

Klassenthema Faszination

# Entwickeln eines motorisierten Kamera-Slider (OpenSlide)

Noé Krebs, Tim Reber





## Inhalt

<b>1. MINDMAP .....</b>	<b>5</b>
<b>2. EINLEITUNG .....</b>	<b>6</b>
2.1. AUSGANSLAGE .....	6
2.2. ABGRENZUNG .....	6
<b>3. ZIELFORMULIERUNGEN .....</b>	<b>7</b>
<b>4. GROBPLANUNG .....</b>	<b>8</b>
<b>5. KONZEPTIONIERUNG .....</b>	<b>12</b>
5.1. MATERIALLISTE .....	12
5.2. TECHNISCHE ASPEKTE .....	14
5.2.1. 3D MODELLIERUNG .....	14
5.2.2. 3D-DRUCK .....	18
5.2.3. EIGENES PROTOKOLL .....	19
<b>6. BENUTZEROBERFLÄCHE .....</b>	<b>20</b>
6.1. TECHNOLOGIE .....	20
6.1.1. NEXT.JS .....	20
6.1.2. TAILWINDCSS .....	20
6.1.3. VERCEL.COM .....	21
6.2. FEATURES .....	21
<b>7. ARDUINO .....</b>	<b>22</b>
7.1. KALIBRIERUNG .....	22
7.2. KOMMUNIKATION .....	22
<b>8. UMSETZUNG .....</b>	<b>23</b>
8.1. CONTROL CENTER .....	24
8.2. END TEIL MIT UMLENKROLLE .....	27
8.3. KAMERAWAGEN .....	28
8.3.1. ABDECKUNG .....	29
<b>9. PREISVERGLEICH .....</b>	<b>30</b>
9.1. PREISAUFSTELLUNG .....	30
9.2. FEATUREVERGLEICH .....	31
9.3. SCHLUSSFOLGERUNG .....	32

<b>10. STRESS-TEST .....</b>	<b>33</b>
10.1. GESCHWINDIGKEIT .....	33
10.2. KAMERAS.....	34
<b>11. VERÖFFENTLICHUNG .....</b>	<b>35</b>
11.1. GITHUB .....	35
11.2. ERREICHBARKEIT .....	36
<b>12. GLOSSAR.....</b>	<b>37</b>
<b>13. ABKÜRZUNGSVERZEICHNIS .....</b>	<b>38</b>
<b>14. SCHLUSSWORT.....</b>	<b>39</b>
14.1. ABWEICHUNG VON DEN ZIELEN .....	39
14.2. WEITERES VORGEHEN .....	40
14.3. BEARBEITUNGSGRAD .....	40
<b>15. ABBILDUNGSVERZEICHNIS .....</b>	<b>41</b>
<b>16. TABELLENVERZEICHNIS.....</b>	<b>42</b>
<b>17. SCHLUSSERKLÄRUNG .....</b>	<b>43</b>
<b>18. ARBEITSJOURNAL.....</b>	<b>44</b>
<b>19. ANHANG.....</b>	<b>48</b>

## 1. Mindmap<sup>1</sup>



<sup>1</sup> Mindmap, selbst erstellt

## 2. Einleitung<sup>2</sup>

Der motorisierte Kamera-Slider ist ein bedeutendes Werkzeug in der Videoproduktion, insbesondere für Filmemacher, die einzigartige Aufnahmen anstreben. Mit präzisen Schienen und einem leistungsstarken Motor ermöglicht dieser Slider exakte Kamerabewegungen bei anpassbarer Geschwindigkeit und Einstellungen. Diese Arbeit untersucht die technischen Aspekte des motorisierten Kamera-Sliders im Kontext der faszinierenden Welt der Videoproduktion und Technik, perfekt passend zum Klassenthema "Faszination".

Wir haben uns für dieses Projekt entschieden, da uns sowohl die Welt der Videographie als auch die Technik begeistert.

Das Projekt Open Source zu machen war für uns ein Anliegen, da wir in unserm Hobby als Entwickler viel Open-Source-Software verwenden und wir gerne auch mal etwas zu dieser Community beitragen möchten.

### 2.1. Ausgangslage

In unserem Projekt wollen wir einen motorisierten Kamera-Slider selbst bauen (OpenSlide). Dieser besteht aus vorherig bestellten Einzelkomponenten, welche wir dann nach eigens kreiertem Konzept zusammenbauen. Damit man den Kamera-Slider auch ansteuern und verwalten kann, gibt es ein Web-UI, wo man Distanz, Geschwindigkeit, vordefinierte Modi und weitere Parameter konfigurieren kann. Der Code für das Web-UI wird komplett selbst geschrieben. Das gilt nicht nur für das Web-UI selbst, sondern auch für die Logik, welche der Kamera-Slider bereitstellen muss, damit die Integration mit dem Web-UI funktioniert. Damit diese beiden Komponenten zusammen kommunizieren können, werden wir ein simples Protokoll schreiben. Am Ende des Projekts wird es möglich sein, den Kamera-Slider via USB-Kabel mit einem PC zu verbinden und via Web-UI die Aktionen, welche der Kamera-Slider ausführen soll, zu bestimmen. Es wird möglich sein, eine Kamera auf dem Kamera-Slider zu befestigen. Dies via standardisiertem Schraubsystem, welches ermöglichen soll, dass möglichst viele Kameras mit unserem Slider kompatibel sind.

### 2.2. Abgrenzung

In unserem Projekt beschränken wir uns darauf, dass die Kamera nicht in verschiedene Richtungen verstellbar, sondern fix auf dem Kamera-Slider angebracht ist. Dies ist bei anderen Lösungen möglich, wir werden dies aber nicht realisieren, weil es den Zeitrahmen dieses Projekts sprengen würde.

Das Web-UI, um den Slider zu verwalten, wird öffentlich für alle erreichbar sein. Es wird keine Prüfung geben, ob man im Besitz einer unserer Kamera-Slider ist.

Das Web-UI muss nur auf offiziell unterstützten Chrome basierten Browsern funktionieren, alles andere wird von unserer Seite her nicht gewährleistet.

---

<sup>2</sup> Einleitung, Selbst erstellt

### 3. Zielformulierungen<sup>3</sup>

- Unser Ziel ist es einen motorisierten Kamera-Slider zu entwickeln. Dabei setzen wir auf herkömmliche, erschwingliche und leicht zugängliche Komponenten. Unsere Fortschritte dokumentieren wir sorgfältig mit Bild und Text. Anschliessend veröffentlichen wir die Ergebnisse (Pläne / Code / Bilder) in einem GitHub Repository.
- Wir wollen einen umfassenden Kostenvergleich für unseren Slider im Vergleich zu ähnlichen kommerziellen Produkten erstellen. Hierzu werden wir Recherchen zu den Preisen herkömmlicher Komponenten durchführen. Diese vergleichen wir mit verfügbaren kommerziellen Kamera-Slider-Produkten im Internet.

Die Ergebnisse dieses Vergleichs werden wir in einer klaren und visuell ansprechenden Form präsentieren. Am Ende dieses Vergleichs werden wir eine Schlussfolgerung ziehen, die die Unterschiede in den Kosten zwischen unserem Slider und den kommerziellen Alternativen zusammenfasst.

- Um die technischen Grenzen unseres Sliders zu erkunden, führen wir eine Vielzahl von Tests durch. An verschiedenen Standorten nutzen wir professionelle Kameras in Kombination mit unserem Slider, um Aufnahmen anzufertigen.

Die Ergebnisse dieser Tests werden in einem eigens erstellten Bewertungsraster mit fertigen Kamera-Slider aus dem Internet verglichen und präsentiert. Hierbei werden die einzelnen Punkte für die ausgewählten Produkte auf einer Skala bewertet. Diese umfassende Bewertung erlaubt uns, die technischen Möglichkeiten unseres Sliders gründlich zu untersuchen.

---

<sup>3</sup> Zielformulierung, Selbst erstellt

## 4. Grobplanung<sup>4</sup>

Woche	Arbeitsschritte	Verantwortlich*	Bemerkungen
KW 43	Start mit der VA <ul style="list-style-type: none"> <li>- Überprüfen und finalisieren der Materialliste</li> <li>- Bestellen der Teile</li> <li>- Serial Protokoll ausbauen</li> <li>- Arbeitsjournal nachführen</li> </ul>	Noé, <b>Tim</b>	Serielles Protokoll wird in NodeJS und in C++ implementiert.  Alle benötigten Komponenten werden online, entsprechend unserer Materialliste bestellt.
KW 44	<ul style="list-style-type: none"> <li>- Initialisierung des Web-UI Projekt</li> <li>- Prototyp des Web-UI deployen (im Internet verfügbar machen)</li> <li>- Definieren und dokumentieren der Anforderungen an das Web-UI</li> </ul>	Noé, Tim	Erstellen eines Next.js Projekts für das Web UI.  Erste Features werden bereits gemäss Anforderungen eingebaut.
KW 45	<ul style="list-style-type: none"> <li>- Entwurf des 3D Slider-Modell</li> <li>- Modell für 3D Druck bereit machen (Slicen).</li> <li>- Modell drucken</li> <li>- Beginn Konstruktion Openslide</li> </ul>	Noé, <b>Tim</b>	Eine erste Version des Modells wird konstruiert und gedruckt. Es werden weitere Iterationen folgen, nachdem wir mit diesem Modell Tests durchgeführt haben.  3D Modelle werden wir in Fusion 360 konstruieren.  Wir beginnen den Bau des Sliders mit den bestellten Komponenten. Die Schaltkreise werden gelötet.

<sup>4</sup> Grobplanung, Selbst erstellt



KW 46	<ul style="list-style-type: none"> <li>- Weitere Iterationen des 3D Modells konstruieren</li> <li>- Rendern des 3D Modells für das Web-UI</li> <li>- Einbinden des 3D Modells ins Web-UI</li> </ul>	<b>Noé, Tim</b>	In Blender werden wir ein Produktbild rendern.
KW 47	<ul style="list-style-type: none"> <li>- Entwicklung der Logik auf Seite des OpenSlides</li> </ul>	<b>Noé, Tim</b>	Bis zu diesem Zeitpunkt haben wir die Logik für den User aus dem Web-UI entwickelt. Damit diese Aktionen umgesetzt werden können, müssen wir noch den Code für den Slider in C++ schreiben.
KW 48	<ul style="list-style-type: none"> <li>- Weiterführung der Slider Entwicklung</li> <li>- Abstimmung WebUI – Slider</li> </ul>	<b>Noé, Tim</b>	WebUI und Slider Firmware müssen aufeinander abgestimmt werden.
KW 49	<ul style="list-style-type: none"> <li>- Testen des Sliders zusammen mit dem Web-UI</li> <li>- Testen der Performance des Sliders</li> <li>- Beheben allfälliger Bugs</li> </ul>	<b>Noé, Tim</b>	<p>Wir müssen den Slider auf Fehler im Zusammenhang mit dem Web-UI testen.</p> <p>Wir wollen herausfinden, wo die Grenzen unseres Sliders liegen. Was ist die maximale Geschwindigkeit? Dies wäre eine berechtigte Frage, die man hier stellen könnte.</p> <p>Falls Fehler auftauchen müssen wir diese beheben.</p>

KW 50	<ul style="list-style-type: none"><li>- Finalisierung und Optimierungen des Sliders</li></ul>	Noé, <b>Tim</b>	<p>Hier werden letzte Optimierungen vorgenommen.</p> <p>Falls unerwartete Fehler in unserem Code oder in unserer Konstruktion auftreten, haben wir hier noch Zeit diese zu beheben.</p>
KW 51	<ul style="list-style-type: none"><li>- Abgabe der Arbeit</li><li>- Kreieren von Aufnahmen</li></ul>	<b>Noé</b> , Tim	<p>Abgabe in zwei unterschiedlichen Ausführungen: PDF und bearbeitetes Worddokument</p> <p>Falls nach der Abgabe noch Zeit übrig bleibt, wollen wir mit dem Slider eigene Aufnahmen machen.</p>

\*Hauptverantwortlicher: **Fett**. Pro Woche wird ein Hauptverantwortlicher festgelegt. Wir arbeiten jeweils an denselben Aufgaben.

