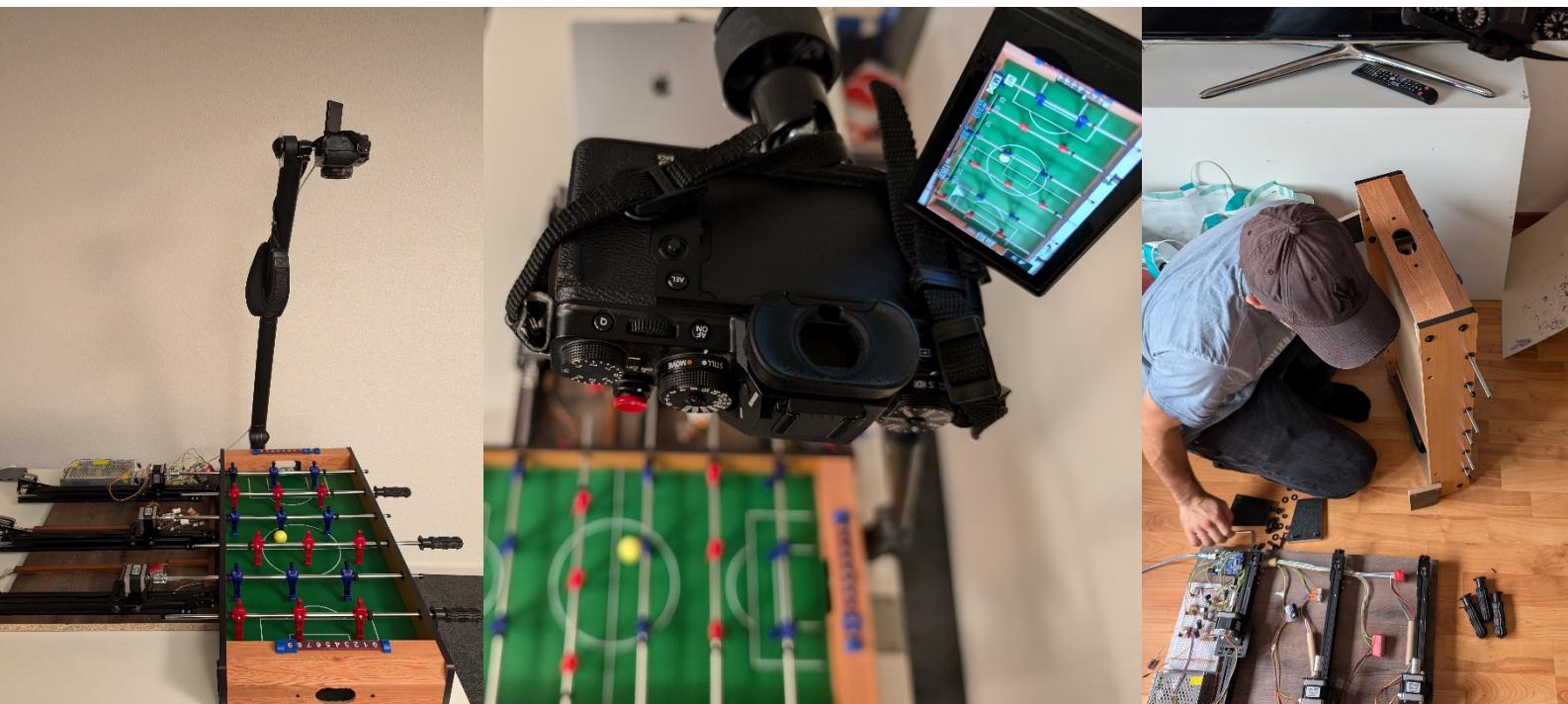




Automatisierter Tischkicker mit Computer Vision

Berufsmaturitätsarbeit zum Thema «Bewegung»

