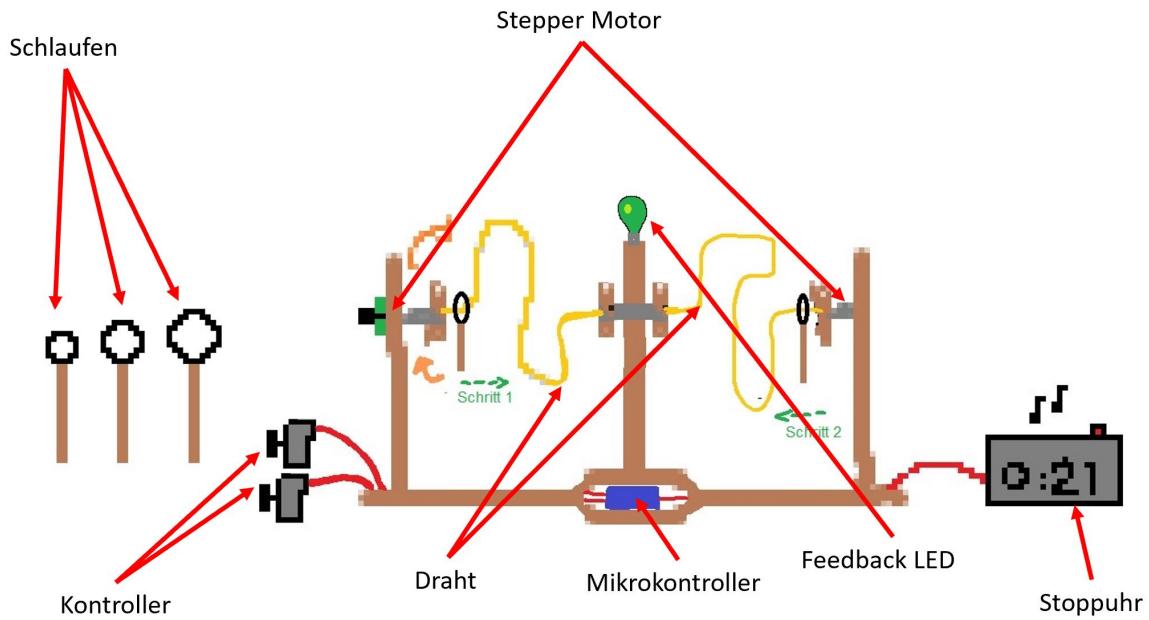


Abbildung 2.3: Visualisierung des dritten Konzepts

Vierte Konzept: Dieses Mal wurden die Türme auf einer Seite und die Brücke gegenüber platziert. Die Drahttürme rotieren um die Y-Achse und die Brücke um die X-Achse. Diese Idee wurde aufgrund mangelnder Nutzerfreundlichkeit schnell verworfen, jedoch wurde das Konzept des äußeren Gerüsts für das finale Design als Basis genutzt. Der Entwurf des Systems sah wie folgt aus:

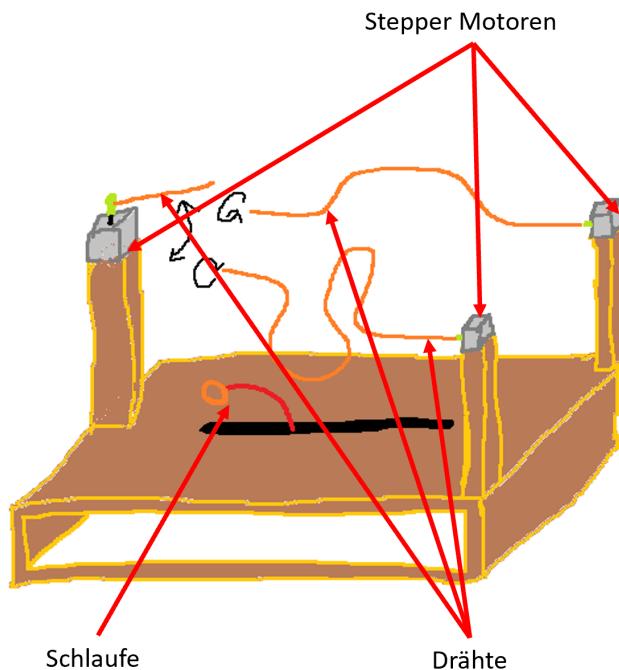


Abbildung 2.4: Visualisierung des vierten Konzepts

Fünfte Konzept: Das letzte Konzept ist eine Kombination aus dem dritten und zweiten Ansatz gewesen. Es gibt zwei Spieler, die gegen das System antreten, mit drei Drähten: links, rechts und ein Mittelstück. Die äußeren Drähte drehen sich um die Y-Achse, das Mittelstück um die X-Achse. Das System sollte Startknöpfe, Fehler-LEDs und Herzen zur Spieler-Rückmeldung besitzen. Die Spieler starten an den äußeren Türmen und müssen auf die andere Seite gelangen. Dieses Konzept war erfolgversprechend und wurde als Grundkonzept genommen und ausgearbeitet. Der Entwurf des Systems sah wie folgt aus:

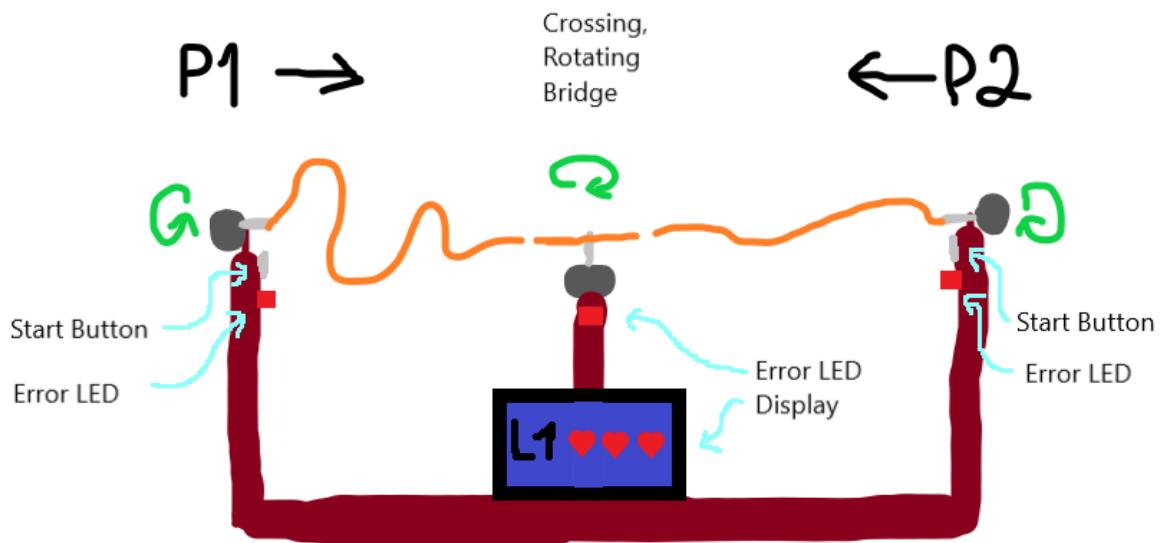


Abbildung 2.5: Visualisierung des fünften Konzepts

2.1 Modellierung des 3D-Design

Bei der Modellierung des 3D-Designs für Wire & Warriors wurde auf ein modulares System geachtet. Die Modularität war ein zentraler Aspekt des Designs, denn dadurch war es möglich die Komponenten für Wartung und Upgrade leicht zugänglich zu machen.

