Teil 1: Lehrveranstaltung zu einem Thema der Mediengestaltung

Dauer	Inhalt
05 min	Einleitung/Einordnung
15 min	Vorstellung Thema
05 min	Demonstration
15 min	Kurze Übung
05 min	Vorstellung einiger Ergebnisse aus Übung
05 min	Zusammenfassung, Ausblick und Fragen

Teil 2: Präsentation eines medialen Produktes

Geplanter Aufbau

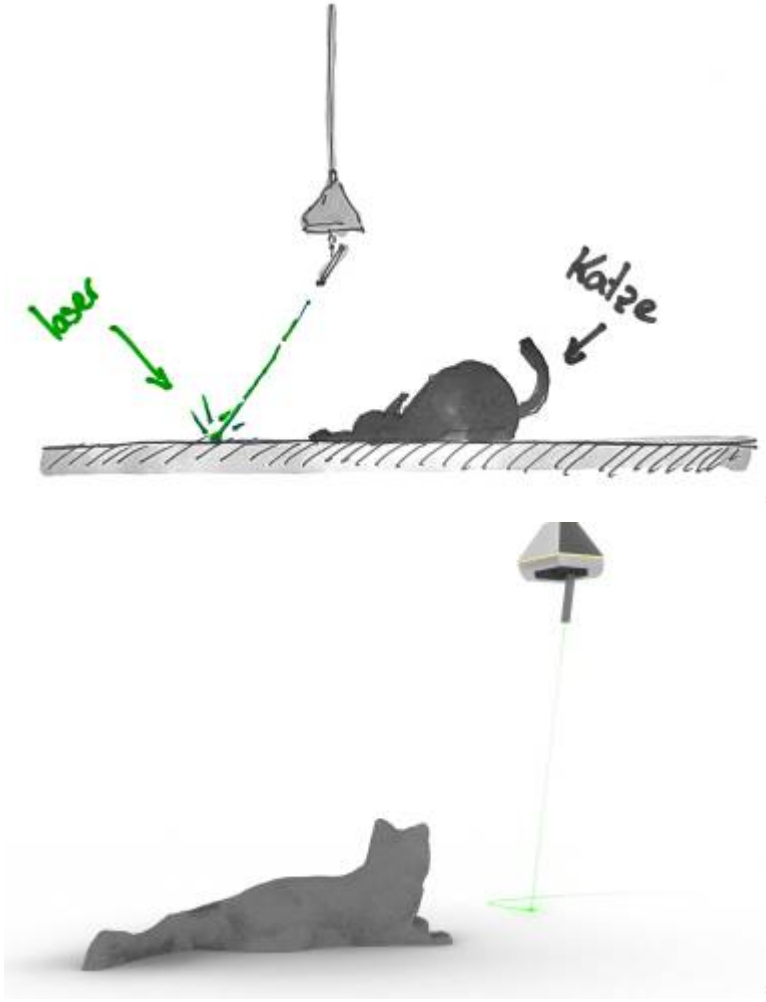
Dauer	Inhalt
05 min	Herleitung und Konzept (Skizzen, 3D-Visualisierung)
05 min	Einblick in Entwicklung des Prototypen (Hardware, Software)
05 min	Demonstration
02 min	Fazit
03 min	Fragen

Die Basis für das “*auf kreative Weise gestaltete Produkt auf Basis eines Raspberry Pi*” bildet ein Gehäuse mit einem integrierten E27-Gewinde. Hiermit kann das Objekt in jedem Zuhause in ein handelsübliches Lampengewinde eingesetzt und betrieben werden. Betätigt man den entsprechenden Lichtschalter startet das integrierte Raspberry Pi. Die auf der unteren Seite revisionierbare und anpassbare Fläche läßt diverse Möglichkeiten zu eigene Hardware zu ergänzen und zu betreiben. Es sind unendlich viele Anwendungsmöglichkeiten denkbar. Was wäre Ihre Idee?