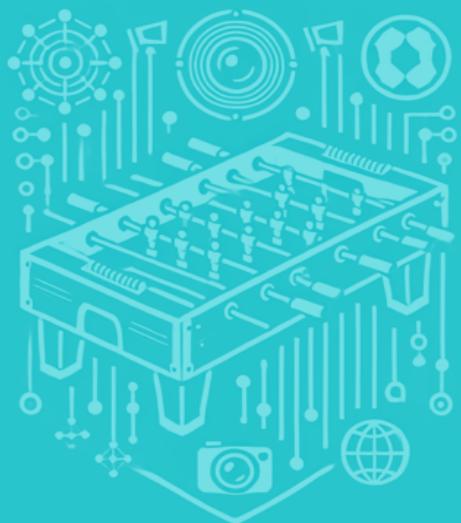


Abgabe am 29. November 2024

Klasse: BGB21C

Verfasst von: Timoniel Perez Vargas, Sy Viet Dao

Betreuende Lehrperson: Niko Van Wyk



Abstract

In dieser Berufsmaturitätsarbeit wird folgende Fragestellung behandelt: Kann man mit einer herkömmlichen Kamera, Computer Vision Technologie und einer motorisierten Steuerung ein System entwickeln, dass bei einem Budget von unter 300 CHF die Spielfiguren eines Tischkickers in Echtzeit so steuert, damit es mit Freizeitspielern auf Augenhöhe konkurrieren kann?

Der Hauptteil der Arbeit umfasst die Entwicklung und Konstruktion eines eigenen Tischkicker System. Zudem wird die Effektivität der Bildverarbeitungsmethoden untersucht und die mechanischen Komponenten miteinander verglichen. Abgeschlossen wird das Ganze mit einem Feldtest, der die Hypothese und die Leitfrage der Arbeit beantwortet.

Durch die Analyse stellt sich heraus, dass es möglich ist, ein System zu entwickeln, das einen Tischkicker in Echtzeit steuert. Der durchgeführte Feldtest falsifiziert jedoch die Hypothese, dass das System einem Freizeitspieler ein ebenbürtiger Gegner ist. Aus 48 analysierten Spielen gewinnt der autonome Tischkicker in etwas über 4 Prozent der Fälle. Die durchschnittlich pro Spiel erzielten Treffer belaufen sich auf zwei Tore.

Dank

Wir möchten Riccardo Russo unseren herzlichen Dank aussprechen für seine wertvolle Unterstützung bei der Problemlösung im Bereich des Programmiercodes. Sowie bei der Analyse und Behebung technischer Schwierigkeiten. Besonders schätzen wir, dass er uns während seiner Arbeitszeit Zeit für die Arbeit und Feldtests eingeräumt hat. Ohne seine Hilfe hätten wir den Tischkicker nicht so gut funktionstüchtig und rechtzeitig fertiggestellt.

Unser Dank gilt auch Manuel Weiss für das aufschlussreiche Interview zu Maschinelles Lernen, insbesondere zur YOLO-Technologie und Computer Vision. Sein Beitrag hat wesentlich dazu beigetragen, unser Verständnis für maschinelles Lernen zu vertiefen.

Wir möchten auch unserem Klassenlehrer Niko Van Why danken, der uns bei Fragen zu mathematischen Themen und dem Umfang des schriftlichen Teils unterstützt hat. Zudem war er stets erreichbar, um uns bei weiteren Fragen und Problemen zu helfen.

Wir danken Mattia Rüegg, Sy Viets Bruder, für das Ausleihen einer qualitativ hochwertigen Kamera. Ausserdem danken wir ihm für die ausführliche Zeit, die er sich genommen hat, um uns die Kameraeinstellungen zu erklären und wertvolle Tipps für besseres Tracking zu geben.

Wir möchten uns bei allen bedanken, die mit ihren kleinen Beiträgen – sei es durch Unterstützung von Familie, Freunden oder Arbeitskollegen – zum erfolgreichen Abschluss dieses Projekts beigetragen haben. Ein besonderer Dank gilt jenen, die uns mit wertvollen Ratschlägen und Tipps begleitet haben. Ihre Bereitschaft, ihr Wissen und ihre Zeit mit uns zu teilen, war für uns von unschätzbarem Wert.