Das gesamte Gerüst des Spieles musste aufgrund der Größenbeschränkung der Holzplatten, jeweils 30x60 Zentimeter groß, durchdacht werden. Die Verwendung von Verzahnung im Modell war ein wichtiger Bestandteil. Diese Bauweise ermöglicht es, dass einzelne Platten fest ineinander greifen konnten, wodurch eine stabile Struktur ohne zusätzliche Verbindungselemente wie Schrauben oder Nägel zu schaffen. Diese Verzahnung sahen wie folgt aus:

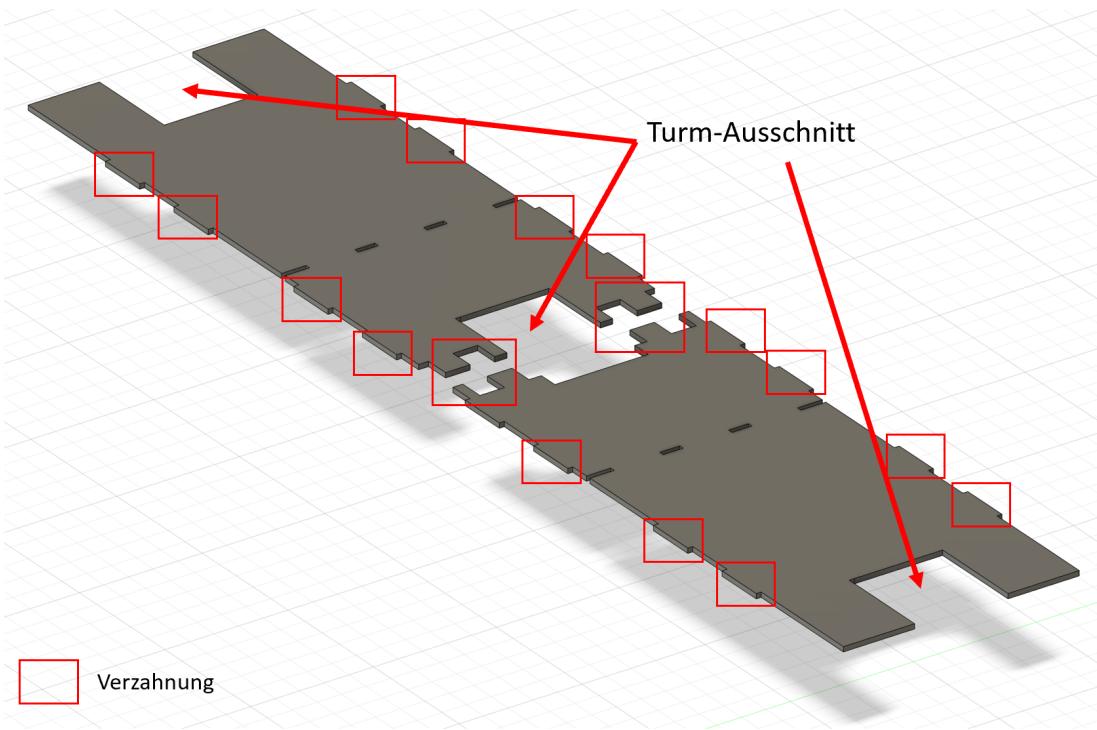


Abbildung 2.6: Verzahnung am Deckel

Zusätzlich wurde des gesamt Gerüst durch die Mittelstücke, die auch als Kabelführung gedient haben, verstärkt. Die Mittelstücke wurden im Deckel, in den Wänden und am Boden angebracht. Diese Mittelstücke sehen wie folgt aus:

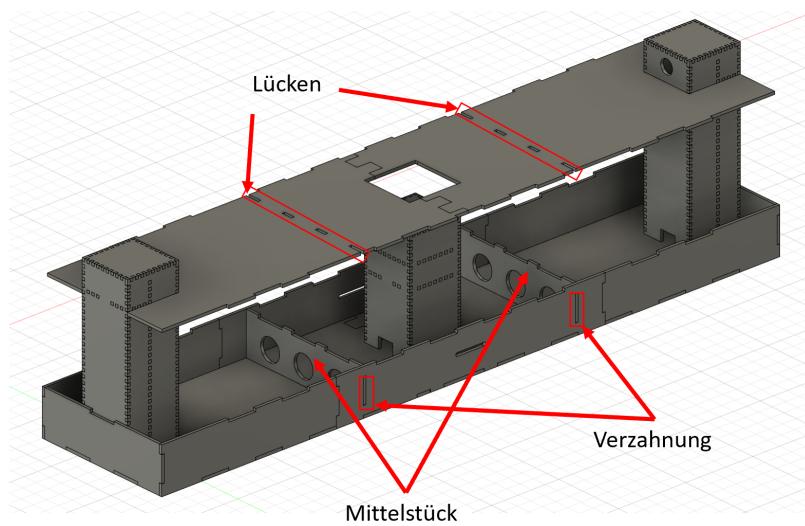


Abbildung 2.7: Stabilisierung des Gerüsts durch Mittelstücke

Die Türme des Spieles wurden in das Gerüst eingesetzt und wurden durch die Deckel befestigt. Jeder Turm hatte einen Ausschnitt für den Stepper-Motor und eine Ablage. Die Türme sahen wie folgt aus:

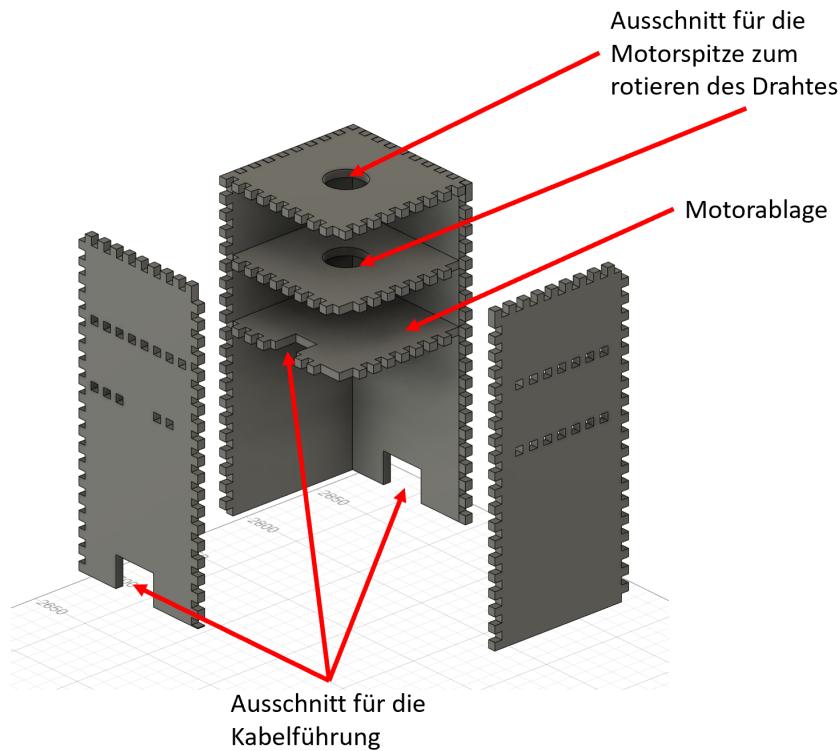


Abbildung 2.8: Einblick in den mittleren Turm

Das gesamte Design diente als Vorlage für die Laserauschnitte, die später aus den Holzplatten gefertigt wurden. Diese Vorbereitung ist entscheidend gewesen, damit alle Teile Präzise zusammenpassen.