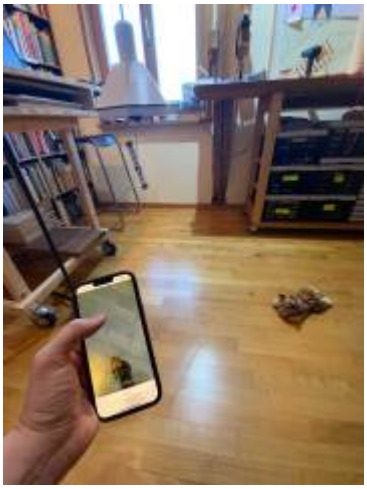
Personalisierbare farbige Ringe (siehe animiertes Gif) lassen das eigene Objekt schnell identifizieren. Dies macht zum Beispiel bei einer größeren Gruppe von Studierenden Sinn, bei der jede Person an einem eigenen Objekt arbeitet.

purrfelinus alpha – Von der Konzepskizze zum Prototypen

Ein Beispiel für eine mögliche Anwendung ist **purrfelinus alpha** [1] – ein auf kreative Weise gestaltetes Produkt auf Basis eines *Raspberry Pi*. Bei diesem medialen Produkt kann ein Laserpointer über ein Webinterface gesteuert werden, um die zuhause gebliebene Katze per remote-Zugriff in der Mittagspause beschäftigen zu können. Der Name *purrfelinus* setzt sich zusammen aus dem Englischen Begriff *purr* (schnurren) und dem Lateinischen *felinus* (“katzenartig” oder “von Katzen”).



⇒

⇒


Der Laserstrahl im rechten oberen Bild wurde in Photoshop eingefügt.

Benutzer*innen können auf einer Webseite einen Kamerastream in Echtzeit anschauen. Über die geöffnete Webseite kann man dann über ein Webinterface mit seiner Katze interagieren. Dabei wird ein Schieberegler verwendet, der die Bewegung eines Servomotors steuert, der mit einem Laserpointer verbunden ist und diesen bewegt. Im Kamerabild sieht man den Laserpointer und die Katze.

Das Konzept ist inspiriert durch folgendes Katzenvideo. Die im Rendering verwendete Katze entstammt Thingiverse (Autor: BenitoSanduchi, 2012) und ist unter Creative Commons Lizenz publiziert.

Das Objekt *purrfelinus alpha* beinhaltet die folgenden Hardware Komponenten:

- Raspberry Pi 4 [2] (Model B Rev 1.1) mit Pi OS (Bullseye, 32-BIT)
- Raspberry Pi 4 Kamera Modul (Rev. 1.3)[3]

- Servomotor SG 90[4)]
- 2x Netzteil 5.1V, 3A[5)]
- E27-Gewinde (ausgebaut aus einer LED-Birne)
- 3D gedruckte Bauteile (siehe Dokumentation weiter unten)

Neben den entsprechenden Treibern und notwendigen Updates kommen folgende Software Module in *purrfelinus alpha* zum Einsatz:

- **Flask** – Ein modulares Web-Framework für Python, das für die schnelle Entwicklung von Webanwendungen und APIs eingesetzt wird.[6)] In der Anwendung stellt Flask das Backend bereit, um die Steuerung des Winkels des eingebauten Servomotors mittels eines runden Schiebereglers (round slider) in der Webanwendung “purrfelinus alpha” zu ermöglichen. Es rendert die Hauptseite, verwaltet den Echtzeit-Video-Feed und verarbeitet die vom Schieberegler gesendeten Winkeländerungen, um den Servomotor entsprechend anzusteuern.
- **roundSlider** – Ein halbkreisförmiger Schieberegler wird als UI-Element verwendet, um den Winkel des eingebauten Servomotors zu steuern. Der Schieberegler wird mit der jQuery roundSlider-Bibliothek implementiert und ermöglicht es Benutzer*innen, einen Wert aus einem Bereich auszuwählen, indem man einen runden Knopf um eine zentrale Achse dreht. Dabei werden der aktuelle Winkelwert an den Server gesendet und die Sichtbarkeit des Winkeltexts durch Drücken der “h”-Taste ein- und ausgeschaltet.
  Durch Anpassung der Übergangseigenschaften und der CSS-Klassen wurde eine Animation des Schiebereglers integriert. Die Animation verwendet eine benutzerdefinierte kubische Bezier-Kurve (cubic-bezier(1.000, -0.530, 0.405, 1.425)) als Übergangstaktgeberfunktion, um eine Bewegungsdynamik zu erzielen. Dadurch ergibt sich ein weicheres und ansprechenderes Verhalten des Schiebereglers beim Ändern des Wertes durch die Benutzer*innen – der Schieberegler verhält sich passend zum Bewegungsablauf bei einem Katz-/Maus-Spiel. Stichwort: Verhaltens-Attribution. Siehe auch folgendes (YouTube-Video).