Inhaltsverzeichnis

1	Einleitung	1
2	Grundlage	2
2.1	Computer Vision.....	2
2.1.1	Einführung Yolo Modell	3
2.1.2	Yolo Alltag Verwendung	4
2.2	Kamera Setup	5
2.2.1	Bild Qualität.....	5
2.2.2	Motion Blur.....	5
2.2.3	Belichtungsdreieck	6
2.3	Motorenauswahl für den Tischkicker	7
2.3.1	Nutzwertanalyse Motoren	7
2.3.2	Ergebnisse der Motorenbewertung.....	8
2.4	Eigenschaften NEMA 17 Schrittmotor	9
2.5	Ähnliche Projekte.....	10
2.5.1	Bosch	10
2.5.2	EPFL	10
2.5.3	From Scratch.....	11
2.6	Unterschied zu diesem Tischkicker	11
3	Planung.....	12
3.1	Konstruktion Konzept.....	12
3.2	Regelkreissystem Konzept	13
3.3	Kamera Auswahl.....	14
3.4	Analyse der FPS-Kompatibilität.....	15
3.4.1	Wahl des YOLOv1 Nano Modells	15
3.5	Nutzwertanalyse zur Auswahl der Technik	16
3.6	Sankey Diagram.....	17
3.6.1	Vergleich der Kostenanteile	17
4	Umsetzung	18
4.1	Konstruktion	18
4.2	Verdrahtung	19
4.2.1	Fehlerquellen bei der Verdrahtung	20
4.3	Maschinelles Lernen.....	21
4.3.1	Bilder Labeln	21
4.3.2	Training eines YOLOv1 Nano Modells	22
4.3.3	Wichtige Erkenntnis.....	23

4.4	Aufbau des Bilderkennungssystems	24
4.4.1	Spielfeld-Erkennung	24
4.4.2	Ball-Erkennung	24
4.5	Optimierung durch Ballbewegungsvorhersage.....	25
4.5.1	Funktionsweise der Richtungsvektor-Methode.....	25
4.6	Motor Logik.....	26
5	Feldtest	27
5.1	Echtzeit Analyse Tischkicker	28
5.2	Menschliche Reaktionszeit.....	29
6	Schlusswort.....	30
7	Quellenverzeichnis.....	31
8	Abbildungsverzeichnis	33
9	Anhang.....	35
9.1	Glossar.....	35
9.2	Quellcode	37
9.3	Videos, Bilder, Feldtest	37

1 Einleitung

Unser Oberthema für die Berufsmaturitätsarbeit lautet «Bewegung» wir werden uns anhand einen automatisierten Tischfussballkasten mit Computer Vision mit dem Thema befassen. Das Ziel ist es, mithilfe moderner Technologie ein System zu entwickeln, bei welchem die Spielfiguren auf einer Seite schnell und präzise gesteuert werden. Die Herausforderung besteht darin, mit Motoren, Drivern, Kamera und Steuergeräten in Kombination mit Echtzeit-Computer-Vision den Ball auf dem Spielfeld zu verfolgen und die Figuren so positionieren, dass sie entweder den Ball abwehren oder gezielte Schüsse abgeben können.

Die zentrale Fragestellung ist: «Kann ein Computer-Vision-System mit einem geringen Budget von unter 300 Franken die Spielfiguren eines Tischkickers in Echtzeit steuern und mit Freizeitspielern auf Augenhöhe konkurrieren?» Unsere Hypothese lautet, dass die aktuellen Fortschritte in der Computer-Vision-Technologie dies ermöglichen. Wir gehen davon aus, dass unser System eine Reaktionszeit erreicht, die der menschlichen überlegen ist, und eine Gewinnchance von mindestens 50 % gegen jugendliche Spieler erreicht.

Ausgesucht haben wir unser Projektthema nicht nur Aufgrund unseres Interesses an moderner Technologie und Tischfussballspielen, sondern auch der Herausforderung wegen, die unsere Fähigkeiten in Informatik, Elektronik und Mechanik fordert. Ebenso um zu zeigen, wie vielseitig Computer-Vision-Technologie einsetzbar ist – eine Technologie, die in der Zukunft immer mehr an Bedeutung gewinnen wird.

Darüber hinaus bietet das Projekt die Möglichkeit, unser technisches Wissen zu erweitern und ein Produkt zu schaffen, auf das wir stolz sein können. Ein weiterer Grund zur Wahl war auch der Bezug zur Arbeitswelt: Im Team kombinieren wir Wissen eines Applikationsentwicklers und eines Elektroinstallateurs. Der Tischkicker dient dabei als spannende Herausforderung.

Wir behandeln dabei die Entwicklung und den Bau des Systems, die Implementation von Computer-Vision, sowie eine Testphase: Wir bestimmen die Reaktionszeit, Gewinnchancen und die erzielten durchschnittlichen Tore gegen Freizeitspieler.

2 Grundlage

In diesem Abschnitt werden die grundlegenden Konzepte erläutert, die für das Verständnis und die Umsetzung eines automatisierten Tischkickers erforderlich sind. Ziel ist es, die theoretischen Grundlagen sowie die praktischen Überlegungen nachvollziehbar darzustellen. Schwerpunkte bilden Themen wie **Computer Vision**, das optimale **Kamera-Setup** und die **Auswahl geeigneter Motoren**.