Die Modularität und die leichte Zugänglichkeit der Komponenten sorgt für eine einfache Wartung bzw. Upgrades. Das hat den Vorteil, dass eine schnelle Anpassung z.B. an neue Spielvarianten möglich wäre. Das Endresultat ist ein durchdachtes und funktionales 3D-Design und sieht wie folgt aus:

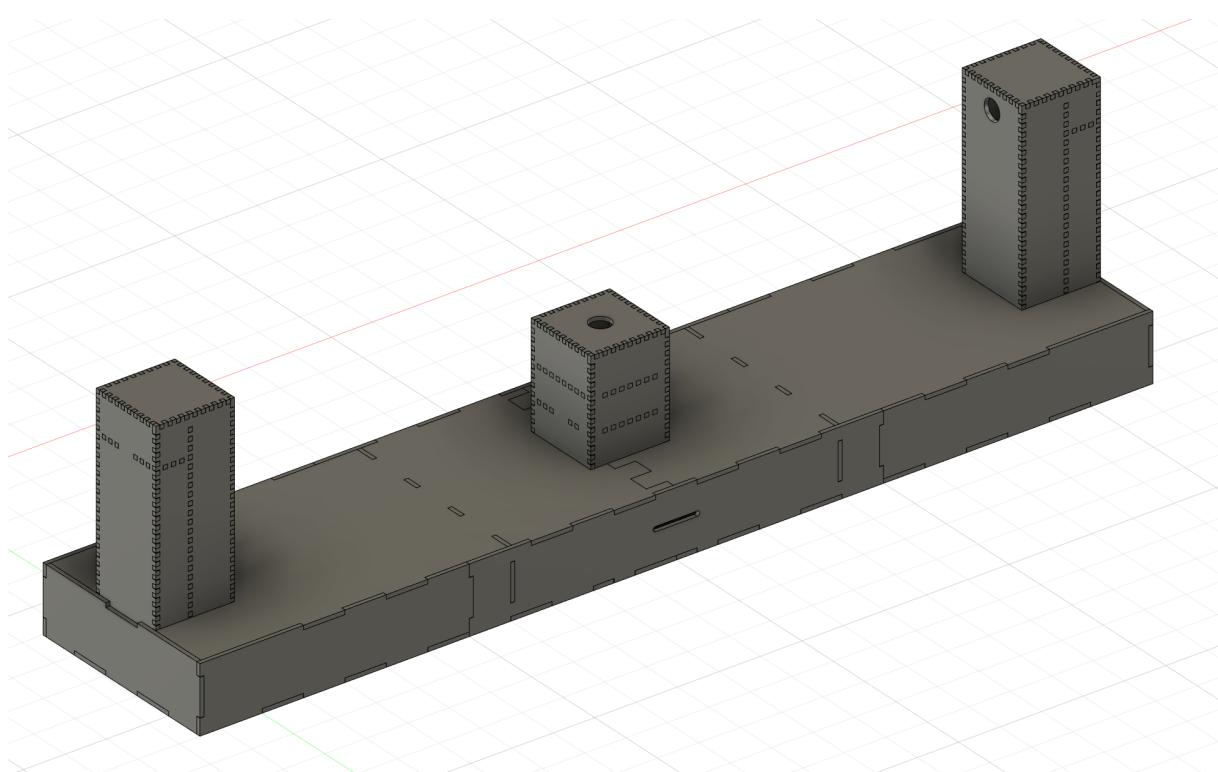


Abbildung 2.9: Das gesamte Design für Wire & Warriors

Kapitel 3

Hardware Umsetzung

Für die Umsetzung von Wire & Warriors wurden verschiedene Hardwarekomponenten ausgewählt und integriert. Es sind folgende Komponenten genutzt für die Realisierung:

Stepper-Motoren: Diese Komponenten wurden für die präzisen Bewegungen der Drähte genutzt. Es sind drei Motoren[Tro, 2023] zum Einsatz gekommen: linker Turm, rechter Turm und mittlerer Turm. Dadurch ist es möglich gewesen, die Drähte rotieren zu lassen und die Geschwindigkeit anzupassen.

Messingdrähte: Die Drähte[Hagebau, 2023] sind ein Meter lang gewesen und hatten einen Durchmesser von 6 Millimetern. Die Messingdrähte wurden gewählt, da sie eine gute Balance zwischen Flexibilität und Festigkeit bieten. Zusätzlich besitzt Messing eine gute elektrische Leitfähigkeit, dass ist wichtig gewesen um Berührungen von Spielern zu detektieren.

Stepper-Treiber TB6600: Um die Motoren zu steuern, wurden drei TB6600 Stepper-Treiber[DFROBOT, 2023] eingesetzt. Die Treiber ermöglichen die präzise Steuerung der Motoren. Dadurch ist es möglich gewesen, die Motoren gleichmäßig und kontrolliert zu bewegen. Ein enormer Vorteil dieser Treiber ist gewesen, dass sie einfach konfigurierbar durch seitliche Schalter sind. Somit ist es möglich gewesen, die Treiber einfach und schnell auf die Motoren einzustellen.

Soundmodul: Ein Soundmodul ist hinzugefügt worden, um akustisch Feedback für die Spieler zu geben. Dies sollte zur Spielatmosphäre beitragen und die Benutzer-freundlichkeit erhöhen.

Netzteil: Es wurde ein Netzteil verbaut, um die Mobilität des Spiels zu gewährleisten. Dadurch kann das Spiel jederzeit flexible an verschiedenen Orten genutzt werden.

LEDs: Es wurden acht LEDs integriert, um visuelles Feedback zu geben. Damit konnten wichtige Spielinformationen wie Spielstatus und die verbleibende Herzen den Spielern angezeigt werden.

Arduino Uno-Einsatz: Dieser Mikrocontroller diente als zentrale Steuereinheit des Spiels. Der Arduino Uno ermöglicht die Steuerung der Stepper-Motoren, das Detektie-

ren von Berührungen und die Ansteuerung der visuellen und akustischen Rückmeldungen durch LEDs und das Soundmodul.

3.1 Prototypen

Mit diesen Komponenten ist es möglich gewesen, das Spiel Wire & Warrios zu realisieren. Für die ersten Annäherung zur Realisierung des Spieles wurde Prototypen entwickelt. Der erste Prototype sah wie folgt aus:

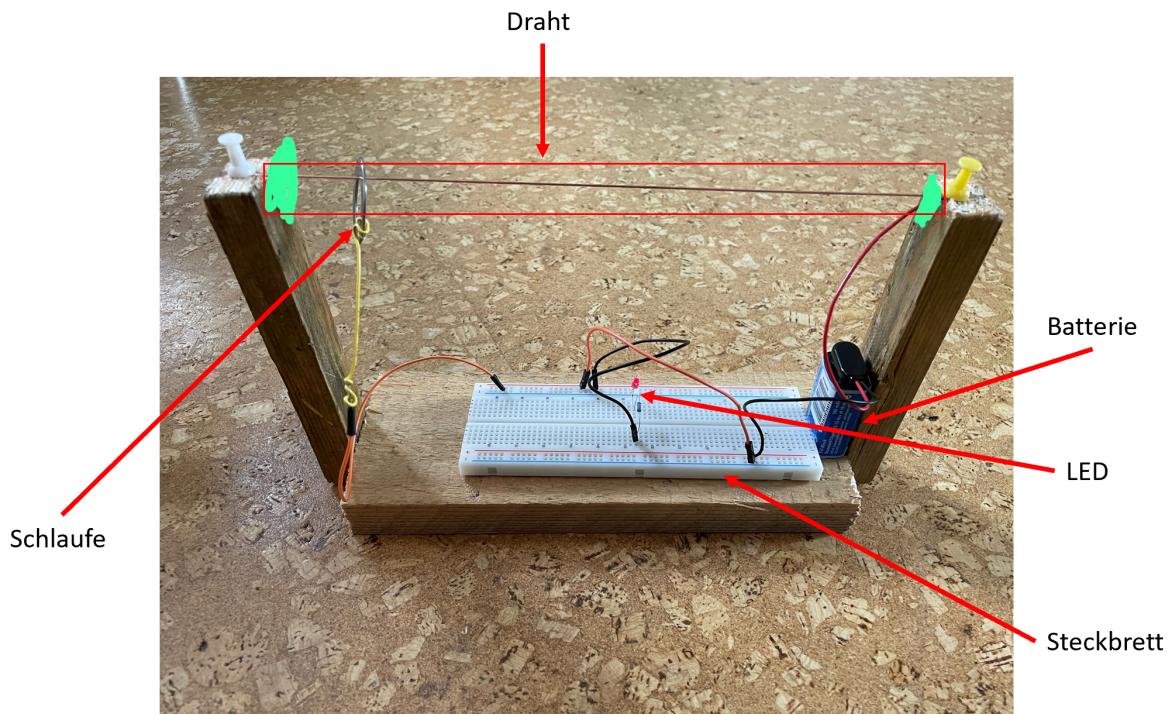


Abbildung 3.1: Die Umsetzung des Prototypens zum detektieren von Berührungen

Der erste und simpelste Prototype, wie in der Abbildung 3.1 zu sehen, ermöglichte die Detektierung von Berührungen. Die Berührungen wurden durch die rote LED signalisiert. Der nächste Schritt ist gewesen einen Motor einzubauen, um den Draht zu bewegen. Dieser Prototyp sah wie folg aus:

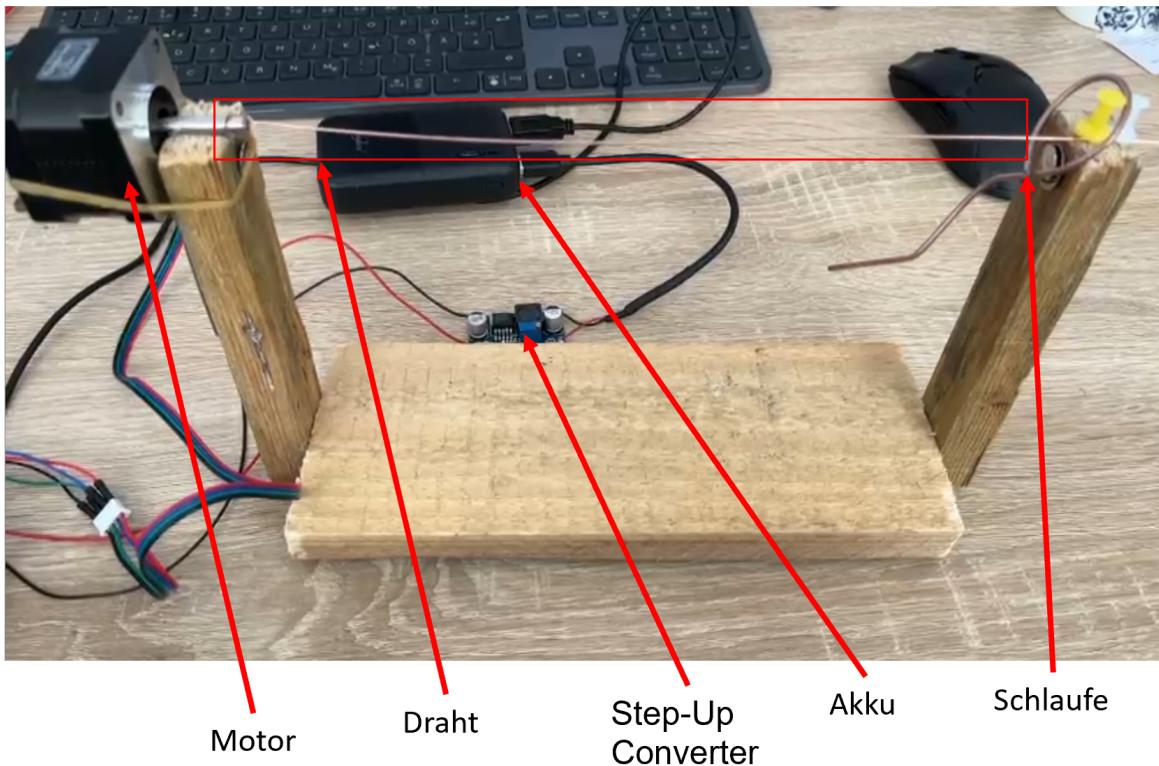


Abbildung 3.2: Die Umsetzung des Prototypens

Mit dem zweiten Prototype, in der Abbildung 3.2 zu sehen, ist es möglich gewesen, den Draht zu drehen und Berührungen zu detektieren. Der Step-Up Converter wurde genutzt, um die notwendige Spannung für den Stepper-Motor zu erreichen. Dieser konnte nicht allein durch die Ausgangsspannung vom Akku erreicht werden. Der Step-Up Converter wurde im Endprodukt durch das Netzteil ersetzt.

Zusätzlich musste der Stepper Motor Driver integriert werden, um den Motor zu steuern. Das Schaltbild für die Integrierung sah wie folgt aus:

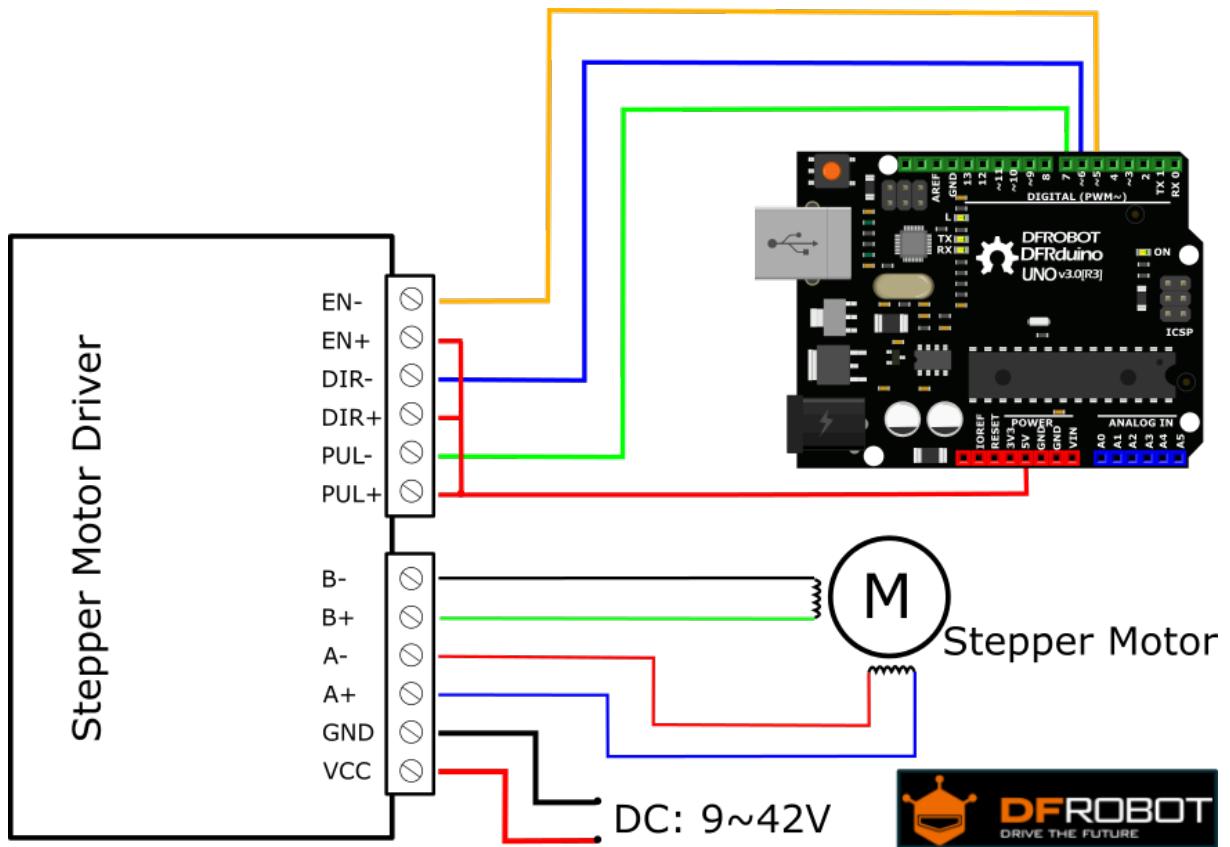


Abbildung 3.3: Schaltplan des Stepper Motor Drivers

Quelle: TB6600 Stepper Motor Driver

Der Schaltplan, wie in der Abbildung 3.3 zu sehen, gibt die Verkabelung des Steppers vor. Bevor der Stepper Motor Driver genutzt werden konnte, musste die Stromstärke Konfiguration angepasst werden. Die Stepper Motoren haben eine Stromstärke von 1A benötigt und deshalb mussten die Schalter vier und sechs umgelegt werden. Die Konfigurationen stehen auf dem Stepper Motor Driver und sahen wie folgt aus:



Abbildung 3.4: Konfigurationen auf dem Driver

Quelle: TB6600 Stepper Motor Driver

Die Ausgänge vom Stepper Driver Motor ‘EN’ und ‘DIR’ wurden nicht genutzt, da die Drähte sich nur eine Richtung drehen sollen. Die Drehung wurde durch Ausgang ‘PUL’ ermöglicht. Die Kabel aus ‘PUL’ wurden am Arduino Uno angebracht.

3.2 Realisierung des Projekts

Das gesamte Gerüst hat eine Länge von einem Meter und eine Breite von 30 Zentimeter. Diese Größe ist kompakt und bietet eine ausreichend große Spielfläche. Das zusammengebaute Gerüst sieht wie folgt aus:

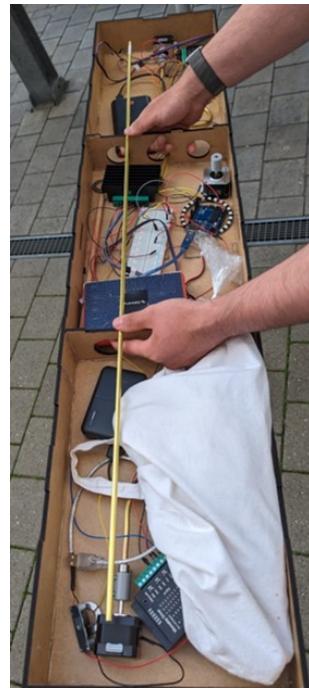


Abbildung 3.5: Gerüst ohne Deckel

Nachdem das Gerüst stand wurden die Türme fertiggestellt und sahen wie folgt aus:

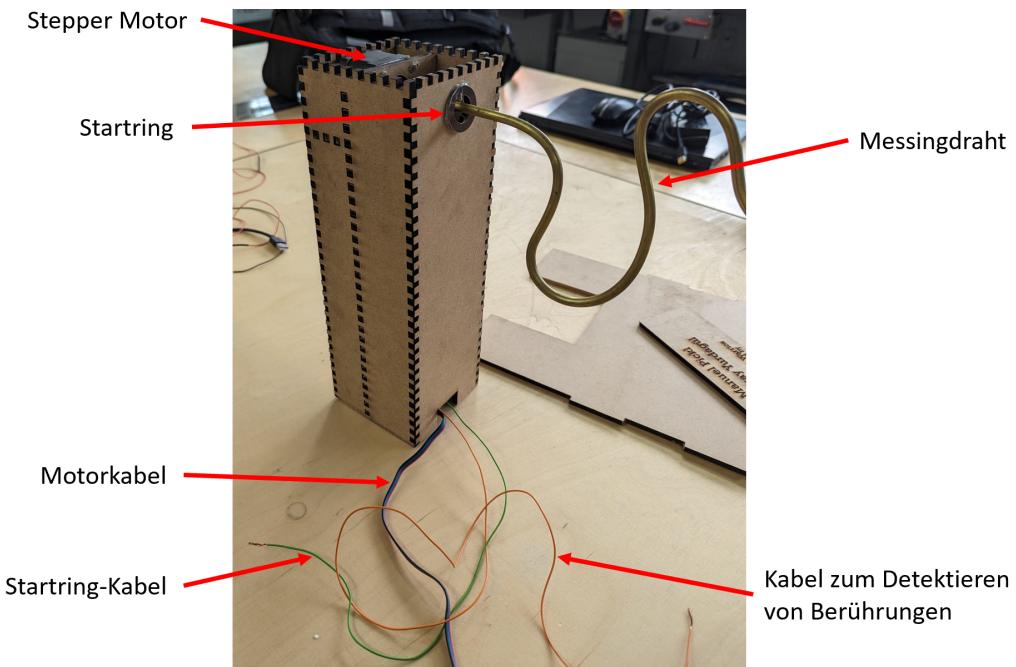


Abbildung 3.6: Veranschaulichung des fertiggestellten Turms

Die Abbildung 3.6 veranschaulicht einen der fertiggestellten Türme. Jeder Turm hat ein Kabel, welches verbunden ist mit dem Draht und eine Ausschnitt am Fuß des Turms für die Kabelführung. Durch die Verbindung mit dem Draht ist es möglich

Berührungen zu detektieren. Zusätzlich besitzen die äußeren Türme ein grünes Kabel, welches verbunden ist mit dem Ring am Turm. Damit kann der Spieler signalisieren, dass er sich in der Startposition befindet bzw. das Spiel beenden möchte.