# Hauptteil

In diesem Kapitel beschreiben wir, wie wir das Endprodukt gebaut haben und gehen auf die einzelnen Aspekte des Projekts ein.








## 5. Konzeptionierung <sup>5</sup>

In folgendem Kapitel haben wir die Planung und die Informationsbeschaffung beschrieben.

### 5.1. Materialliste <sup>6</sup>


In der folgenden Materialliste sind alle relevanten Teile zu finden, die wir für den Bau des Sliders eingeplant und bei Bedarf bestellt haben.

Tabelle 1 Material

Komponente	Beschreibung	Link
<b>Schienen</b>	Als Schienen verwenden wir standardisierte V-Slot Aluminium Extrusions. Diese sind einfach zu verarbeiten und leicht zu beschaffen.	
<b>Motor</b>	Damit das Cart auch automatisch auf den Schienen fahren kann, braucht es einen Motor. Wir haben uns für den Nema 17 42 Stepper Motor entschieden.	
<b>Keilriemen</b>	Der Keilriemen wird einmal um den Kamera-Slider gespannt, er ist dafür verantwortlich, dass das Cart auf den Schienen mit Hilfe des Motors überhaupt fahren kann. Da wir nicht auf eine Standardgrösse zurückgreifen konnten, haben wir uns dazu entschieden ein Set zu verwenden. Dies ermöglicht, den Keilriemen selbst zu spannen. Damit können wir ihn auf unsere bestimmte Grösse anpassen. Das hat einerseits den Vorteil, dass er von der Grösse her sicher passt. Andererseits können wir dadurch auch ein wenig experimentieren und optimieren, da wir die Zugfestigkeit selbst einstellen können.	
<b>Cart</b>	Das Cart wird durch den Keilriemen auf den Schienen hin und her gezogen, das heisst es ist das bewegende Element in unserer Konstruktion. Darauf wird dann schlussendlich die Kamera befestigt, damit diese in der Lage ist, nach Belieben hin und her zu fahren. Das Cart hat eine Standardgrösse, welche den Schienen entspricht. Es ist ausserdem verstellbar, je nach dem für welche Schienen man sich entschieden hat. Dadurch können wir uns sicher sein, dass es für unser Projekt funktioniert.	
<b>Arduino</b>	Damit wir die Logik auf Seite des Kamera-Sliders implementieren können verwenden wir einen Arduino Micro-Controller. Dieser hat den Vorteil, dass er sehr klein ist und alle benötigten Anschlüsse bietet.	
<b>Netzteil</b>	Um den Motor anzutreiben, brauchen wir mehr Strom als wir über die USB-Schnittstelle zur Verfügung stellen können. Deswegen brauchen wir ein externes Netzteil damit der Motor ordnungsgemäss funktioniert.	
<b>End-Schalter</b>	Damit wir im Code wissen können wo sich das Cart auf dem Slider befindet, müssen wir am Anfang eine Kalibrierung durchführen. Dafür benötigen wir zwei Schalter, welche jeweils am Anfang und am Ende des Sliders befestigt	

<sup>5</sup> Konzeptionierung, Selbst erstellt

<sup>6</sup> Materialliste, Stark bearbeitet

	sind und uns benachrichtigen, wenn es einen Kontakt mit dem Cart gibt.	
<b>Motor-Treiber</b>	Er wird für die Kommunikation mit dem Motor eingesetzt und befiehlt dem Motor: Geschwindigkeit, Stromverbrauch, Präzision und Temperatur.	
<b>Kondensator</b>	Kondensatoren speichern kurzfristig elektrische Ladungen und die damit verbundene elektrische Energie, dies sorgt dafür, dass konstant gleich viel Strom am Endgerät ankommt und somit Schwankungen in der Spannung eliminiert werden.	
<b>Zubehör</b>	Um diverse Komponenten zu befestigen, benötigen wir unter anderem diverse Schrauben, sowie Heissleim. Diese Utensilien hatten wir bereits zur Hand.	

## 5.2. Technische Aspekte

In diesem Kapitel gehen wir genauer auf die technische Planung ein.

### 5.2.1. 3D Modellierung

Damit wir unser Projekt richtig planen und uns für die richtigen Komponenten entscheiden konnten, mussten wir eine Idee manifestieren, wie genau der Kamera-Slider schlussendlich aussehen soll.

Zum Brainstorming haben wir von Hand einige Skizzen erstellt, bis wir uns schlussendlich auf diesen groben Entwurf geeinigt haben. Dieser eignet sich am Besten, da alles auf möglichst kleinem Raum untergebracht werden kann. Ob alles genau so passen würde, wie wir uns das vorgestellt haben, ist noch unbekannt.

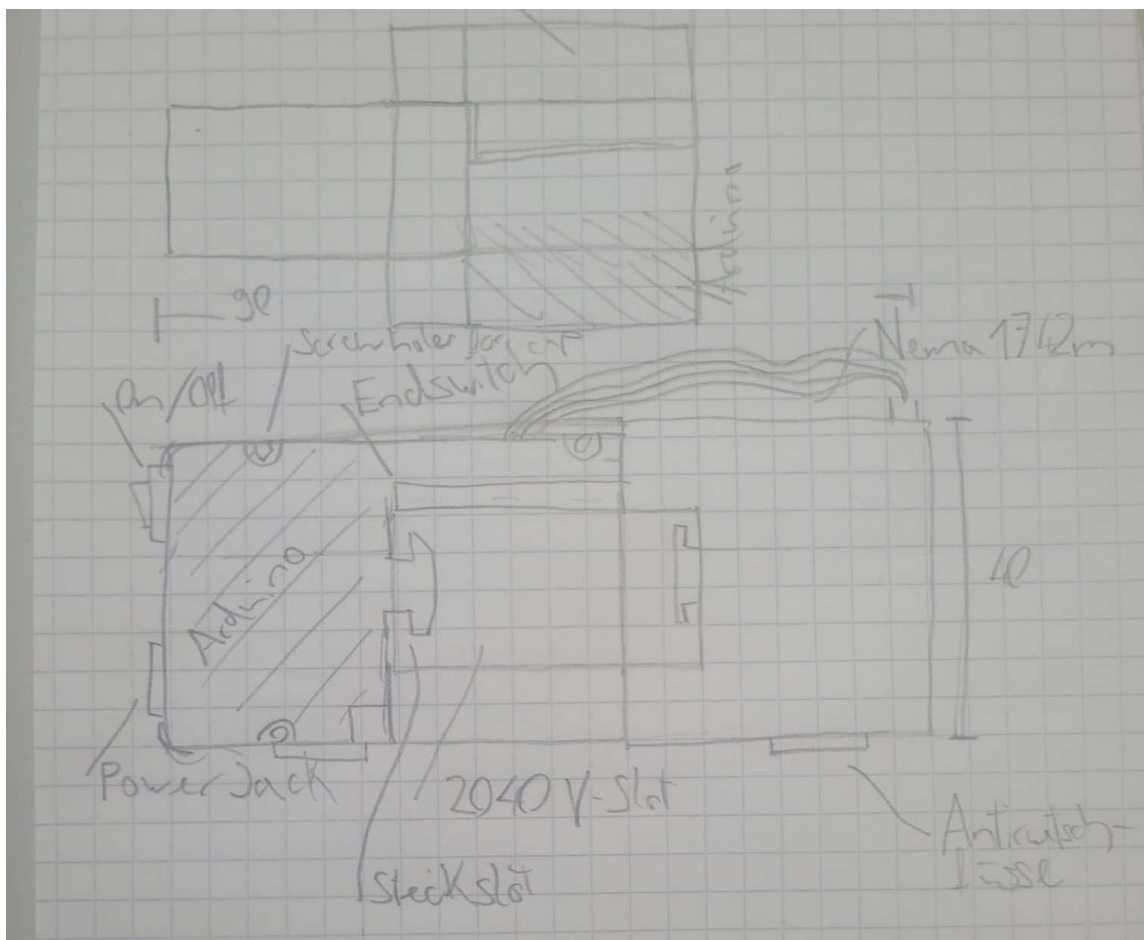


Abbildung 1 Originalfotografie der ersten Skizze

Mit dieser Skizze als Basis haben wir uns die Zeit genommen ein 3D Modell zu zeichnen. Um diese Modelle erstellen zu können, verwenden wir die Community Version der CAD Software Fusion 360 von Autodesk. Das Modell wurde Millimeter genau gezeichnet, da diese Teile mit dem 3D Drucker ausgedruckt werden. Um diese komplexen Teile passgenau drucken zu können, waren mehrere Iterationen nötig.

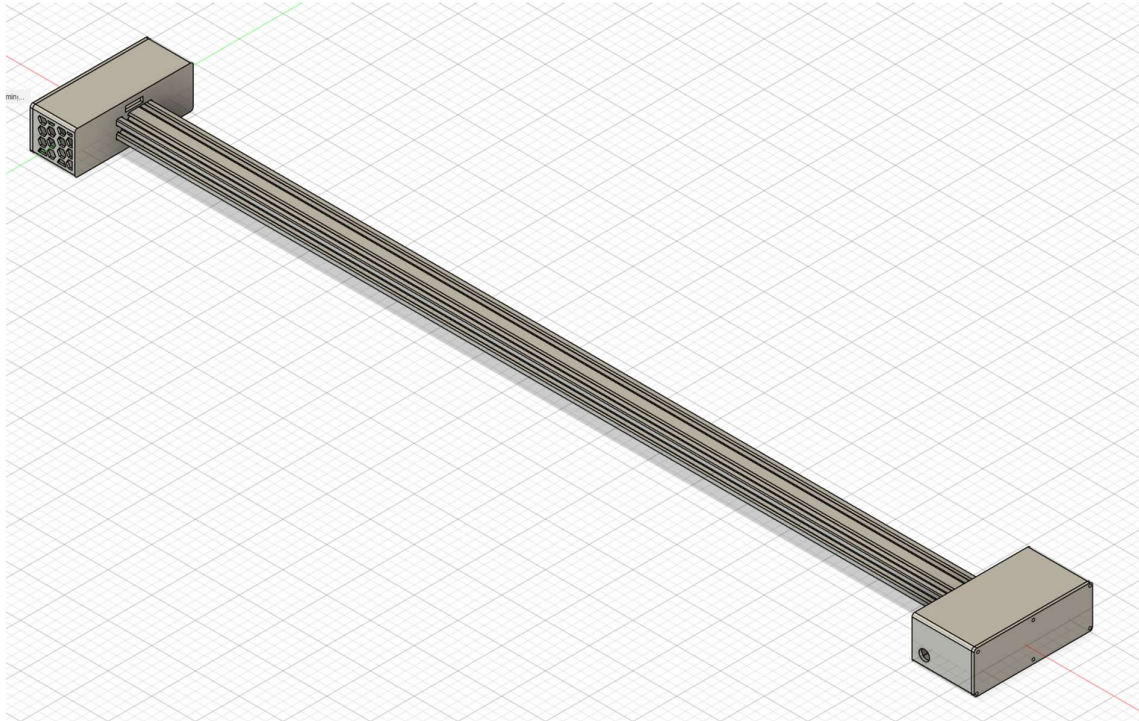


Abbildung 2 Finales 3D Modell des Sliders

So stellen wir uns den fertigen Kamera-Slider vor. Im rechten 3D gedruckten End-Teil, oder von uns auch Control Center genannt, wird die ganze Elektronik untergebracht, unter anderem der Motor und das Arduino, welche für die Funktionalität des Sliders verantwortlich sind. Links in der Abbildung ist das andere End-Teil. Dort ist nur ein End-Schalter und eine Umlenkrolle für den Keilriemen eingebaut.