2.1 Computer Vision

Computer Vision ist ein Bereich der künstlichen Intelligenz, der es Computern ermöglicht, visuelle Informationen zu verarbeiten und zu interpretieren, ähnlich wie der menschliche Sehsinn. Mit Kameras und Sensoren wie LiDAR oder Wärmebildsensoren können Daten wie Farben, Formen, Tiefeninformationen und Temperaturen analysiert werden. Dies ermöglicht es, Muster zu erkennen, Objekte zu lokalisieren und Szenen zu verstehen. Anwendungen finden sich in Bereichen wie autonomem Fahren, medizinischer Bildanalyse und Überwachung. In den letzten Jahren gab es grosse Fortschritte bei der Objekterkennung und Bewegungsverfolgung in Echtzeitanwendungen. (Ultralytics, 2024).

2.1.1 Einführung Yolo Modell

YOLO („You Only Look Once“) ist ein Open-Source-KI-Modell, das für nahezu Echtzeitanwendungen in der Objekterkennung und -verfolgung optimiert wurde. Es zeichnet sich durch seine Effizienz aus, da es Bildinformationen in einem einzigen Durchlauf verarbeitet, wodurch Objekte schnell und präzise erkannt werden, können (Glenn 2024).

Stand 21.10.2024 ist Version 11 das stärkste und effizienteste Modell (Ultralytics, 2024).

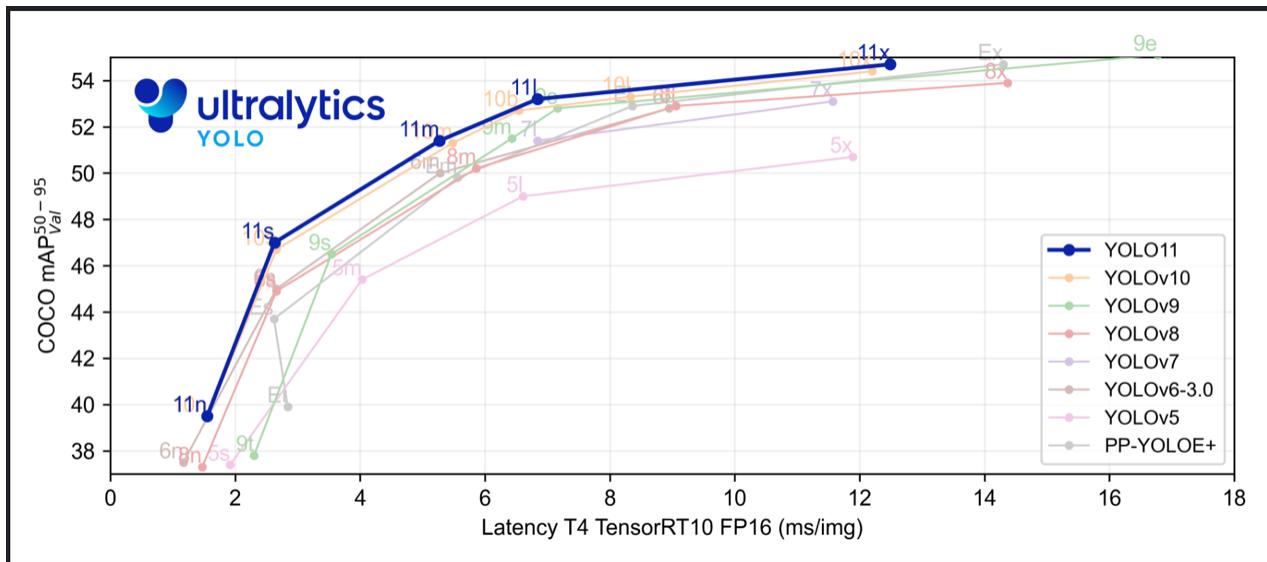


Abb. 1: zeigt eine Übersicht der YOLO-Modelle bis Version 11 und veranschaulicht die Leistung jedes Modells in Millisekunden bei einem spezifischen Prozess (Ultralytics, 2024).