Für die Herz-Anzeige wurden sechs LEDs an einem Gerüst in Reihe montiert und fixiert. Diese Gerüst wurde so konstruiert, sodass es die LEDs direkt unter dem Deckel angebracht werden. Damit ist es möglich gewesen, die herzförmigen Ausschnitt im Deckel des Gerüst optimal zu beleuchten. Dadurch konnten die Spieler genau die verbleibenden Leben sehen und visuelles Feedback erhalten. Das Gerüst sieht wie folgt aus:

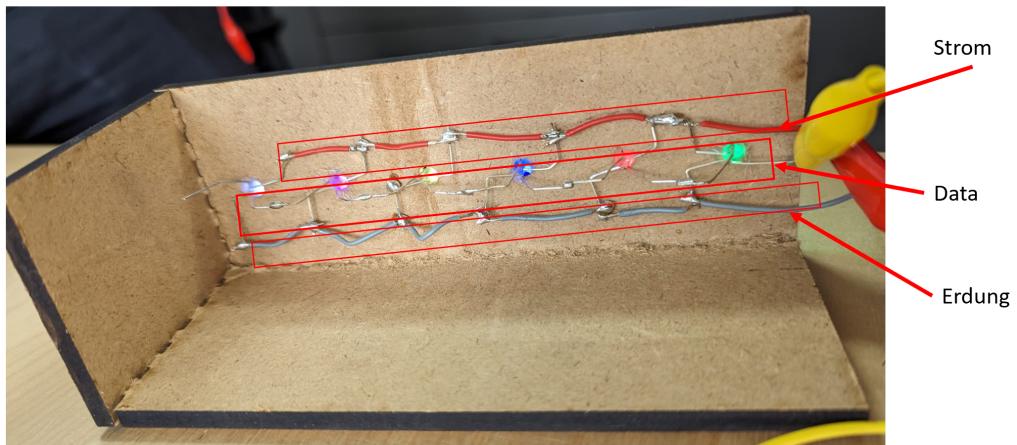


Abbildung 3.7: Zusammenlöten der LEDs

Mit dem erfolgreichen Aufbau der Prototypen und der Implementierung des finalen Gerüsts ist eine solide Basis für das spannende Spielerlebnis von Wire & Warriors geschaffen worden.

Kapitel 4

Software Umsetzung

Das Projekt stellte für uns nicht nur in Bezug auf die Hardware hohe Anforderungen, sondern erforderte auch für ein Mikrocontroller-Projekt eine ungewöhnlich komplexe Softwarelösung. Von Beginn an war es daher essenziell, eine klare und gut organisierte Struktur zu etablieren. Diese Vorgehensweise gewährleistet, dass der Code sowohl wartbar als auch übersichtlich bleibt, was für die effiziente Entwicklung des Projekts von großer Bedeutung ist.

Des Weiteren wurde von Beginn an besonderer Wert darauf gelegt, den Code unter Versionskontrolle zu stellen, wofür Git gewählt wurde. Dieser Schritt gewährleistete eine effiziente Verwaltung der Code-Änderungen und bot eine robuste Plattform für die Zusammenarbeit im Team an.

4.1 Entwicklungsumgebung

Für die Entwicklung der Software wurde Visual Studio Code (VSCode) als integrierte Entwicklungsumgebung (IDE) ausgewählt. VSCode ist aufgrund seiner vielseitigen Fähigkeiten hervorragend für die Programmierung von Mikrocontrollern geeignet. Diese Plattform zeichnet sich durch eine Vielzahl an Erweiterungen (Plugins) aus, die speziell für verschiedene Programmiersprachen und Projektspezifikationen konzipiert sind. Diese Flexibilität und Anpassungsfähigkeit von VSCode macht es zu einem idealen Werkzeug auch für dieses Entwicklungsprojekt.

4.2 Programmiersprache

Aufgrund der Nutzung des Arduino Uno als Mikrocontroller fiel die Wahl für die Entwicklungssprache auf C++. Arduino zeichnet sich durch seine umfangreichen Bibliotheken aus, die die Integration von Funktionen und Komponenten erheblich erleichtern. Diese Bibliotheken bieten eine reiche Auswahl an vorgefertigten Codes und Modulen, die es ermöglichen, komplexe Funktionen mit geringerem Aufwand zu implementieren und somit die Entwicklungszeit zu verkürzen.

4.3 PlatformIO

Wie bereits erwähnt, zeichnet sich Visual Studio Code durch seine umfangreiche Plugin-Unterstützung aus. Für dieses Projekt wurde speziell das Plugin PlatformIO verwendet, das sich als äußerst nützlich für die Arbeit mit Mikrocontrollern erwiesen hat. PlatformIO bietet umfassende Unterstützung bei der Handhabung des Arduino Uno. Es ermöglicht nicht nur das Kompilieren des Codes, sondern auch das bequeme Übertragen auf den Mikrocontroller. Zudem bietet es die Möglichkeit, verschiedene Ausgaben über den Serial Monitor zu überwachen.

Ein zentrales Element von PlatformIO ist die 'platformio.ini'-Datei. In dieser Konfigurationsdatei werden wichtige Projektinformationen festgelegt, wie zum Beispiel der verwendete Mikrocontroller-Typ (Arduino Uno), das Framework (Arduino) und die eingebundenen Bibliotheken.



Abbildung 4.1: PlatformIO Unterstützung in VSCode

4.4 Komponenten

Bei der Entwicklung wurde ein besonderes Augenmerk einen fein gegliederten und modularen Codeaufbau gelegt. Um diesen modularen Ansatz zu unterstützen, wurde für jede Komponente (repräsentiert durch eine .cpp-Datei) eine zugehörige Header-Datei (.h) erstellt. In diesen Header-Dateien sind alle nach außen sichtbaren Methoden und Variablen der jeweiligen Komponente definiert. Dies fördert eine klare Trennung zwischen der Implementierung und der Schnittstellendefinition, was die Lesbarkeit und Wartbarkeit des Codes wesentlich verbessert.

Die Applikation wurde in folgende Hauptkomponenten unterteilt:

4.4.1 main

Die main.cpp dient als zentrale Datei der Applikation. Ihre Hauptfunktion besteht darin, verschiedene Services und Module zu initialisieren. Zusätzlich beinhaltet die main.cpp spezifische Funktionen für Arduino, wie:

```
void setup() { ... } // run once
void loop() { ... } // run every tick
```

Quellcode 4.1: Arduino spezifische Funktionen

4.4.2 game

Die Game-Komponente stellt den umfassendsten Teil der gesamten Applikation dar. Sie beinhaltet die vollständige Logik, die den Zustand (State) und den Ablauf des Spiels bestimmt.

In jedem Zyklus (Tick) des Spiels wird die dazugehörige Funktion der Komponente aufgerufen. Diese Funktion übernimmt essentielle Aufgaben wie die korrekte Steuerung der LED-Anzeigen oder das Auslösen von Bewegungen der Motoren. Zusätzlich dazu beinhaltet die Game-Komponente wichtige Funktionen, die das Level-System des Spiels betreffen, wie beispielsweise die Fortschrittsverwaltung oder die Schwierigkeitssteigerung.

Hier ein vereinfachter Überblick über die Hauptfunktion dieser Komponente:

```
void tick() {
    showHeartLights(); // show the heart lights

    if (checkWireTouch()) { ... } // check for wire touch

    if (gameStarted) {
        turnBridge(); // turn bridge motor
    }
}
```

Quellcode 4.2: Tick Funktion der game Komponente

4.4.3 gameConstants

Diese Komponente beinhaltet eine Sammlung von Konstanten, die sowohl für das Design des Spiels als auch für dessen Dynamik von entscheidender Bedeutung sind. Diese Konstanten umfassen sowohl feste Parameter, die das grundlegende Erscheinungsbild und Verhalten des Spiels definieren, als auch variierbare Elemente, die es ermöglichen, den Spielfluss sowie visuelle und auditive Effekte anzupassen. Die Flexibilität dieser Konstanten ermöglicht es den Entwicklern, mit verschiedenen Spielaspekten zu experimentieren und das Spielerlebnis zu optimieren.