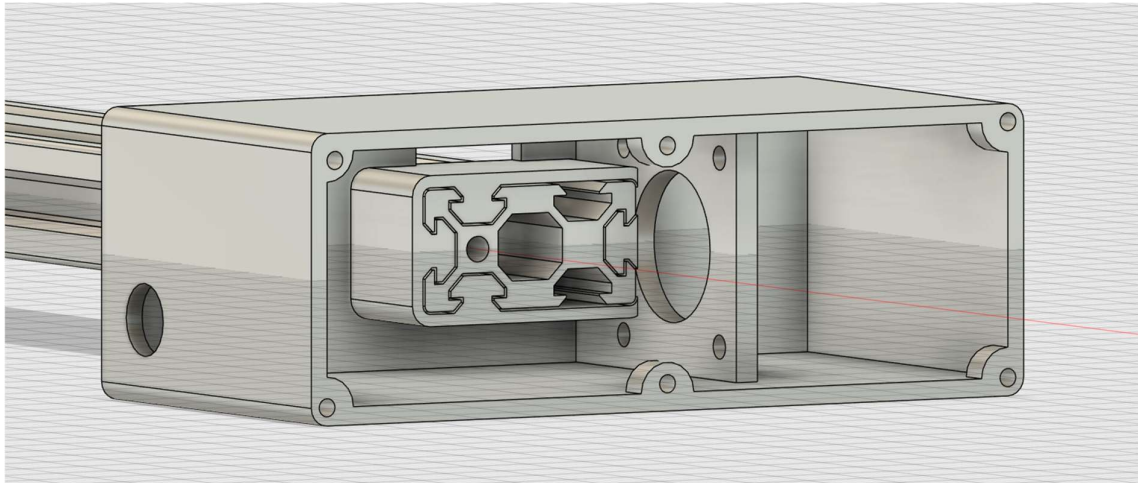


Abbildung 3 End-Teil mit Elektronik (Control Center)

Das Control Center besteht aus drei separaten Teilen, welche nach dem Ausdrucken zusammengesteckt werden können. Zur Einfachheit der Montage wird die Motorplatte unabhängig vom Gehäuse konstruiert. Der Motor wird auf der rechten Seite mit vier Schrauben an der Motorhalterung angebracht. Der Keilriemen wird später durch die Aussparungen in der Aluminiumschiene geführt.

Der Deckel kann mit vier bis sechs Schrauben am Gehäuse befestigt werden. Dies ist für beide End-Teile gleich.

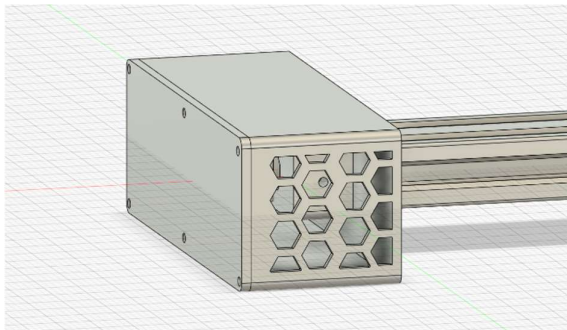


Abbildung 4 End-Teil mit Umlenkrolle

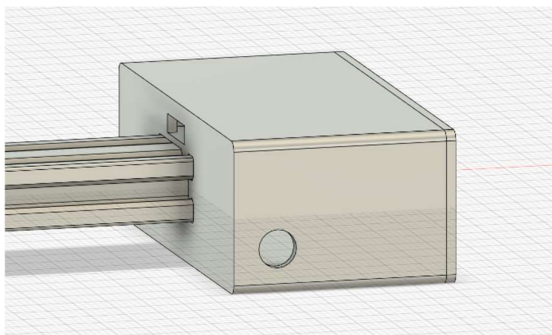


Abbildung 6 Control Center Seitenansicht



Abbildung 5 Strombuchse

Auf der linken Seite des Control Centers wird später die Strombuchse für den Slider montiert.

<sup>7</sup> Power Jack ([www.aliexpress.com](http://www.aliexpress.com)) 19.12.2023, direkt übernommen



Damit wir uns den Slider etwas besser vorstellen können, haben wir das 3D-Design in die Open Source Software Blender importiert.

Blender ist eine Grafiksoftware, mit welcher man Körper modellieren und animieren kann. Ausserdem kann man damit auch realistische Darstellungen von 3D Modellen erstellen. Wir haben uns genau diese Funktion zunutze gemacht und haben ein Bild des Modells erstellt.



*Abbildung 7 Rendering des Sliders*

### 5.2.2. 3D-Druck

Im folgenden Abschnitt erklären wir, wie wir unsere zuvor erstellten 3D Modelle auf den 3D Druck vorbereiten.

Um ein Modell drucken zu können, muss es zuerst in einem Slicer aufbereitet werden. Wir verwenden den Open Source Slicer «Prusa Slicer» von Prusa. Im Slicer muss der Drucker und die Materialien, mit dem das Modell gedruckt wird, konfiguriert werden. Ausserdem kann jedes Detail des Druckvorganges genau eingestellt werden. Das ist aber in den meisten Fällen überflüssig, da der Slicer bereits sehr solide Standardwerte bietet.

Da das Druckbett zu klein ist um alle Teile gleichzeitig zu drucken, müssen wir verschiedene Druckaufträge erstellen. Im ersten Auftrag ziehen wir die wichtigsten Teile auf das Druckbett.

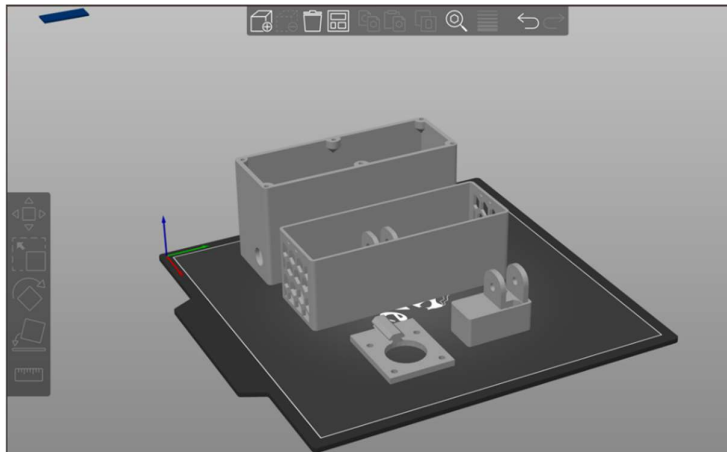


Abbildung 8 Druckauftrag 1

Wir haben lediglich die grundlegenden Einstellungen für den Druck vorgenommen. Hierzu zählen insbesondere die Auswahl der Düsendröße (0.20mm), die Konfiguration des Druckers (Creality Ender-3), sowie die Festlegung des verwendeten Filaments (PLA).

Für diese Einstellungen sind wiederum Standardwerte definiert, wie zum Beispiel die Flussmenge für das Filament oder die Druckbettgröße für den Drucker. Es ist jedoch nicht notwendig, diese selbst zu definieren, da entsprechende Profile vom Slicer bereitgestellt werden.

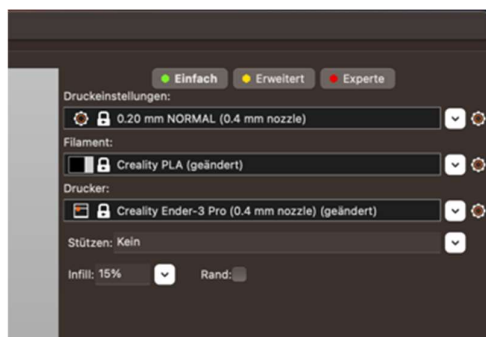


Abbildung 9 Slicer Grundeinstellungen

Nachdem alle Teile eingefügt und die entsprechenden Einstellungen vorgenommen wurden, ist es nun an der Zeit, das Projekt zu "slicen". Dieser Prozess bezeichnet die Umwandlung des 3D-Modells in Anweisungen, die der 3D-Drucker verarbeiten kann.

Anschliessend kann der Auftrag an den Drucker geschickt und gestartet werden.

### 5.2.3. Eigenes Protokoll

Das Problem bei der seriellen Kommunikation, so wie wir sie verwenden ist, dass jeder Charakter als einzelnes Byte ans Arduino gesendet wird.

Wenn man das Wort: "beispiel" ans Arduino sendet, interpretiert es jeden Buchstaben einzeln: «b», «e», «i», «s», «p», «i», «e», «l». Das ist für unseren Anwendungszweck nicht geeignet. Wenn wir zum Beispiel eine Position vom Web-UI ans Arduino senden wollen, wird bereits mehr als ein Zeichen in Anspruch genommen, da wir einen Startpunkt und einen Endpunkt senden müssen.

Damit wir mehr als nur ein einzelnes Zeichen senden oder empfangen können, haben wir ein simples Protokoll entwickelt. Wir haben im Code definiert, dass wir auf ein Startzeichen warten. Dies ist in unserem Fall «<» kleiner als. Das heisst, wenn ein solches Zeichen kommt, wissen wir, dass jetzt eine neue Nachricht beginnt.

Wir haben ebenfalls ein Schlusszeichen definiert: «>» grösser als. Wenn dieses Zeichen erscheint, wissen wir, dass die Nachricht komplett ist. Allen Charakteren, welche sich jetzt zwischen diesen beiden Zeichen befinden, werden nun zusammengefügt und entsprechen der ganzen Nachricht.

Zum Verständnis: Wenn wir jetzt eine Start-und End-Position ans Arduino senden wollen, müsste die Nachricht wie folgt aussehen: «<MOVE:start=10,end=35>». MOVE ist der auszuführende Befehl. Start und end sind beliebige Parameter die zusätzlich zum Befehl übergeben werden. Alles zusammen entspricht einer ganzen Nachricht.

#### Bandbreite

Da unser Protokoll auf Buchstaben basiert, braucht es viel mehr Bandbreite als ein Bit basiertes Protokoll. Wir sind diesen Kompromiss eingegangen, da bei uns die übertragene Datenmenge klein ist und es verständlicher ist.

Um mit unserem Protokoll diese Nachricht (<MOVE:start=10,end=35>) zu übermitteln, braucht es rund **25 Bytes**.

Mit einem rein binären Protokoll könnte man diese Zahl auf unter 5 Bytes bringen, da keine unnötigen Informationen übertragen werden, die nur zur Verständlichkeit dienen.