Stromversorgung via E27 Gewinde und modifiziertes 5.1V Netzteil

Bitte unbedingt folgendes beachten: Die auf dieser Seite aufgeführte Dokumentation ist keine Anleitung zum Nachbau und dient lediglich dem Zwecke einer Präsentation von Felix Beck am 17. April an der FH Münster.

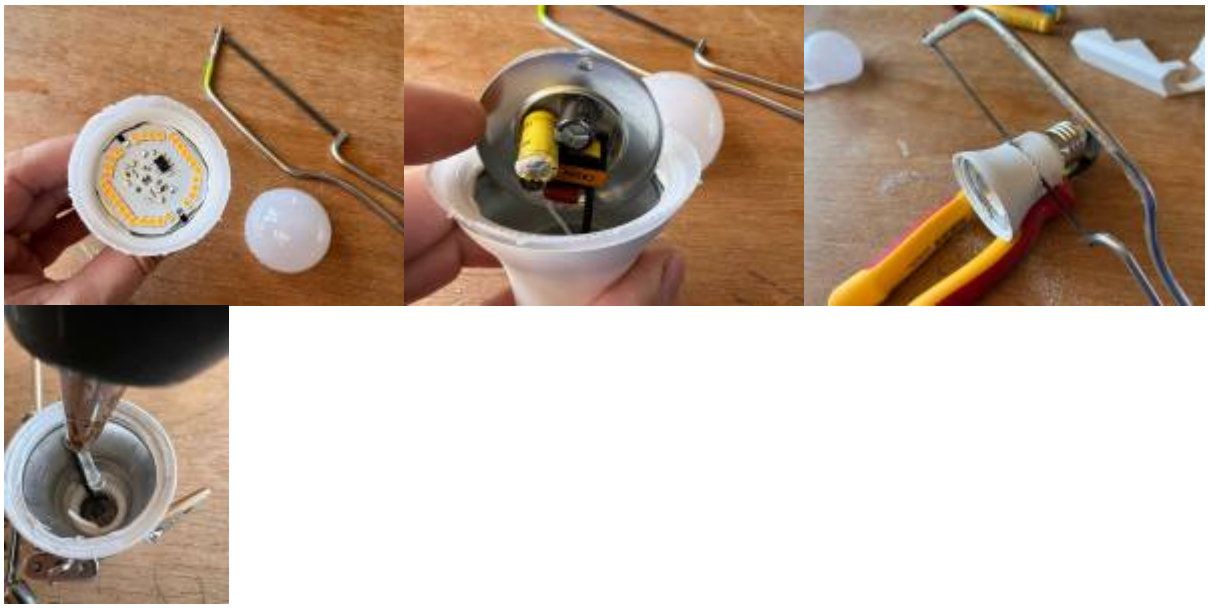
Das Arbeiten mit 230V Strompotenzial kann äußerst gefährlich sein und darf nur von geschultem Personal durchgeführt werden. Fehlerhafte oder unsachgemäße Verkabelung, unzureichende Schutzmaßnahmen oder Missachtung von Sicherheitsvorkehrungen können zu schweren Verletzungen oder sogar zum Tod führen.

Wenn Sie ein Projekt mit hohen Stromspannungen planen, sollten Sie sich immer an erfahrene Fachleute wenden und immer geeignete Sicherheitsvorkehrungen und Schutzmaßnahmen treffen, um ein sicheres Arbeitsumfeld zu gewährleisten. Suchen Sie entsprechende Labore auf. Bitte denken Sie daran, dass Ihre Gesundheit und Sicherheit immer Vorrang haben sollten, und seien Sie immer vorsichtig, wenn Sie mit elektrischen Komponenten arbeiten.

Beim Bau des Prototypen wurde eine E27-Fassung als Schnittstelle zur Stromzufuhr verwendet. Diese wurde aus einer defekten LED-Glühbirne ausgebaut. Es wurde sichergestellt, das das Leuchtmittel

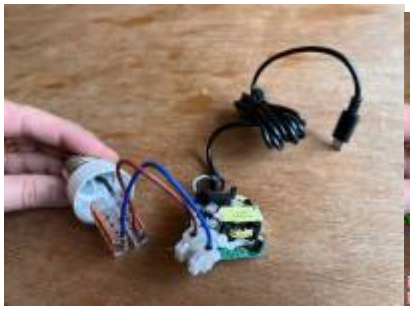
vollständig ausgeschaltet – vom Stromkreis getrennt – und abgekühlt war. Es wurde darauf geachtet, dass keine Glasbirne verwendet wurde, sondern eine LED-Birne mit Plastikgehäuse. Um Verletzungen durch scharfe Kanten oder elektrische Teile zu vermeiden, wurde eine Schutzbrille und Schutzhandschuhe getragen.