Nachfolgend finden Sie einen nicht vollständigen Auszug aus dieser Sektion:

```
// fixed values
const int lifeLEDsCount = 6;
const int pulsesForFullRotation = 6400; // motor specific

// variable values
const int errorCooldown = 1000; // milliseconds
const int volume = 25; // 0-30
```

Quellcode 4.3: Spiel Konstanten

4.4.4 lights

Die lights-Komponente ist für die Kontrolle sämtlicher Beleuchtungselemente im Spiel zuständig. Ein wesentlicher Bestandteil dieser Komponente sind die Start- und End-LEDs, welche als feste grüne LEDs an den Türmen integriert sind.

Das zweite zentrale Element der lights-Komponente sind die LEDs für die Darstellung der Herzen. Diese sind als smarte LEDs konzipiert, was bedeutet, dass sie in der Lage sind, verschiedene Farben anzunehmen. Die Farbänderungen erfolgen entsprechend dem aktuellen Zustand (State) im Spielverlauf.

Für die Ansteuerung der Herzen-LEDs wird die Adafruit_NeoPixel-Bibliothek verwendet. Diese Bibliothek bietet eine umfangreiche Palette an Funktionen für die Kontrolle von RGB-LEDs.

Einige der Funktionen der lights-Komponente umfassen:

```
void showGameNotStarted() { ... } // show default lights
void showHearts() { ... } // display red hearts
void showError() { ... } // show error on wire touch
void showRingLights() { ... } // show light on ring touch
```

Quellcode 4.4: Wichtigste Beleuchtungsfunktionen

4.4.5 logging

Die Logging-Komponente ist ausschließlich für Debugging-Zwecke konzipiert. Ihr Hauptmerkmal ist eine globale Log-Funktion, die für die Protokollierung von Informationen während der Entwicklungs- und Testphasen des Spiels verwendet wird. Diese Funktion nutzt den Standardmechanismus des Serial Monitors, um Log-Nachrichten auszugeben, mit dem einzigen Unterschied, dass eine Überprüfung stattfindet, ob die Logging-Funktion aktiviert ist, bevor irgendwelche Nachrichten protokolliert werden.

4.4.6 motor

Die motor-Komponente ist zuständig für die Steuerung aller Motoren im Spiel. Sie umfasst sowohl die Kontrolle des Motors für das Brückenelement als auch die Ansteuerung der Motoren, welche die Drähte an den Türmen bewegen.

Außerdem ein wichtiger Bestandteil dieser Komponente sind die Überprüfungsfunktionen, die kontrollieren, ob und wann eine Drehung der Motoren durchgeführt werden soll. Details dazu werden später genauer erläutert.

Als Beispiel für die Funktionalität dieser Komponente dient die Funktion zum Drehen der Turmrähte:

```
void turnSides(int level) {
    // check if tower motors should turn
    if (!timeToTurnSides()) {
        return;
    }

    for (int i = 0; i < getMotorRotations(level); i++) {
        // turn side motors one step
        digitalWrite(stepPinWire, HIGH);
        delayMicroseconds(1); // we need the delay
        digitalWrite(stepPinWire, LOW);
    }
}
```

Quellcode 4.5: Code zum Drehen der Turmrähte

4.4.7 pins

Die pins-Datei spielt eine wesentliche Rolle in der Organisation und Konfiguration des Projekts, indem sie eine klare und strukturierte Übersicht über die Belegung aller Pins bietet. In dieser Datei werden alle Pins, die für die Verbindung des Arduino-Boards mit den verschiedenen Komponenten des Systems verwendet werden, über Konstanten definiert.

Diese systematische Zuordnung ist besonders hilfreich während der Verkabelungsphase, da sie eine genaue und fehlerfreie Verbindung der Hardware-Komponenten mit den entsprechenden Pins des Arduino-Boards gewährleistet. Änderungen in der Hardware-Konfiguration können somit leichter umgesetzt werden, indem die entsprechenden Werte in der pins-Datei angepasst werden, ohne den Hauptcode des Projekts zu beeinflussen.

4.4.8 touch

Diese Komponente ist eine unterstützende Funktionseinheit, die speziell für die Erkennung von Berührungen an bestimmten Pins konzipiert wurde. Diese Funktion ist ein kritischer Bestandteil des Spiels und wird in verschiedenen Szenarien eingesetzt, wie beispielsweise beim Start des Spiels, beim Erreichen des Ziels oder bei der Erkennung eines Fehlers, also einer Berührung des Drahtes.

Die Kernfunktion der touch-Komponente nutzt die digitalRead()-Funktion, die von der Arduino-Bibliothek bereitgestellt wird.

```
bool playerTouchesPin(int pin) {
    // use function from arduino framework
    return !digitalRead(pin);
}
```

Quellcode 4.6: Detektion von Benutzerberührung

4.5 Asynchronität der Software

Die Herausforderung, mehrere Funktionen gleichzeitig auszuführen, war ein komplexes Problem im Rahmen dieses Projekts. Insbesondere mussten die Motoren kontinuierlich laufen, während das Spiel fortschritt, Berechnungen durchgeführt und ständig auf Berührungen geprüft wurden. Dies stellte sich als besonders schwierig heraus, da außerdem im Spiel zahlreiche zeitbasierte Funktionen existieren, wie etwa Fehler-Cooldowns mit Animationen sowie Start- und Endanimationen für jedes Level.

Anfangs wurde versucht, das Problem mit Verzögerungen (Delays) im Code zu lösen. Diese Methode führte jedoch zu nicht nachhaltigen Lösungen und verursachte während der Entwicklung immer wieder Bugs. Die endgültige Lösung des Problems fand sich in der Verwendung von Zeitstempeln. An verschiedenen Stellen in der Anwendung wird ein Zeitstempel gesetzt und anschließend überprüft, ob eine bestimmte Zeitspanne verstrichen ist.

Diese Methode ermöglicht es, eine Art von Asynchronität zu simulieren. Durch das Arbeiten mit Zeitstempeln statt festen Verzögerungen kann das Programm fortlaufend andere wichtige Aufgaben ausführen, während es gleichzeitig auf das Erreichen bestimmter Zeitpunkte wartet. Dieser Ansatz verbessert nicht nur die Ansprechbarkeit (Reaktion) des Programms, sondern erhöht auch die Stabilität und Zuverlässigkeit des Codes, indem er die typischen Probleme vermeidet, die mit der Verwendung von Delays einhergehen.