#### Error Korrektur

Bei der Übertragung von Daten besteht immer die Möglichkeit, dass einzelne Buchstaben oder Bits fehlerhaft übertragen werden. Das bedeutet, dass der Empfänger nicht genau das empfängt, was gesendet wurde. Um dies zu verhindern, integrieren viele Protokolle sogenannte Fehlerkorrekturmechanismen. Diese ermöglichen es, bis zu einem bestimmten Prozentsatz an korrekten Daten die ursprüngliche Nachricht wiederherzustellen.

Auch hier hat unser Protokoll einen Nachteil. Allerdings spielt es bei solch kleinen Datenmengen und für unsere Anwendungen kaum eine Rolle, wenn wir keine Error Korrektur verwenden.

## 6. Benutzeroberfläche<sup>8</sup>

Die Benutzeroberfläche ist einer der Schwerpunkte unseres Sliders, in diesem Kapitel erklären wir, wie sie aufgebaut ist.

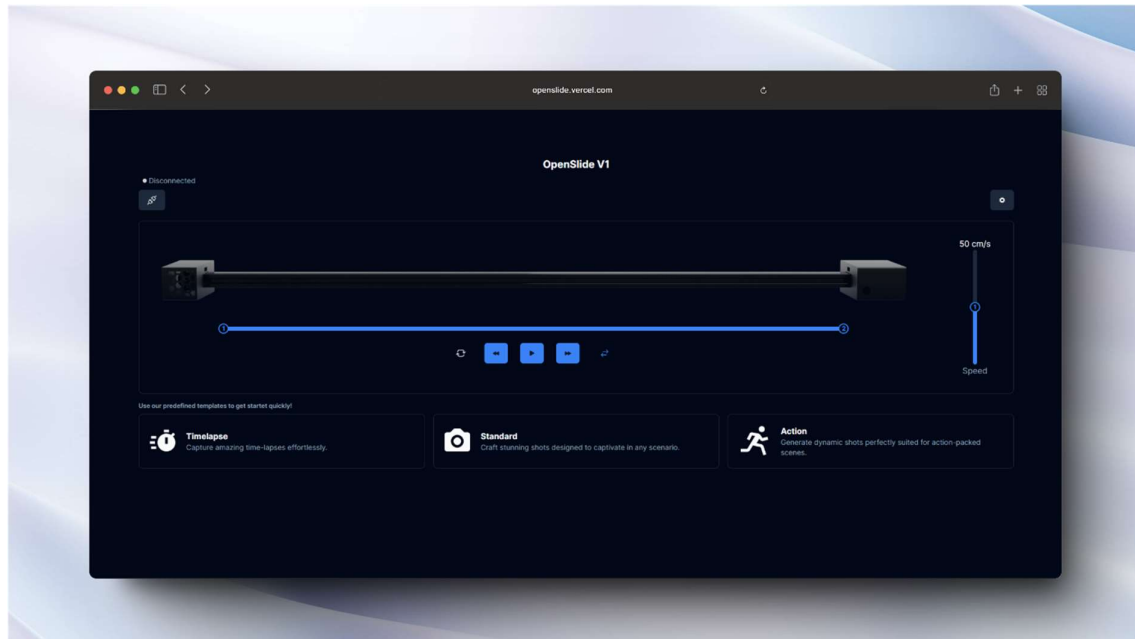


Abbildung 10 Benutzeroberfläche

### 6.1. Technologie

Für die Benutzeroberfläche haben wir Gebrauch von einigen nützlichen Technologien gemacht. Die Relevantesten haben wir unten kurz beschrieben.

#### 6.1.1. Next.js

Bei der Entwicklung des Web-UI haben wir auf NextJS gesetzt, NextJS ist ein Open-Source Framework, das von dem Unternehmen Vercel betreut wird. Dieses Framework erlaubt es React basierte Web-Anwendungen mit serverseitigem Rendering und statischer Website-Generierung zu erstellen. Die ganze Benutzeroberfläche wurde in der Sprache TypeScript geschrieben. TypeScript wurde von Microsoft entwickelt und fügt Typisierung zu JavaScript hinzu.

#### 6.1.2. Tailwindcss

Tailwind CSS ist ein beliebtes CSS-Framework, das auf Utility-Klassen basiert und die Gestaltung von Webseiten beschleunigt. Es ermöglicht uns, schnell und effizient responsive Designs zu erstellen, indem Klassen direkt im HTML-Code eingebettet werden, anstatt CSS-Klassen von Grund auf neu zu schreiben. Mit einer umfangreichen Dokumentation und einer aktiven Community hat sich Tailwind CSS zu einem starken Werkzeug für die Frontend-Entwicklung entwickelt.

---

<sup>8</sup> Benutzeroberfläche, Selbst erstellt

### 6.1.3. Vercel.com

Vercel, die Macher von NextJS bieten einen kostenlosen Service an wo wir unser Projekt bereitgestellt haben. Das heisst, sie sind verantwortlich, dass unsere Benutzeroberfläche auf folgender Adresse: [openslide.noekrebs.ch](https://openslide.noekrebs.ch) die ganze Zeit online und für Jeden verfügbar gemacht wird.

## 6.2. Features

Unser Web-UI bietet eine Vielzahl von nützlichen Funktionen für die Steuerung und Anpassung unseres Kamera-Sliders. Der Benutzer hat die volle Kontrolle über die Fahrt der Kamera auf dem Slider. Von wo sie startet und wohin sie sich bewegt, ist ganz nach den Wünschen des Benutzers einstellbar. Aber das ist noch nicht alles!

Der Benutzer kann nicht nur die Start und Zielpunkte festlegen, sondern auch die Geschwindigkeit definieren, mit der sich die Kamera über den Slider bewegen soll. Diese Flexibilität ermöglicht es, die Bewegung genau nach den eigenen Bedürfnissen anzupassen.

Ausserdem kann man auch die Bewegungsrichtung des Sliders ändern.

Eine weitere spannende Option ist der Loop-Modus. In diesem Modus fährt der Slider immer wieder von Start zu Endpunkt bis der Benutzer diese Funktion wieder deaktiviert. Diese Wiederholungsfunktion eröffnet aufregende Möglichkeiten für kreative Aufnahmen.

Wir haben eine Auswahl an vordefinierten Kombinationen von Geschwindigkeit und Position definiert, worauf man auch immer zurückgreifen kann, sollte man sich gerade nicht sicher sein welche Einstellungen zu der aktuellen Szene am Besten passen. Diese nennen wir Templates. Es gibt 3 verschiedene Varianten.

- **Timelapse:** Eignet sich hervorragend für Timelapse-Aufnahmen, das Cart fährt dementsprechend extrem langsam auf dem Slider, wobei aber immer noch flüssige und ruckfreie Aufnahmen entstehen.
- **Standard:** Dieser Modus eignet sich perfekt für die meisten Situationen und entspricht den Standardwerten.
- **Action:** Dieser Modus ist für eher schnellere, actionreiche Szenerien geeignet. Er ist schneller und fährt eine kürzere Distanz als der Standardmodus, um diese Emotionen in die Aufnahmen zu übertragen.

Das Web-UI hat eine zusätzliche wichtige Aufgabe: Es ist dafür verantwortlich, die Befehle an den Slider zu übertragen. Dies geschieht mit Hilfe der Web-Serial API. Nachdem der Benutzer den Slider über USB mit seinem Endgerät verbunden hat, werden die ausgeführten Befehle mit unserem eigenen Protokoll über USB an den Slider übertragen.

## 7. Arduino<sup>9</sup>

Das Arduino wird verwendet, um die Befehle, welche über das Web-UI gesendet werden effektiv umzusetzen. Das Arduino ist via USB-Kabel mit dem PC verbunden und kann darüber mit dem Web-UI kommunizieren.

Das Arduino muss in der Lage sein, die vom Web UI kommenden Befehle umzusetzen. Dabei muss es eventuell auch die Werte umrechnen. Das Web UI ist für jeden OpenSlide gleich, deswegen haben wir darauf geachtet, dass sich möglichst alle Hardware spezifische Logik auf dem Arduino befindet.

### 7.1. Kalibrierung

Das Arduino erfasst jede Position als Prozentwert von 0 bis 100, wobei 0 die ganz Linke Seite repräsentiert und 100 die rechte Seite. Da der Motor nach dem Einschalten nicht weiss, wo sich der Wagen momentan auf dem Slider befindet, erfolgt zunächst eine Kalibrierung. Der Motor startet bei Schritt 0 (nicht zwangsläufig ganz links oder rechts) und bewegt sich dann um eine bestimmte Anzahl von Schritten, bis er einen der Endschalter erreicht. An dieser Stelle setzt er seinen neuen Nullpunkt. Von diesem neuen Nullpunkt aus fährt er nun zum anderen Ende, um genau zu bestimmen, wie lang der Slider ist.

Von jetzt an kann der Slider Positionen in Form von Schritten entgegennehmen.

**Beispiel:**

- 0 = Ganz links
- 2000 = Ganz rechts
- 1000 = In der Mitte

Wir können nun diese Formel verwenden um den Wertebereich 0 – 100 in 0 – 2000 umzuwandeln.

$$Y = (x - in\_min) * (out\_max - out\_min) / (in\_max - in\_min) + out\_min$$

X ist der umzuwandelnde Wert. In\_min ist der minimale Wert von x (0), in\_max der maximale Wert von X (100). Out\_max (2000) ist der grösstmögliche Ausgabewert, out\_min (0) der Kleinste. Dem Motor wird jetzt der y-Wert übergeben und er wird sich an die gewünschte Position bewegen.

Dank dieser Umrechnung kann der Slider nun Positionen in Form von Prozentwerten verarbeiten und das Web-UI benötigt keine Informationen über die Länge des Sliders um ihn zu steuern.

### 7.2. Kommunikation

Das Arduino wird über unser eigenes Protokoll, welches im Kapitel «Eigenes Protokoll» ausführlich beschrieben wurde, angesteuert.

---

<sup>9</sup> Arduino, Selbst erstellt

## 8. Umsetzung<sup>10</sup>

Nachdem wir intensiv über Software und Hardware nachgedacht haben, dokumentieren wir in dem folgenden Kapitel die konkrete Umsetzung.