Als erstes wurden alle Teile der LED-Birne, die nicht am Gewinde befestigt waren, einschließlich der elektrischen Komponenten und der Platine entfernt. Dafür wurde in einem ersten Schritt der transluzent/weiße Teil mit einer Metallsäge angesägt und konnte dann mit einer Drahtschere abgeschnitten werden. Mit einer Zange konnte die LED-Platine vorsichtig gelöst und herausgezogen werden. Im unteren Foto sieht man zwei Drähte, die mit 230 Volt Wechselspannung betrieben werden. Der Draht, der zum Gewinde der Fassung führt, wird als *Außenleiter* oder *Phase* bezeichnet und ist schwarz (manchmal auch braun). Der andere Draht, der zur Spitze der Glühbirne führt, wird als *Neutralleiter* bezeichnet und ist normalerweise blau (in diesem Fall weiß). Es wurde darauf geachtet die beiden Drähte im inneren des Gewindes nicht abzubrechen oder die Fassung zu beschädigen. Nachdem die Fassung freigelegt war und sichergestellt wurde, dass Außen- und Neutralleiter noch fest verbunden waren, wurde der Innenraum mit Heißkleber aufgefüllt.



Die abgeschnittenen Plastikteile und die defekte LED-Platine, die im Projekt keine Verwendung finden, wurden gemäß den örtlichen Entsorgungsvorschriften entsorgt.

Außen- und Neutralleiter werden über eine Dosen- und eine Lüsterklemme mit einem kleinen USB-C Netzteil (5.1V, 3A) verbunden. An dieser Stelle muss nochmals auf obige Warnung zur Verwendung von 230V hingewiesen werden: Das Arbeiten mit elektrischen Teilen kann gefährlich sein. Man sollte immer vorsichtig und achtsam sein, um Verletzungen zu vermeiden!

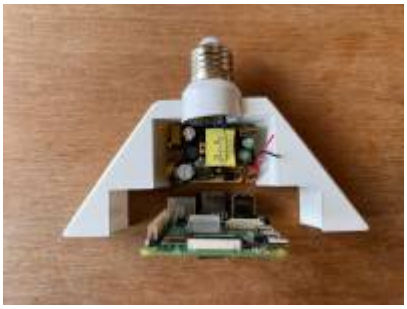


Entwicklung des Gehäuses (CAD, CAM)

Die in den unteren Bildern dargestellten und herunter-ladbaren *.stl-Dateien sind mit der webbasierten Software TinkerCAD erstellt worden. Es handelt sich dabei um ein einfach zu bedienendes und im Browser laufendes CAD-Programm, das von *Autodesk* entwickelt wurde und insbesondere für Anfänger geeignet ist, die 3D-Modelle erstellen möchten. Benutzer*innen können vorgefertigte geometrische Formen kombinieren, skalieren und manipulieren, um schnell und einfach 3D-Modelle zu erstellen. Es gibt leicht zugängliche Funktionen bspw. zur Generierung von Text. Auch gibt es die Möglichkeit Dateien aus anderen Quellen zu importieren. Erstellte 3D-Modelle können in verschiedene Dateiformate, wie z.B. OBJ oder STL exportiert werden, die mit anderen Anwendungen und 3D-Druckern kompatibel sind.

Die nachfolgenden Iterationen geben einen Überblick zu den einzelnen Entwicklungsschritten, die es brauchte den Hardware-Prototypen zu bauen und die Elektronik-Bauteile zu integrieren.