4.5.1 Beispiel

Dieses Vorgehen lässt sich gut am Beispiel der Funktionalität zum Drehen des Brückenzimmers illustrieren. Die turnBridge()-Funktion beginnt mit einer Überprüfung, ob eine

Drehung der Brücke überhaupt möglich ist. Diese Überprüfung wird durch die Funktion timeToTurnBridge() realisiert, die auf Zeitsteuerung basiert. Es wird kontrolliert, ob die festgelegte Zeitspanne seit dem letzten Aktivieren der Brückendrehung verstrichen ist. Sobald festgestellt wird, dass die definierte Zeitspanne abgelaufen ist, wird der Brückenmotor aktiviert und der Zeitstempel aktualisiert, um den nächsten Zeitpunkt für eine mögliche Aktivierung zu markieren.

```
void turnBridge() {
    // leave function of bridge should not be turned
    if (!timeToTurnBridge()) {
        return;
    }

    ...
}

bool timeToTurnBridge() {
    // elapsed time is bigger than motor waiting time
    bool timeToTurn = elapsedTime >= motorDelay;

    if (timeToTurn) {
        // update last turn time
        lastBridgeTurnTime = currentTime;
    }

    return timeToTurn;
}
```

Quellcode 4.7: Zeitgesteuertes Drehen des Brückenmotors

Kapitel 5

Ausblick und Fazit

5.1 Ausblick

Das aktuelle Design des Spiels ist im Großen und Ganzen gut durchdacht und funktioniert einwandfrei. Jedoch haben wir eine Schwachstelle identifiziert, und zwar im Bereich des Brückensegments. Dieses Element ist essentiell für das Spiel, da es den Spielern ermöglicht, einander zu passieren und zur gegenüberliegenden Seite zu gelangen. Gleichzeitig stellt es jedoch einen kritischen Punkt dar, an dem die Fairness der Spieler gefordert ist. Theoretisch könnten Spieler diese Stelle nutzen, um aus dem vorgegebenen System auszubrechen und das Spiel auf eine unfaire Weise zu beenden.

Wir haben verschiedene Entwürfe und Konzepte für einen Tunnel erarbeitet, der das Überqueren der Brücke erlaubt, ohne die Möglichkeit zu bieten, das System zu umgehen. Leider konnte keine dieser Ideen das Problem zufriedenstellend lösen. Letztendlich haben wir uns entschieden, die Situation so zu belassen, wie sie ist, und setzen auf das faire Verhalten der Spieler.

Diese Entscheidung spiegelt einen Kompromiss wider, der oft in der Spielentwicklung gemacht werden muss, zwischen idealer Designlösung und praktischer Umsetzbarkeit. Wenn in Zukunft an dem Spiel weitergearbeitet wird, bietet die Lösung dieses Problems eine sinnvolle Zielsetzung.

Eine Überarbeitung des Designs könnte außerdem eine wertvolle Verbesserung für das Spiel darstellen. Die derzeitige Konstruktion aus dünnen Holzplatten, welche zusammengesteckt und teilweise mit Heißkleber und Flachwinkeln verstärkt wurden, bietet zwar die grundlegende Funktionalität, jedoch gibt es sowohl in Bezug auf Stabilität als auch auf Ästhetik Raum für Verbesserungen.

Eine einfache und kosteneffiziente Möglichkeit wäre das Färben oder Lackieren der Holzplatten. Dies würde nicht nur das ästhetische Erscheinungsbild verbessern, sondern könnte auch die Haltbarkeit des Holzes erhöhen. Verschiedene Farben oder Lackierungen könnten auch dazu beitragen, verschiedene Spielbereiche visuell zu unterscheiden. Eine umfassendere Überarbeitung könnte den Ersatz der Holzplatten durch stabilere und langlebigere Materialien beinhalten.

5.2 Fazit

Es war ein anspruchsvolles, aber ungemein wertvolles Projekt!

Die Projekte sind eigentlich für drei Personen ausgelegt, und leider haben wir schon früh im Projektverlauf einen Teamkollegen verloren, sodass wir nur noch zu zweit waren. Als Medieninformatiker war der Bereich der Hardware für uns ein völlig neues Terrain. Wir hatten kaum Erfahrung und verfügten über wenig Vorwissen, insbesondere in grundlegenden Aspekten der Elektrotechnik. Zwar hatten wir im Studium Module wie 'Grundlagen digitaler Systeme' oder 'Mobile and Ubiquitous Computing', doch diese sind schon einige Zeit her, und die relevanten Bereiche für dieses Projekt wurden nur teilweise abgedeckt. Da der Rest des Studiums wenig Berührungspunkte mit diesen Themen aufwies, gingen viele Kenntnisse leider wieder verloren. Daher standen wir vor der Herausforderung, uns das nötige Wissen von Grund auf anzueignen.

Besonders herausfordernd war für uns das Verständnis grundlegender Konzepte der Elektronik, insbesondere von Strom, Spannung und Widerstand. Diese Aspekte waren jedoch essentiell für das Funktionieren unseres Spiels. Glücklicherweise erhielten wir vom Team des MakerSpace wertvolle Unterstützung und Nachhilfe in diesen Bereichen, was uns enorm weiterhalf.

Ein weiteres signifikantes Problem stellte die Inbetriebnahme und Steuerung der Motoren dar, insbesondere im Zusammenhang mit den Schrittmotoren (Stepper-Motoren). Ursprünglich planten wir, kleinere Steuerungsboards einzusetzen, aber wir mussten feststellen, dass unsere Kenntnisse dafür nicht ausreichten. Schließlich entschieden wir uns für den Einsatz größerer Boards, da wir nur mit diesen eine funktionierende Lösung realisieren konnten.

Diese große Aufgabe war jedoch eine Erfahrung, die uns sehr wertvoll erscheint. Auf der Softwareseite traten glücklicherweise keine Probleme auf, was uns sehr zugute kam. Trotz der Herausforderungen und des reduzierten Teams gelang es uns, ein Projekt zu realisieren, das unsere anfänglichen Erwartungen nicht nur erfüllte, sondern in vielerlei Hinsicht sogar übertraf. Diese Erfahrung hat nicht nur unsere Fähigkeiten erweitert, sondern uns auch als Team zusammen geschweißt.

Literaturverzeichnis

[Tro, 2023] (2023). Tronxy 3D Printer SL42STH40-1684A 1.8A 78Oz-in 42 Stepper Motor. <https://tronxyglobal.com/products/tronxy-3d-printer-sl42sth40-1684a-1-8a-78oz-in-42-stepper-motor>.

[DFROBOT, 2023] DFROBOT (2023). TB6600_Stepper_Motor_Driver_SKU__DRI0043- DFRobot. https://wiki.dfrobot.com/TB6600_Stepper_Motor_Driver_SKU__DRI0043.

[Hagebau, 2023] Hagebau (2023). GAH ALBERTS Rundstange, Braun, Messing - hagebau.de. <https://www.hagebau.de/p/gah-alberts-rundstange-braun-messing-anP7000155175/>.

Abbildungsverzeichnis

2.1	Visualisierung des ersten Konzepts	4
2.2	Visualisierung des zweiten Konzepts	5
2.3	Visualisierung des dritten Konzepts	6
2.4	Visualisierung des vierten Konzepts	6
2.5	Visualisierung des fünfte Konzepts	7
2.6	Verzahnung am Deckel	8
2.7	Stabilisierung des Gerüsts durch Mittelstücke	8
2.8	Einblick in den mittleren Turm	9
2.9	Das gesamte Design für Wire & Warriors	10
3.1	Die Umsetzung des Prototypens zum detektieren von Berührungen	12
3.2	Die Umsetzung des Prototypens	13
3.3	Schaltplan des Stepper Motor Drivers	14
3.4	Konfigurationen auf dem Driver	15
3.5	Gerüst ohne Deckel	16
3.6	Veranschaulichung des fertiggestellt Turms	16
3.7	Zusammenlöten der LEDs	17
4.1	PlatformIO Unterstützung in VSCode	19

Quellcodeverzeichnis

4.1	Arduino spezifische Funktionen	19
4.2	Tick Funktion der game Komponente	20
4.3	Spiel Konstanten	21
4.4	Wichtigste Beleuchtungsfunktionen	21
4.5	Code zum Drehen der Turmdrähte	22
4.6	Detektion von Benutzerberührungen	23
4.7	Zeitgesteuertes Drehen des Brückenmotors	24