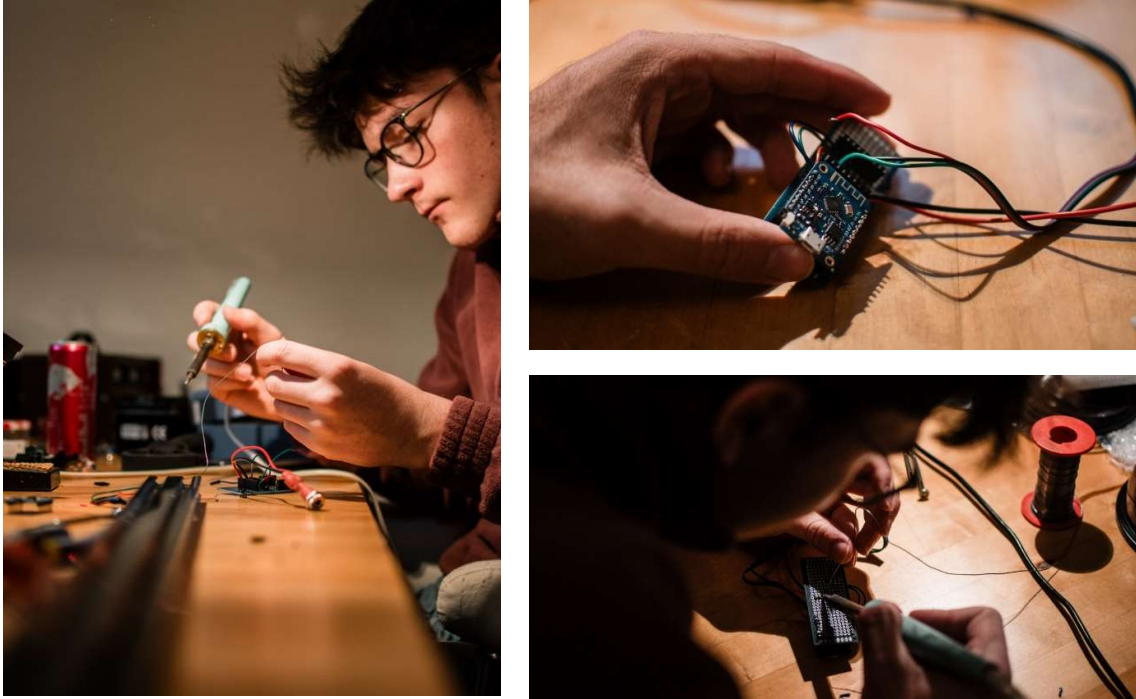


Abbildung 11:Einblick Umsetzung

---

<sup>10</sup> Umsetzung, Selbst erstellt

## 8.1. Control Center

Kern unseres Sliders ist der Arduino Micro Controller. Dieser besteht aus den drei unten auf-gezeigten Komponenten, welche ausschliesslich für die Ausführung der logischen Operatio-nen auf dem Kamera-Slider verantwortlich sind.

Den Treiber und das Arduino haben wir nicht direkt mit der Leiterplatte verlötet, sondern über Leiterplattenbuchsen. So können die beiden Komponenten bei einem Ausfall einfach ersetzt werden, ohne dass sie neu gelötet werden müssen.

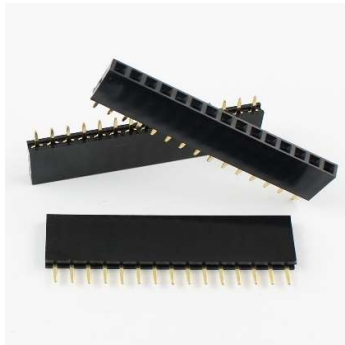


Abbildung 12 Leiterplattenbuchsen<sup>11</sup>

Um sicherzustellen, dass die Geräte miteinander kommunizieren können, wurden alle ge-nannten Teile miteinander durch versteckte Kabel verbunden.

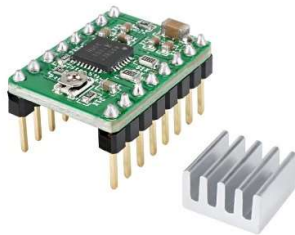


Abbildung 13 A4899<sup>12</sup>

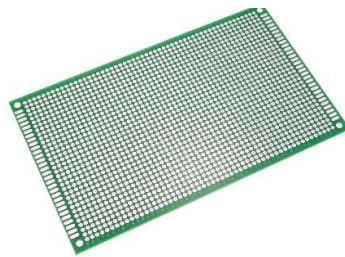


Abbildung 14 Leiterplatte<sup>13</sup>

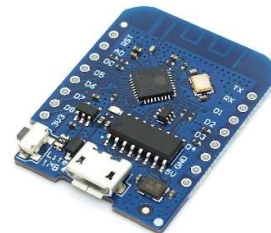


Abbildung 15 Arduino D1 Mini<sup>14</sup>

<sup>11</sup> Leiterplattenbuchsen (<https://forum.arduino.cc/>) 19.12.2023, direkt übernommen

<sup>12</sup> A4899 (<https://davincilab.co.za/products/a4988-stepper-driver>) 19.12.2023, direkt übernommen

<sup>13</sup> Platte (<https://www.amazon.de/>) 19.12.2023, direkt übernommen

<sup>14</sup> Arduino (<https://www.berrybase.ch/d1-mini-lite-esp8285-entwicklungsboard>) 19.12.2023, direkt übernommen



Damit wir genau wissen, welche Anschlüsse wir verbinden müssen, haben wir dieses Schema als Vorlage genommen.

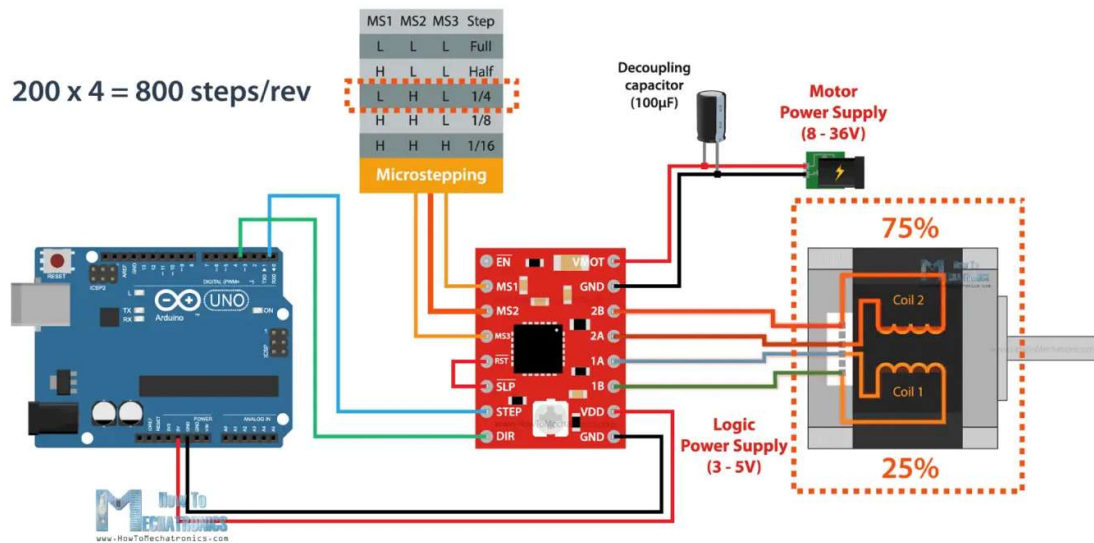


Abbildung 16 Verkabelung Driver, Stepper, Arduino<sup>15</sup>

Die ganze Steuereinheit wird durch ein 12v 2A Netzteil mit Strom versorgt. Damit das möglich ist, haben wir den Power Jack mit dem Driver verbunden. Nun würde das Konstrukt bereits funktionieren, allerdings könnten Schwankungen in der Spannung die elektronischen Teile beschädigen. Um das zu verhindern, schalten wir einen 1000 Mikروفarad-Kompensator direkt hinter dem Stromstecker dazwischen. Wo sich dieser allerdings befindet, spielt keine Rolle, solange er mit den Masse und 12V Versorgungspins verbunden ist.

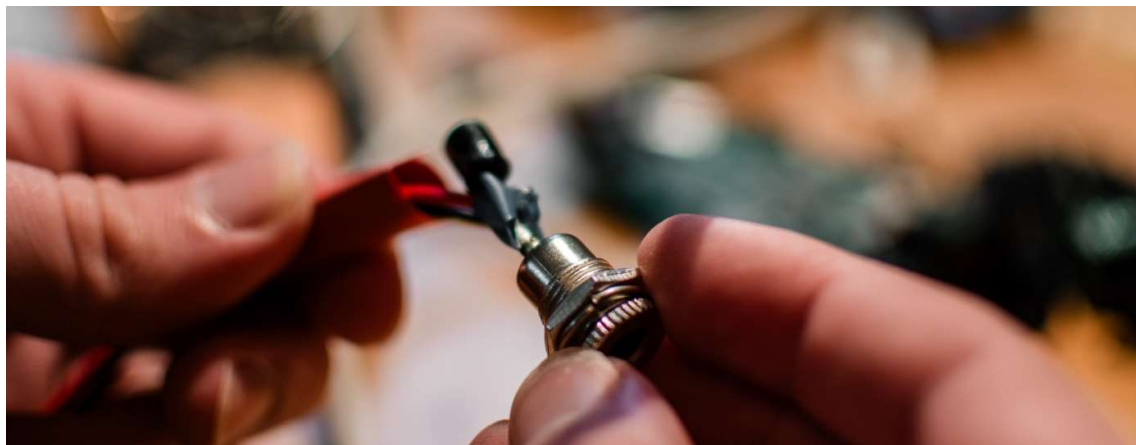


Abbildung 17 Power Jack mit Kompensator

<sup>15</sup> (Howtomechatronics <https://howtomechatronics.com>) 28.11.2023, direkt übernommen



Abbildung 18: Schrumpfschlauch anbringen

Um den Kondensator und das Kabel bestmöglich zu schützen habe wir den Power Jack isoliert. Die beiden Verbindungen im Kabel wurden jeweils mit Isolierband voneinander getrennt. Dies war notwendig damit kein Kurzschluss entstehen kann. Danach wurde noch ein Schrumpfschlauch am Kabel angebracht, welcher zusätzlich schützt.

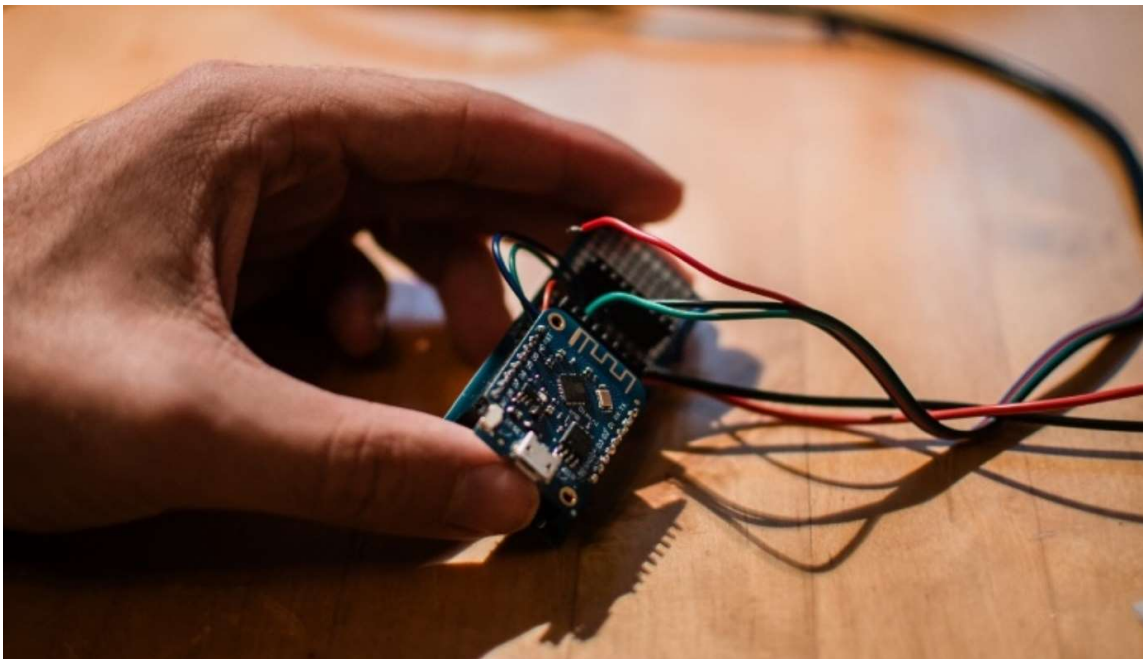
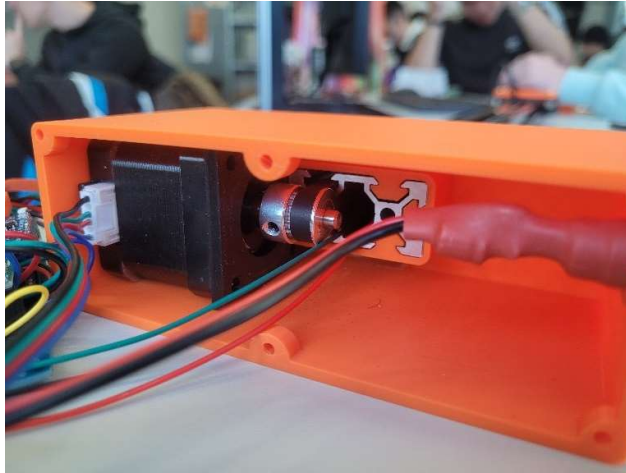


Abbildung 19: Steuereinheit



Daraufhin wurden alle Komponenten in die zuvor gedruckten Teile eingebaut. Auf der linken Seite ist der Nema 17 Stepper Motor zu sehen und auf der rechten Seite der Stromstecker.