Iteration 1

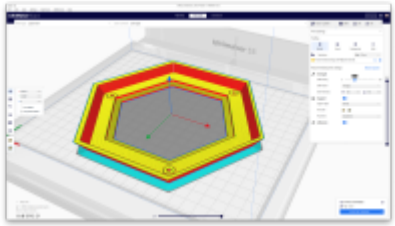
- Download Case – V01 (STL-file)
- Download Section – V01 (STL-file)

Iteration 2

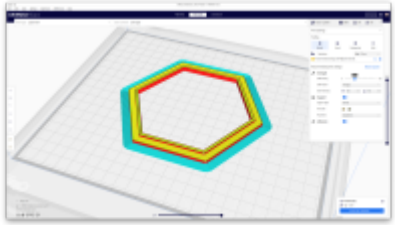
- Download Case – V02 (STL-file)

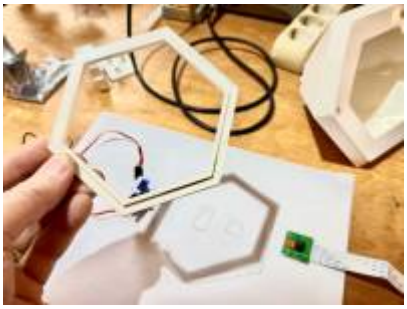
Iteration 3

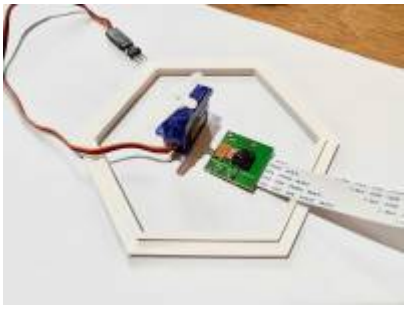
- Download Case – V03 (STL-file)



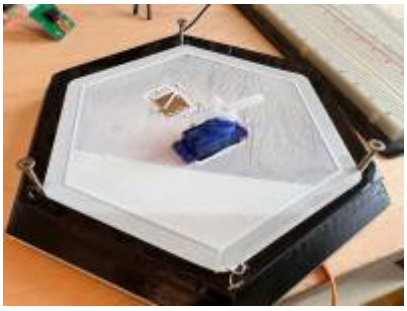
- Download Cover part A – V01 (STL-file)
- Download Cover part A – V02 (STL-file)

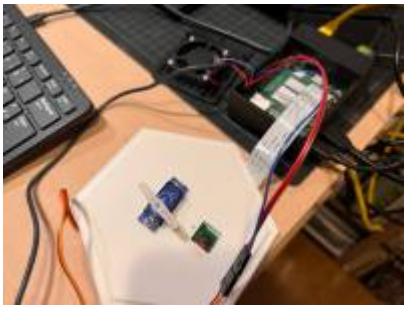




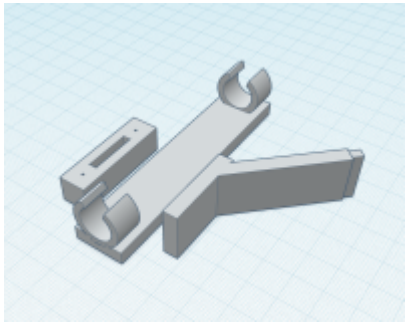


- Download Cover part B – V01 (STL-file)





- Download Cover part C – V01 (STL-file)



- Download Laser Pen Holder – V01 (STL-file)
- Download Laser Pen Holder – V02 (STL-file)
- Download Laser Pointer Switch – V01 (STL-file)

Integration von Hardware

Realisierung Software Sketch

Benutzer*innen können auf einer Webseite einen Kamerastream in Echtzeit anschauen. Auf der geöffneten Webseite soll man über das Webinterface mit seiner Katze interagieren können. Dabei wird ein Schieberegler verwendet, der die Bewegung eines Servomotors steuert, an dem ein Laserpointer befestigt ist. Im live-Bild der Kamera soll man letztendlich den Laserpointer-Punkt und die Katze sehen.

In den unteren Screenshots sieht man die Ausrichtung des Laserpointers passend zum eingestellten Winkel. In beiden Screenshots jeweils links ein Winkel von 48°, in beiden Bildern jeweils rechts ein Winkel von 135°.

Die Grundidee ist also die folgende:

1. Benutzer*in öffnet die Webseite mit Kamerastream und UI-Element (Drehregler).
2. Benutzer*in ändert den Wert des runden Schiebereglers.
3. Die Webseite sendet den neuen Winkelwert an den Server.
4. Der Server empfängt den Winkelwert und steuert den Servomotor entsprechend.