Diese Ansicht ähnelt bereits sehr der Originalskizze.

Abbildung 20 Control Center aufbau

## 8.2. End Teil mit Umlenkrolle

Die Kabel der Endschalter wurden durch die Mitte der V-Slots Schienen geführt, so dass von aussen keine Kabel sichtbar sind. Auf dem Bild sind sie links neben der Rolle zu erkennen.

Die Umlenkrolle für den Riemen wurde mit Hilfe einer Schraube an der dafür vorgesehenen Halterung montiert. Der Riemen verläuft durch einen der V-Slots.



Abbildung 21 Umlenkrolle



Abbildung 22 Endschalter

Die Endschalter wurden auf beiden Seiten direkt über der Schiene in die dafür vorgesehene Aussparung verleimt, so dass sie vom Cart gedrückt werden können.

### 8.3. Kamerawagen

Auf dem Kamerawagen (Cart) wird später die Kamera montiert. Wie raffiniert er aufgebaut ist wird in folgendem Kapitel erklärt.



Abbildung 23 Wagen

Der Riemen wird auf dem Wagen mit Hilfe von zwei angeschraubten Platten gespannt und fixiert. Diese Platten werden auf die Oberfläche des Wagens gedrückt, wodurch der Riemen eingeklemmt wird und sich nicht mehr bewegen kann. Um die 3D-gedruckten Platten zu befestigen, wurden Gewinde eingeschmolzen.



Abbildung 24 Messing Gewinde zum einschmelzen <sup>16</sup>

<sup>16</sup> Threaded Inserts (<https://shop4fasteners.co.uk/>) 19.12.2023, direkt übernommen



### 8.3.1. Abdeckung

Damit die Kamera befestigt werden kann und der Slider etwas schöner daher kommt haben wir eine Abdeckung angefertigt, die wir mit zwei Schrauben am Cart montiert haben. Auch hier haben wir wieder Messing Gewinde verwendet.

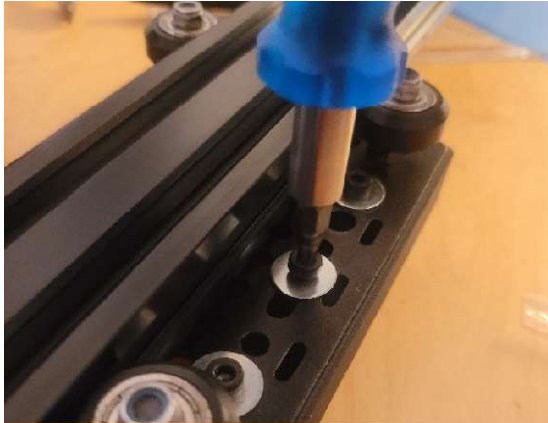


Abbildung 25 Montage Cart Abdeckung



Abbildung 26 Cart mit Abdeckung



Abbildung 27 1/4 Zoll Gewinde für Kamera

In der Mitte der Abdeckung wurde ebenfalls ein Gewinde eingeschmolzen. Es handelt sich um ein 1/4 Zoll Gewinde, da dies der Standard für Kameras ist.

Mit Hilfe von einem Multi-Filament/Farbdruck konnte oben links noch ein Label eingefügt werden.

## 9. Preisvergleich<sup>17</sup>

In folgendem Kapitel haben wir einen Preisvergleich zwischen dem OpenSlide V1 und dem [GVM Motorized Camera Slider](#) aufgestellt. Der GMV Slider ist ein typischer Mittelklasse Slider, der im Internet erhältlich ist.

Der Fokus liegt dabei nicht nur auf den finanziellen Aufwendungen, sondern auch darauf, welche Funktionen und Eigenschaften diese beiden Slider mit sich bringen.

### 9.1. Preisaufstellung

Um den Preis zu kalkulieren, haben wir für sämtliche Produkte aus der Materialliste die Kosten aufgeführt und im Anschluss summiert.

Nicht im Preis enthalten ist das Material, das vom 3D-Drucker für die Herstellung der Teile verwendet wurde. Dabei wurde nicht nur Material für das finale Produkt verwendet, sondern es mussten auch zahlreiche Modelle zum Zwecke der Entwicklung erstellt werden. Es handelt sich hierbei lediglich um wenige Franken, da das verwendete Material äusserst kostengünstig ist.

Tabelle 2 Preiskalkulation

Produkt	Bestellungen	Preis pro Stück (CHF)	Rabatt	Gesamtpreis (CHF)
<b>3D Printer Parts GT2 Timing Belt</b>	1	3.11	-	3.11
Double-headed 1/4" Screw Adapter	1	1.4	-32%	0.95
20PCS 1/4 Heat Set Insert Brass Nut	1	3.2	-3%	3.1
Nema 17 42 Stepper Motor	1	6.26	-35%	13
A4988 Stepper Motor Driver	1	0.92	-	0.92
1pc 2020-20100 TV-Slot Aluminum Profile	1	17	-	17
Endstop Switch For Arduino	6	1.97	-	11.82
12.6V 2A 18650 Battery Charger	1	4.19	-	4.19
Lieferung				25
<b>Total</b>				<b>79.09</b>

Preise können je nach Bestelldatum variieren.

<sup>17</sup> Preisvergleich, Selbst erstellt

## 9.2. Featurevergleich

In folgender Tabelle haben wir die wichtigsten Features und Eigenschaften zusammengefasst und den zuvor kalkulierten Preis eingefügt.

Tabelle 3 Slidervergleich

<b>Eigenschaft</b>	<b>OpenSlide V1</b>	<b>GMV<sup>18</sup> Slider</b>
<b>Preis</b>	79.-*	266.-
<b>Länge</b>	100 cm	83 cm
<b>Steuerung</b>	Webapp	Fernbedienung
<b>Akku</b>	Nein**	Ja
<b>Zeitraffer Funktion</b>	Ja	Ja
<b>Open Source</b>	Ja	Nein
<b>Lautstärke</b>	Je nach Geschwindigkeit eher laut und ist nicht für Aufnahmen mit Ton geeignet.	Angeblich leise und auch während Interviews gut verwendbar ***

\* für diesen Preis kann man sich lediglich die Komponenten kaufen. Der Slider kommt nicht fertig zusammengebaut und ist nicht funktionsfähig.

\*\* kann einfach erweitert werden mit einem standardmässigen 12v Akku.

\*\*\* basierend auf Informationen aus dem Internet.

<sup>18</sup> GMV Slider Info (<https://gvmled.com/gvm-slider-80/>), stark bearbeitet

### 9.3. Schlussfolgerung

Beide Slider weisen sowohl Vor- als auch Nachteile auf. Neben offensichtlichen Vorteilen wie Preis und Länge gibt es auch versteckte Aspekte. Der OpenSlide übertrifft insbesondere in Erweiterbarkeit und Anpassungsfähigkeit. Es besteht jederzeit die Möglichkeit, Funktionen von uns oder aus der Open-Source-Community hinzuzufügen, was ein enormes Potenzial bietet. Ein klarer Nachteil ist jedoch, dass der OpenSlide V1 nicht als fertiges Produkt erhältlich ist, im Gegensatz zum GMV Slider.

Der GMV Slider punktet auch im Bezug auf die Lautstärke. Die speziell für diesen Zweck entwickelten Motoren des GMV Sliders ermöglichen einen leiseren Betrieb im Vergleich zum OpenSlide. Ein weiteres Komfortmerkmal, das der OpenSlide V1 derzeit nicht bietet, ist ein integrierter Akku. Obwohl der OpenSlide auf Branchenstandards basiert und problemlos mit einem externen 12-Volt-Akku betrieben werden kann, ist dies beim GMV von Haus aus möglich.

In Bezug auf Softwarefunktionen hat der OpenSlide V1 wiederum klare Vorteile. Dies liegt daran, dass der gesamte Code Open Source ist und leicht erweitert werden kann. Der OpenSlide bietet bereits heute viele Einstellungsmöglichkeiten, so dass er genau an die jeweilige Situation angepasst werden kann. Der GMV kann zwar auch über eine Fernbedienung gesteuert werden, jedoch sind grössere Softwareupdates hier eher weniger zu erwarten.

Nachdem wir zahlreiche Fakten verglichen haben, bleibt die Frage: Welchen Slider sollte ich jetzt kaufen?

#### **Ich kaufe einen OpenSlide V1, wenn...**

- Ich viel über verwendete Komponenten, Code und Hardware lernen möchte und bereits mit den technischen Grundlagen vertraut bin.
- Ich einen kostengünstigen Slider für eine beliebige Kamera suche und Aufnahmen machen möchte, bei denen der Ton nicht oberste Priorität hat.

#### **Ich kaufe einen GMV Slider oder ähnliches, wenn...**

- Ich wenig selbst konfigurieren möchte und vor allem Aufnahmen machen will, bei denen der Ton oberste Priorität hat.
- Ich kein Interesse am Selbstbau habe, sondern einen einsatzbereiten Slider für meine Aufnahmen benötige.
- Ich bereit bin mehr Geld auszugeben.



## 10. Stress-Test<sup>19</sup>

Das Wichtigste am Kamera Slider ist seine eigentliche Funktion. Um herauszufinden, wo die Grenzen liegen, haben wir einige spannende Tests durchgeführt, um die Funktionalität des Sliders zu gewährleisten. Dabei geht es nicht nur um den Slider selbst, sondern auch um die Komponenten, die mit ihm zusammen verwendet werden, zum Beispiel die Kameras.

Damit wir die ganzen Kriterien beurteilen können, gibt es eine fünfstufige Gütestufenskala. Alle Tests werden mit dieser Skala bewertet. Dies ist auch für uns eine grosse Hilfe, da wir so herausfinden können welche Einstellungen wir zum Beispiel für die maximale Geschwindigkeit verwenden können. Hierbei geht es nicht darum den Slider möglichst schnell zu machen, sondern eine Einstellung zu finden, welche zwar eine hohe Geschwindigkeit ermöglicht, gleichzeitig aber auch zuverlässig funktioniert.

Tabelle 4 Gütestufen

Hervorragend
Gut
Befriedigend
Schlecht
Sehr schlecht

### 10.1. Geschwindigkeit

Hier haben wir versucht die optimale maximale Geschwindigkeit für den Slider herauszufinden.

Tabelle 5 Vergleich Geschwindigkeit

Test	Gütestufe	Beschreibung	Auswertung
<b>Geschwindigkeit auf 1500 spp</b>	Schlecht	Der Motor wird mit einer Maximalgeschwindigkeit von 1500 spp konfiguriert.	Der Motor läuft sehr zuverlässig, jedoch sind die einzelnen Schritte des Motors in den Aufnahmen erkennbar.
<b>Geschwindigkeit auf 3000 spp</b>	Hervorragend	Der Motor wird mit einer Maximalgeschwindigkeit von 3000 spp konfiguriert.	Die Zuverlässigkeit zum vorherigen Test hat nicht abgenommen, die Geschwindigkeit hat sich enorm verbessert. Der Slider kann so optimal verwendet werden.
<b>Geschwindigkeit auf 3500 spp</b>	Befriedigend	Der Motor wird mit einer Maximalgeschwindigkeit von 3500 spp konfiguriert.	Die Zuverlässigkeit nimmt drastisch ab, der Motor kennt teilweise die aktuelle Position nicht mehr. Die Geschwindigkeit ist zwar nochmals höher, schlussendlich ist das Risiko auf Versagen zu hoch, um die Geschwindigkeit zu rechtfertigen.

*\*spp steht für «steps per seconds» also die Anzahl Schritte, welcher der Motor in einer Sekunde macht.*

<sup>19</sup> Stress-test, Selbst erstellt

## 10.2. Kameras

Hier haben wir getestet welche Kamera sich besonders eignet für unseren Slider. Für diesen Test wurden drei verschiedene Arten von Kameras auf dem Slider montiert.

Test	Gütestufe	Beschreibung	Auswertung
<b>GoPro Hero 11</b>	Hervorragend	Hier wurde der Slider in Kombination mit der GoPro Hero 11 verwendet.	Der Slider hat mit der GoPro keinerlei Probleme, er kann die Kamera problemlos hin und her bewegen. In den Aufnahmen sind keinerlei Ruckler zu erkennen.
<b>bmppcc 4k</b>	Hervorragend	Hier wurde der Slider in Kombination mit der Blackmagic Design Pocket Cinema Camera 4K verwendet.	Der Slider hatte auch mit dieser Kamera keine Probleme. Aufgrund des grösseren Gewichts der Kamera hätten wir erwartet, dass der Slider langsamer oder träger reagiert. Das war aber nicht der Fall.
<b>Smartphone</b>	Hervorragend	Hier wurde der Slider in Kombination mit dem Samsung Galaxy S20 FE verwendet.	Auch hier hatte der Slider keine Probleme beim Bewegen des Smartphones. Auch die Aufnahmen weisen hier keinerlei Ruckler auf.

Tabelle 6 Kameravergleich



Abbildung 28 Blackmagic Design Pocket Cinema Camera 4K



Abbildung 29 GoPro Hero 11

## 11. Veröffentlichung<sup>20</sup>

Von Anfang an war es uns ein Anliegen alle Pläne und Dokumente öffentlich zugänglich zu machen, so dass sich jeder ein «OpenSlide» bauen kann. Dieses Projekt eignet sich sehr dafür, da wir auf herkömmliche Produkte gesetzt haben, die sich jeder bestellen kann. So kann also jemand mit Hilfe unserer Anleitung, Pläne und Code einen eigenen Slider bauen.

Es ist aber für andere auch möglich unser Code anzupassen und eine Einfügung in das Projekt zu beantragen. So kann das Projekt auch nach der Vertiefungsarbeit stetig verbessert werden.

### 11.1. GitHub

GitHub ist eine webbasierte Plattform für die Versionsverwaltung von Quellcode und bietet Funktionen für Zusammenarbeit und Nachverfolgung von Änderungen in Softwareprojekten. Entwickler können ihre Projekte auf GitHub verwalten, gemeinsam arbeiten und Features wie Pull-Requests und Wikis nutzen.

Genau diese Funktionen nutzen wir für Opensilde intensiv. Daher haben wir sämtliche relevanten Daten in ein GitHub-Repository hochgeladen.

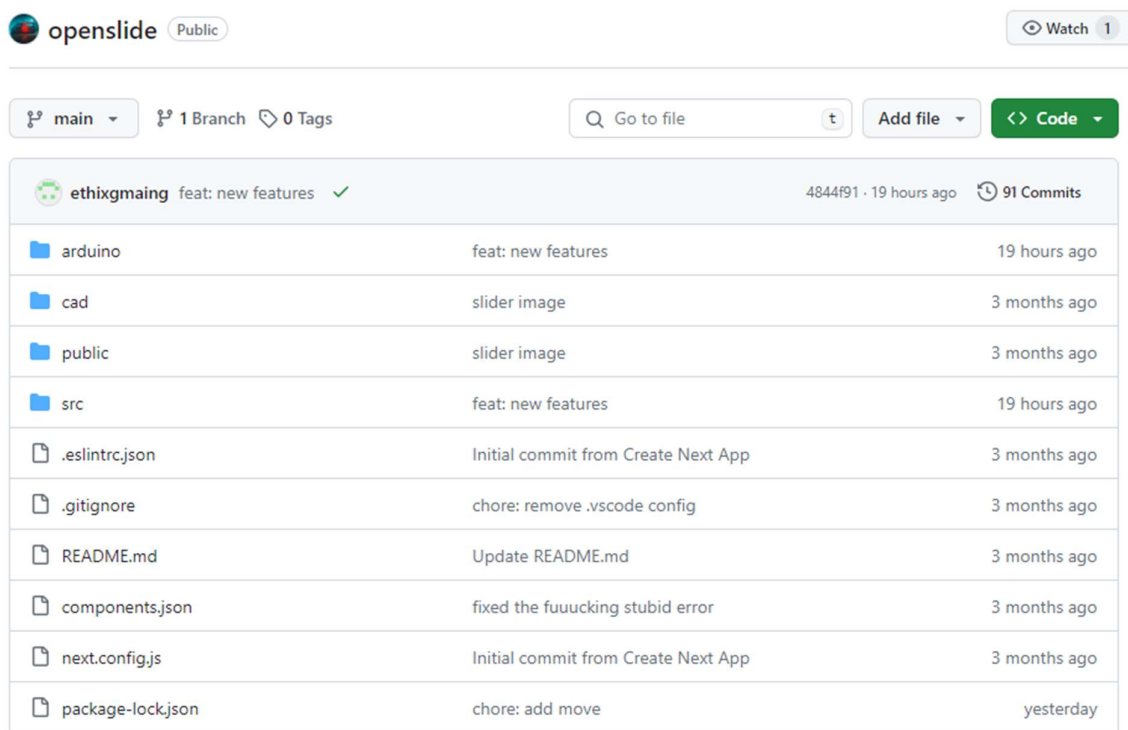


Abbildung 30 OpenSlide Github Repository

<sup>20</sup> Veröffentlichung, Selbst erstellt

## 11.2. Erreichbarkeit

Das Repository ist für jeden auf GitHub unter [github.com/sempex/openslide](https://github.com/sempex/openslide) zugänglich, unter [openslide.noekrebs.ch](https://openslide.noekrebs.ch), ist das Web UI erreichbar.

Mit diesen beiden Quellen ist es möglich, den Slider nachzubauen und anschliessend zu steuern.

Tim und Noé wünschen viel Spass beim Nachbauen und Experimentieren!

## 12. Glossar<sup>21</sup>

Begriff	Definition / Erklärung
<b>3D Drucker</b>	3D-Druck ist ein Fertigungsverfahren, bei dem dreidimensionale Objekte schichtweise aufgebaut werden, indem Materialien wie Kunststoff oder Metall schichtweise aufgetragen oder verfestigt werden.
<b>Frontend Framework</b>	Frontend Frameworks erleichtern die Entwicklung sogenannter Single-Page-Webanwendungen.
<b>Github</b>	GitHub ist eine Plattform zur Speicherung von Code, für Versionskontrolle und Zusammenarbeit.
<b>Kamera-Slider</b>	Ein Kamera-Slider ist ein Werkzeug, mit dem Sie Ihre Kamera sanft und gleichmässig entlang eines Schienensystems bewegen können.
<b>Open Source</b>	Open Source bezieht sich auf Software, deren Quellcode frei verfügbar ist, so dass Benutzer ihn anzeigen, verwenden, ändern und verteilen können. Es kann sich hierbei auch um andere digitale Produkte handeln.
<b>Pull Requests</b>	Ein Pull Request ist eine Anfrage, die ein Entwickler stellt, um Änderungen, die er an einem Repository vorgenommen hat, in das Original-Repository zu integrieren.
<b>Repository</b>	Ein Repository ist das grundlegendste Element auf GitHub. Es ist ein Ort, an dem man Code, Dateien und die Versionshistorie jeder Datei speichern kann.

---

<sup>21</sup> Glossar, Selbst erstellt

## 13. Abkürzungsverzeichnis<sup>22</sup>

Abkürzung	Definition / Erklärung
API	Application Programming Interfaces
CAD	Computer Aided Design
PCB	Printed Circuit Board
PLA	Poly lactide

---

<sup>22</sup> Abkürzungsverzeichnis, Selbst erstellt

## 14. Schlusswort<sup>23</sup>

Der Gesamteindruck der Umsetzung dieses Projekts ist sehr positiv. Wir haben dabei eine Menge neue Sachen gelernt, was aber auch relativ zeitaufwendig war. Wir mussten sehr viele verschiedene Dinge in kürzester Zeit lernen und umsetzen. Das Projekt war vom Aufwand her sehr gross, was wir aber mit unserer Begeisterung und Leidenschaft für das Projekt wieder wettmachen konnten. Wir haben selbst eine Riesenfreude an dem fertigen Ergebnis und können dies nun auch beim Produzieren von Videos verwenden.

Wir haben meistens zusammengearbeitet, was sich enorm bewährt hat. Durch die stetige Zusammenarbeit hatten wir allgemein mehr Spass an der Umsetzung und wir kamen viel schneller voran, da wir zu zweit einer Menge Probleme vorbeugen konnten. Wir konnten uns stetig gegenseitig mit unserem Wissen ergänzen.

Mit der fertigen Umsetzung des Projekts haben wir den Grundstein gelegt, um in Zukunft ein OpenSource Projekt zu pflegen, wo verschiedenste Leute ihren eigenen «OpenSlide» bauen können, da wir alle Baupläne, Komponenten und Sourcecode kostenlos zur Verfügung stellen. Als Informatiker profitieren wir jeden Tag von kostenlosen Inhalten und können somit auch etwas zurückgeben.

### 14.1. Abweichung von den Zielen

Wir haben Ziel drei etwas anders umgesetzt als ursprünglich geplant. Wir haben ausführliche Tests mit unserem Slider durchgeführt und in einem Bewertungsraster dokumentiert, jedoch konnten wir diese Tests nicht mit einem anderen Slider vergleichen. Dafür haben wir jedoch eine zweite Tabelle erstellt, in der wir Fakten und Daten unseres Sliders mit Angaben zu anderen Slidern aus dem Internet verglichen. So können wir mit einer kleinen Abweichung zur ursprünglichen Planung das Ziel bestmöglich abdecken.

Der Kamera Slider ist momentan noch nicht mit dem Handy steuerbar, da dies mit der Implementation der seriellen Kommunikation aktuell nicht möglich ist. Dies möchten wir aber in Zukunft, wie im «weiteren Vorgehen» beschrieben, verbessern.