Datenfluss-Diagramm

Obiges Datenfluss-Diagramm gibt einen Überblick, wie die verschiedenen Teile der Anwendung zusammenarbeiten, um die Steuerung des Servomotors über den runden Schieberegler in der Webanwendung zu ermöglichen.

Schrittweise Erklärung des Codes

Ähnlich wie bei der Hardware-Entwicklung auch, hat es einige Iterationen gebraucht, um einen akzeptablen Stand zum Laufen zu bringen. Bei der Entwicklung hat es geholfen, die einzelnen Komponenten für sich zu bearbeiten. So wurde erst mit dem Servo getestet, später mit dem Einzelnen Bestandteilen des Interfaces, etc. Das Gesamtpaket wurde in kleinere Aufgaben zerteilt, von der jedes Teil für sich alleine getestet werden konnte. Erst später wurden die einzelnen Bestandteile mit Hilfe von ChatGPT zusammengeführt. Nachfolgend die Beschreibung des Python Codes:

1. Importieren der erforderlichen Bibliotheken: Flask, pigpio, cv2 und time.
2. Definieren des Servo-Pins, der an GPIO-Pin 18 angeschlossen ist.
3. Erstellen einer pigpio-Instanz und Setzen des Servo-Pins auf output.
4. Erstellen einer Flask-App-Instanz.
5. Initialisieren der globalen Variable current_angle auf 0, um den aktuellen Winkel des Servomotors zu speichern.
6. Definition der gen_frames()-Funktion, die als Generator zum Streamen der Videoframes dient. Die Kamera wird geöffnet und das Bild wird in jedem Frame skaliert, bevor es als JPEG kodiert und an den Client gesendet wird.
7. Definition einer Route /video_feed, die den Videostream bereitstellt.
8. Definition der Haupt-Route /, die eine index.html-Datei aus dem Vorlagenverzeichnis rendert.
9. Definition der angle_to_pulse_width()-Funktion, die einen Winkel in eine Pulsbreite für den Servomotor umrechnet.
10. Definition der set_servo_angle_slowly()-Funktion, die den Servomotor schrittweise und langsam auf den gewünschten Winkel dreht.
11. Definition einer Route /set_angle, die den Servo-Winkel einstellt, wenn sie über eine POST-Anfrage aufgerufen wird.
12. Ausführen der Flask-App und anschließendes Beenden des Servomotors und der pigpio-Instanz, wenn die Anwendung beendet wird.

Python Code

Wie im vorherigen Absatz beschrieben, wird im unten gezeigten Code eine Webanwendung mit Flask erstellt, die ein Live-Video-Streaming von einer angeschlossenen Kamera zeigt und die Möglichkeit bietet, einen Servomotor über einen Web-Controller zu steuern.

++++ Python Quell-Code ein-/ausklappen|

```
# Nicht vergessen "sudo pigpiod" auszuführen!
```

```
from flask import Flask, render_template, request, Response
import pigpio
import cv2
import time

servo_pin = 18

# Erstellen der pigpio-Instanz
pi = pigpio.pi()

# Setze den Modus des Servo-Pins auf OUTPUT
pi.set_mode(servo_pin, pigpio.OUTPUT)

app = Flask(__name__)

current_angle = 0  # Speichern des aktuellen Servo-Winkels

# Generator-Funktion zum Streamen der Videoframes
def gen_frames():
    camera = cv2.VideoCapture(0)
    scale_percent = 30
    first_frame = True
    while True:
        success, frame = camera.read()
        if not success:
            break
        else:
            # Skalieren des Frames
            width = int(frame.shape[1] * scale_percent / 100)
            height = int(frame.shape[0] * scale_percent / 100)
            dim = (width, height)
            resized_frame = cv2.resize(frame, dim,
interpolation=cv2.INTER_AREA)

            # Frame-Auflösung beim ersten Frame ausgeben
            if first_frame:
                print(f"Frame-Auflösung: {height}px x {width}px")
                first_frame = False

            # Kodiere den Frame als jpg
            ret, buffer = cv2.imencode('.jpg', resized_frame)
            frame = buffer.tobytes()
            yield (b'--frame\r\n'
                   b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

# Route für den Videostream
@app.route('/video_feed')
def video_feed():
    return Response(gen_frames(),
                    mimetype='multipart/x-mixed-replace; boundary=frame')
```

```
# Haupt-Route
@app.route("/")
def index():
    return render_template("index.html")

# Funktion zum Umrechnen von Winkeln in Pulsbreite
def angle_to_pulse_width(angle):
    min_pulse_width = 600
    max_pulse_width = 2300
    pulse_width_range = max_pulse_width - min_pulse_width
    return int(min_pulse_width + (angle / 180.0) * pulse_width_range)

# Funktion, um den Servo schrittweise zu bewegen (Schritte und
Geschwindigkeit)
def set_servo_angle_slowly(servo_pin, target_angle, step_size=1,
step_delay=0.001):
    global current_angle

    while current_angle != target_angle:
        if current_angle <target_angle:
            current_angle = min(current_angle + step_size, target_angle)
        else:
            current_angle = max(current_angle - step_size, target_angle)

        pulse_width = angle_to_pulse_width(current_angle)
        pi.set_servo_pulsewidth(servo_pin, pulse_width)
        time.sleep(step_delay)

# Route zum Einstellen des Servo-Winkels
@app.route("/set_angle", methods=["POST"])
def set_angle():
    angle = int(request.form["angle"])
    set_servo_angle_slowly(servo_pin, angle)
    return "OK", 200

# Hauptprogramm
if __name__ == "__main__":
    try:
        app.run(host="0.0.0.0", port=5000)
    finally:
        # Servo-Impuls aus und stop und Tschuess!
        pi.set_servo_pulsewidth(servo_pin, 0)
        pi.stop()
```