---

<sup>23</sup> Schlusswort, Selbst erstellt

## 14.2. Weiteres Vorgehen

Wir werden auch nach der Vertiefungsarbeit an dem OpenSlide weiter arbeiten, da wir so begeistert davon sind. Wir planen, die Steuerung des Sliders kabellos zu konstruieren. Hierfür möchten wir die aktuelle kabelgebundene Kommunikation auf Bluetooth umstellen. So wäre auch die Steuerung mit einem Smartphone möglich.

Kurz vor Abschluss der Umsetzung fiel uns auf, dass die Befestigung der Kamera noch nicht optimal ist. Daher haben wir nach einer Lösung gesucht und einen Kugelkopf bestellt. Dieser ermöglicht eine einfachere Einstellung der Kameraposition. Leider wird er voraussichtlich erst nach der VA-Zeit eintreffen, sodass wir diese Verbesserung erst bei der Präsentation vorstellen können.

## 14.3. Bearbeitungsgrad

Bei jedem Hauptkapitel wurde der Bearbeitungsgrad (1., 2., ...) angegeben. Falls eine Spezifizierung für ein Unterkapitel nötig wäre, wird bei diesem der Bearbeitungsgrad explizit angegeben. Ansonsten gilt die Angabe des Hauptkapitels für alle Unterkapitel.

Bilder, bei denen keine spezifische Quelle angegeben ist, wurden selbst erstellt. Bilder mit Quellen sind jeweils mit einem Bearbeitungsgrad gekennzeichnet.



## 15. Abbildungsverzeichnis<sup>24</sup>

Abbildung 1 Originalfotografie der ersten Skizze .....	14
Abbildung 2 Finales 3D Modell des Sliders.....	15
Abbildung 3 End-Teil mit Elektronik (Control Center).....	16
Abbildung 4 End-Teil mit Umlenkrolle .....	16
Abbildung 6 Control Center Seitenansicht .....	16
Abbildung 5 Strombuchse .....	16
Abbildung 7 Rendering des Sliders .....	17
Abbildung 8 Druckauftrag 1 .....	18
Abbildung 9 Slicer Grundeinstellungen .....	18
Abbildung 10 Benutzeroberfläche.....	20
Abbildung 11:Einblick Umsetzung .....	23
Abbildung 12 Leiterplattenbuchsen .....	24
Abbildung 13 A4899 .....	24
Abbildung 14 Leiterplatte.....	24
Abbildung 15 Arduino D1 Mini .....	24
Abbildung 16 Verkabelung Driver, Stepper, Arduino.....	25
Abbildung 17 Power Jack mit Kompensator .....	25
Abbildung 18: Schrumpfschlauch anbringen.....	26
Abbildung 19 Steuereinheit .....	26
Abbildung 20 Control Center aufbau.....	27
Abbildung 21 Umlenkrolle.....	27
Abbildung 22 Endschalter.....	27
Abbildung 23 Wagen .....	28
Abbildung 24 Messing Gewinde zum einschmelzen .....	28
Abbildung 25 Montage Cart Abdeckung .....	29
Abbildung 26 Cart mit Abdeckung .....	29
Abbildung 27 1/4 Zoll Gewinde für Kamera .....	29
Abbildung 28 Blackmagic Design Pocket Cinema Camera 4K.....	34
Abbildung 29 GoPro Hero 11.....	34
Abbildung 30 Openslide Github Repository .....	35

<sup>24</sup> Abbildungsverzeichnis, Selbst erstellt

## 16. Tabellenverzeichnis<sup>25</sup>

Tabelle 1 Material.....	12
Tabelle 2 Preiskalkulation.....	30
Tabelle 3 Slidervergleich .....	31
Tabelle 4 Gütestufen .....	33
Tabelle 5 Vergleich Geschwindigkeit .....	33
Tabelle 6 Kameravergleich .....	34

---

<sup>25</sup> Tabellenverzeichnis, Selbst erstellt

## 17. Schlusserklärung

„Hiermit versichern wir, dass die vorliegende Arbeit selbstständig angefertigt wurde. Alle Quellen und beanspruchten Hilfeleistungen, welche den Inhalt der Arbeit beeinflussen, sind am Ende eines Kapitels deklariert. Die Erarbeitungsgrade entsprechen der Wahrheit. Wir sind uns bewusst, dass der Einsatz von KI-Schreibwerkzeugen keine Garantie für die Qualität von Inhalten und Text darstellt.“

Tim Reber

Noe Krebs

Unterschrift

Unterschrift

## 18. Arbeitsjournal<sup>26</sup>

KW/Datum	Zeit	Wer	Tätigkeit	Ort	Details	Reflexion & Pendenzen
KW 42 19.10.2023	2h	Tim & Noé	<ul style="list-style-type: none"> <li>- Planung der Grobplanung</li> <li>- Vorbereiten des Dokuments</li> </ul>	GIBB	Wir konnten einen Plan erstellen, wann wir welche Aufgaben erledigen möchten. Somit kennen wir die nächsten Schritte und wissen was wir in den folgenden Wochen alles zu tun haben.	Wir dokumentieren unser Konzept, und schliessen die Planung des Projekts ab. Wenn dies fertig ist, können wir mit der Umsetzung unseres Projekts beginnen.
KW 43 26.10.2023	2h	Tim & Noé	<ul style="list-style-type: none"> <li>- Aufbau und formatieren des Dokuments</li> </ul>	GIBB	Wir hatten einige Probleme mit der Formatierung und Nummerierung der Überschriften. Wir mussten vieles neu erstellen, damit die Vorlagen korrekt angewendet wurden. Nun sollte aber das Dokument korrekt sein und in Zukunft besser funktionieren.	Mit den ersten Arbeiten an dem Slider beginnen und diese gleich dokumentieren.
KW 44 01.11.2023	2h	Tim & Noé	<ul style="list-style-type: none"> <li>- Initialisierung des Web-UI</li> <li>- Prototyp des Web-UI deployen</li> <li>- Dokumentation nachführen</li> </ul>	GIBB	Es ist uns leicht gefallen die Anforderungen an das Web UI zu definieren, da wir genau wussten, was wir brauchen. Schlussendlich haben wir diese dann als Features dokumentiert.	Nächstes Mal werden wir bereits mit dem Zusammenbau der Teile beginnen.
KW 45	4h	Tim & Noé	<ul style="list-style-type: none"> <li>- Beginn der Umsetzung</li> <li>- Dokumentation nachführen</li> <li>- Anpassung Arduino Code</li> </ul>	Su-mis-wald	Zu Beginn sind wir sehr gut vorangekommen. Allerdings haben wir lange nach der korrekten Verbindung zwischen Stepper Motor und Driver gesucht. Wie sich aber später herausgestellt hat, war das eigentliche Problem eine Fehlkonfiguration im Code. Wir werden in Zukunft noch besser darauf achten,	Wir haben alle Teile kontrolliert und mit dem Zusammenbau begonnen. Der Motor kann nun via Driver mit dem Arduino angesteuert werden. Beispiel code im Arduino Skript eingefügt, um die Funktionalität zu testen.

<sup>26</sup> Arbeitsjournal, Selbst erstellt

					dass wir jeden Arbeitsschritt genau kontrollieren.	
KW46	2h	Tim & Noé	<ul style="list-style-type: none"> <li>- Dokumentation Umsetzung erster Teil</li> <li>- Rendern des 3D Modells</li> <li>- Slider in Web UI integrieren.</li> <li>- Anpassen des 3D Modells</li> </ul>	GIBB	<p>Wir haben begonnen, die Erfahrungen und Ereignisse von letzter Woche zu dokumentieren.</p> <p>Wir konnten das gerenderte 3D-Modell ins Web-UI integrieren, dies trägt dem Verständnis der Benutzung der Oberfläche bei.</p>	Es war spannend aufzuzeigen, was wir zusammengebaut haben. Wir konnten viele Fortschritte machen, mussten aber sehr viel löten, was zu diversen Kopfschmerzen führte. Das nächste Mal sollten wir eventuell eine Maske tragen.
KW47	2h	Tim & Noé	<ul style="list-style-type: none"> <li>- Arduino Logik</li> <li>- 3D Print Einstellungen</li> </ul>	GIBB	Heute haben wir uns auf das Programmieren des Arduinos fokussiert. Wir haben den Stepper Motor angesteuert und erstmals mit dem Web UI verbunden.	Zu Beginn hatten wir einige Probleme mit der Steuerung über das Web-UI. Dank unserer eigens erstellten Debugging-Konsole, konnten wir diese jedoch schnell und einfach beheben. Wir haben bereits eine solide Grundlage gelegt, aber es gibt noch einiges zu tun. Das nächste Mal werden wir mit der Programmierung fortfahren und möglicherweise die Kalibrierungsmethode implementieren. Es gelingt uns gut, zusammen an denselben Aufgaben zu arbeiten.
KW 48	2h	Tim & Noé	<ul style="list-style-type: none"> <li>- Slider Kalibrierung</li> </ul>	GIBB	Heute haben wir den Code geschrieben, der für die Kalibrierung des Sliders nötig ist.	Wir haben zuerst zusammen auf Papier skizziert, wie die Kalibrierung genau aufgebaut sein soll. Bei diesem Prozess haben wir beide unser Wissen

						eingesetzt. So zusammenzuarbeiten hat Spass gemacht.
KW 49	2h	Tim & Noé	- Funktionstests	GIBB	Wir haben die Kommunikation zwischen Web UI und Slider ausführlich getestet und anschliessend die Software erweitert. Dabei haben wir besonderen Fokus auf die Stabilität gelegt.	<p>Während der Tests wurde das Arduino und der Motor Driver zerstört. Dies war ein sehr grosser Rückschlag für uns beide. Allerdings haben wir bereits mit solch einer Situation gerechnet und konnten dank unseres modularen PCB-Designs die Komponenten mit einem Steckmechanismus austauschen. Es hat uns gefreut, dass sich unser sorgfältig entwickeltes PCB ausgezahlt hat.</p> <p>Wir werden das nächste Mal noch weiter daran arbeiten, den Slider möglichst stabil zu entwickeln.</p>
KW 50	2h	Noé & Tim	<ul style="list-style-type: none"> <li>- Optimierung</li> <li>- Lasttests</li> </ul>	GIBB	<p>Da wir in unseren Zielen definiert haben, dass wir unser Slider auch ausführlich testen wollen, haben wir uns heute dafür Zeit genommen. Wir haben die maximale Geschwindigkeit, Kraft und weitere Eigenschaften getestet.</p> <p>Zusätzlich haben wir eine genaue Zeitachse definiert, welche uns dabei helfen soll, unsere Arbeit erfolgreich abzuschliessen.</p> <ul style="list-style-type: none"> <li>- Di, 19.12.23 16 Uhr Dokument Fertig als PDF</li> </ul>	<p>Wir haben den Plan so erstellt, dass wir am Schluss genug Zeit für die Finalisierung haben. Dies ist uns heute zugutegekommen.</p> <p>Da in der Grobplanung nur die Wochen geplant sind, haben wir noch eine Spezifizierung gemacht, um eine reibungslose Abgabe der Arbeit zu ermöglichen. Das ist wichtig, damit uns nichts entgeht.</p>

					<ul style="list-style-type: none"><li>- Mi 20.12.23 ausdrucken und heften in der Firma.</li><li>- Do 21.12.23 Abgabe</li></ul>	
--	--	--	--	--	--	--

## 19. Anhang

All unser Code und sonstige Dateien sind wie beschrieben auf GitHub zu finden.