++++

html Code

++++ html Quell-Code ein-/ausklappen|

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PurrFelinus Alpha</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/jquery.roundslider/1.3/roundslider.min.css">
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/jquery.roundslider/1.3/roundslider.min.js"></s
cript>
    <style>
        body {
            margin: 0;
            padding: 0;
            overflow: hidden;
        }
        img {
            transform: rotate(270deg);
            position: absolute;
            top: 50%;
            left: 50%;
            min-height: 100%;
            min-width: 100vh; /* Verwenden Sie die Viewport-Höhe, um die
Breite anzupassen */
            object-fit: cover;
            object-position: bottom;
            z-index: -1;
            transform-origin: 50% 50%;
            transform: translate(-50%, -50%) rotate(270deg);
        }

        #slider-container {
            position: absolute;
            bottom: 0;
            left: 50%;
            transform: translateX(-50%);
            z-index: 1;
        }

        .rs-control .rs-range-color {
            background-color: rgba(0, 0, 0, 0);
        }
        .rs-control .rs-path-color {
            background-color: rgba(0, 0, 0, 0);
        }
        .rs-control .rs-handle {
            background-color:#57ff51;
        }
        .rs-control .rs-bg-color {
```

```
            background-color: rgba(0, 0, 0, 0);
        }

        #slider.rs-animation .rs-transition  {
            transition-timing-function: cubic-bezier(1.000, -0.530, 0.405,
1.425);
        }

        h1 {
            font-size: 3em;
            font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
            font-weight: 10;
            color: white;
            letter-spacing: 6px;
            text-align: center;
            text-shadow: 0 0 8px rgba(0, 0, 0, 0.5);
            margin-top: 135px; /* Vertikale Position der Überschrift */

        }
        p {
            margin-left: 20px;
            margin-top: 0;
        }

        .winkel {
            color: white;
            font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
        }

        .winkel .bold { /* Neue Regel, um die Klasse .bold hinzuzufügen */
            font-weight: bold;
        }

    </style>
</head>
<body>
    <img src="{{ url_for('video_feed') }}">
    <h1>purrfelinus alpha</h1>
    <div id="slider-container">
        <p class="winkel" id="angle-text" style="display: none;">winkel
servo motor: <span class="bold" id="angle">90</span><span
class="bold">°</span></p>
        <div id="slider" class="rs-animation" data-
animation="animation1"></div>
    </div>
    <script>
        $("#slider").roundSlider({
            circleShape: "half-top",
            radius: 320,
            width: 10,
```

```
            min: 0,
            value: 90,
            max: 180,
            step: 1,

            handleShape: "dot",
            handleSize: "+36",

            showTooltip: true,
            editableTooltip: true,
            mouseScrollAction: true,

            sliderType: "min-range", // Fügen Sie die gewünschte Option
hinzu

            // Fügen Sie die Klasse "rs-animation" hinzu, um die Animation
zu aktivieren
            cssClass: "rs-animation",

            change: function (e) {
                var angle = e.value;
                $("#angle").text(angle);

                $.post("/set_angle", { angle: angle }).done(function (data)
{
                    console.log("Erfolg");
                }).fail(function () {
                    console.log("Fehler");
                });
            }
        });

        $(document).on("keydown", function(event) {
            if (event.key === "h") {
                $("#angle-text").toggle();
            }
        });
    </script>

</body>
</html>
```

++++

Nächste Schritte & "Nice to Have"

Die aktuelle Version könnte an vielen Punkten weiter ausgearbeitet werden. Nachfolgend einige

Überlegungen zu nächsten Schritten:

- ~~⇒ **anstelle von RPi.GPIO library die pigpio library** verwenden (siehe auch hier). * D<del>er Slider verschiebt sich um einige Pixel nach oben, wenn der Anfasser auf den Min bzw Max Wert gesetzt wird.~~
- Servospeed anpassen ⇒ weichere Bewegungsabläufe
- Die Kamerabild-Einstellungen optimieren: Farben entsättigen (evtl. nur Graustufen) und grünen laser-Punkt deutlicher hervorheben.
- Die Kamera physikalisch um 90° rotieren damit das Videobild besser zur aspect-ratio des Browserfensters passt und die Benutzer*innen mehr Fläche zum Spielen zu Verfügung haben.
- Autostart von *purrfelinus alpha*-Programm: Die Anwendung soll automatisiert starten sobald das Pi Strom bekommt.
- Integration von Status-LEDs: Anzeige von Zuständen, wie bspw. WiFi-Netzwerk erreichbar, Program XY läuft, etc.
- Das WiFi-Signal erscheint schlechter, wenn das Raspberry Pi im Gehäuse verbaut ist. Es müßte getestet werden, ob die Geometrie des Gehäuses hier einen Einfluss hat.
- Automatisierte Bilderkennung: Die Position der Katze könnte erkannt werden und der Laser automatisiert immer in die gegenüberliegende Richtung steuern.

Weiterführende Literatur

- A Touch of Code, Interactive Installations and Experiences, Robert Klangen, Sven Ehmann, Verena Hanschke, Berlin, Gestalten, 2011
- Coding Languages for Absolute Beginners, Zach Webber, 2018
- Creative Code, Aesthetic und Programmierung am MIT Media Lab, John Magda, Birkhäuser, Basel, 2004
- Design und künstliche Intelligenz, Theoretische und praktische Grundlagen der Gestaltung mit maschinell lernenden Systemen, Marc Engelhaft, Sebastian Löwe, Birkhäuser, Basel, 2022
- Designing Interactions, Bill Moggridge, MIT Press, Cambridge, 2007
- HelloWorld – The Big Book of Computing Content, Raspberry Pi Foundation, 2022 (PDF download link)
- Interaktive Systeme, Band 1, Grundlagen, Graphical User Interfaces, Informationsvisualisierung, Bernhard Reim, Raimund Dachselt, Springer-Verlag Berlin Heidelberg, 2010
- MAKE: Getting started with Sensors, Measure the World with Electronics, Arduino, and Raspberry Pie, Kimmo Karvinen, Tero Karvinnen, Maker Media, San Francisco, 2016
- Raspberry Pi* – one Vorkenntnisse, Benjamin Saphir, PBD Verlag, Östringen, 2021
- The Manga Guide to Electricity, Kazuhiro Fujitaki, No Starch Press, San Francisco, 2009
- Zukünfte gestalten – Spekulation, Kritik, Innovation, Benedikt Groß, Eileen Mandir, Verlag Hermann Schmidt, Mainz, 2022

- siehe auch Top 100 Bücherliste

Fazit

Es handelt sich bei dieser Idee nur um einen konzeptionellen Ansatz, mit dem Ziel unterschiedliche Einsatzmöglichkeiten des Moduls zu verdeutlichen. Ich gehe davon aus, dass ein Laser, der Katzen in

die Augen schießt, grundsätzlich schädlich ist. Aus diesem Grunde sollte das Projekt nicht realisiert werden.

— *Felix Hardmood Beck 2023/04/17 16:00*

1)

2)

Zu beziehen bspw. von Amazon

3)

Zu beziehen bspw. von Amazon

4)

(Zu beziehen bspw. von Amazon

5)

(Zu beziehen bspw. von Amazon

6)

siehe auch folgendes Tutorial: https://www.youtube.com/watch?v=6jDyr0ydBgU