Die beigefügte Grafik zeigt den Vergleich verschiedener YOLO-Versionen unter optimalen Hardware- und Softwarebedingungen. Auf der Y-Achse ist der mAP-Wert dargestellt, der angibt, wie gut ein Modell bei der Objekterkennung abschneidet, hier gemessen an COCO mAP 50–95. Die X-Achse zeigt die Latenz in Millisekunden pro Bild, also die Zeit, die ein Modell benötigt, um eine Inferenz durchzuführen (Ultralytics 2024).

2.1.2 Yolo Alltag Verwendung

YOLO wird in verschiedenen Bereichen eingesetzt, beispielsweise in der Überwachung zur Erkennung von Personen und deren Verweildauer (siehe Abb. 2). In der Forschung autonomer Fahrsysteme unterstützt YOLO die Identifikation von Fußgängern, Verkehrsschildern und anderen Fahrzeugen für eine sichere Navigation.

Weitere Einsatzmöglichkeiten umfassen



Abb. 2: YOLO-Erkennung von Menschen in Echtzeit.

die Objekterkennung in Haushaltsgeräten, die medizinische Diagnostik, automatisierte Produkterkennung an Kassen sowie die Analyse von Bewegungen in Fitness-Apps und die Echtzeit-Interaktion in VR-Anwendungen (Ultralytics 2024).

2.2 Kamera Setup

2.2.1 Bild Qualität

Für den Einsatz von YOLO-Modellen ist die Bildqualität der Kamera entscheidend. Eine hohe Auflösung ermöglicht detaillierte Aufnahmen und verbessert die Erkennung kleiner Objekte, wie etwa eines Balls. Allerdings kann eine zu hohe Auflösung die Verarbeitungsgeschwindigkeit beeinträchtigen. Da YOLO-Modelle jeden Pixel eines Bildes analysieren, führt eine grössere Pixelanzahl zu einem erhöhten Datenvolumen und längeren Rechenzeiten. Pixel, die kleinsten Bestandteile eines Bildes, enthalten Informationen zu Farbe und Helligkeit, die das neuronale Netz zur Mustererkennung nutzt. Daher ist es wichtig, ein ausgewogenes Verhältnis zwischen Auflösung und Leistung zu finden, um eine optimale Balance für die Anwendung sicherzustellen.

2.2.2 Motion Blur

Die Bildschärfe ist entscheidend für die Objekterkennung. Bewegungsunschärfe, sogenannter „Motion Blur“, durch schnelle Objektbewegungen erschwert die Identifikation und Lokalisierung erheblich, da Objekte verwischt dargestellt werden (siehe Abb. 3).

Je höher die Anzahl der Bilder pro Sekunde (FPS), desto geringer ist das Risiko von Motion Blur. In diesem Fall wurden 60 FPS verwendet, was jedoch oft nicht ausreicht, um schnelle Objektbewegungen scharf darzustellen.



Abb. 3: Ball mit Motion Blur, aufgenommen mit einer 60-fps Kamera ohne spezifische Einstellung.

Lösung: Kürzere Belichtungszeit

Eine kürzere Belichtungszeit reduziert Motion Blur, da weniger Zeit für die Bildaufnahme benötigt wird. Dies mindert jedoch die Lichtmenge auf dem Sensor, was die Bildqualität bei wenig Licht beeinträchtigen kann. Ein Ausgleich zwischen Belichtungszeit, Lichtausbeute und Bildrate ist daher essenziell.

2.2.3 Belichtungsdreieck

Hier kommt das Konzept des "Belichtungsdreieck" ins Spiel, welches die drei Variablen Belichtungszeit, Blende und ISO-Wert beschreibt. Siehe Abb. 4 zur Darstellung des Konzepts.

1. Belichtungszeit: Kürzere Belichtungszeiten minimieren Motion Blur, reduzieren aber auch die Lichtmenge.
2. Blende: Größere Blende (kleinere Zahl) lässt mehr Licht durch, reduziert jedoch die Schärfentiefe.
3. ISO-Wert: Höherer ISO-Wert erhöht die Lichtempfindlichkeit, kann jedoch Bildrauschen verursachen.

Beispielrechnung zur Optimierung:

Bei einer Bewegungsgeschwindigkeit von 10 m/s und einer Belichtungszeit von 1/500 Sekunde entsteht eine Unschärfe von 0,02 m. Eine Blende von f/2.8 und ein ISO-Wert von 800 können helfen, bei dieser Belichtungszeit ein gutes Bild zu erzielen. Dieses Konzept ist für das Projekt entscheidend und wird die Objekterkennung verbessern.

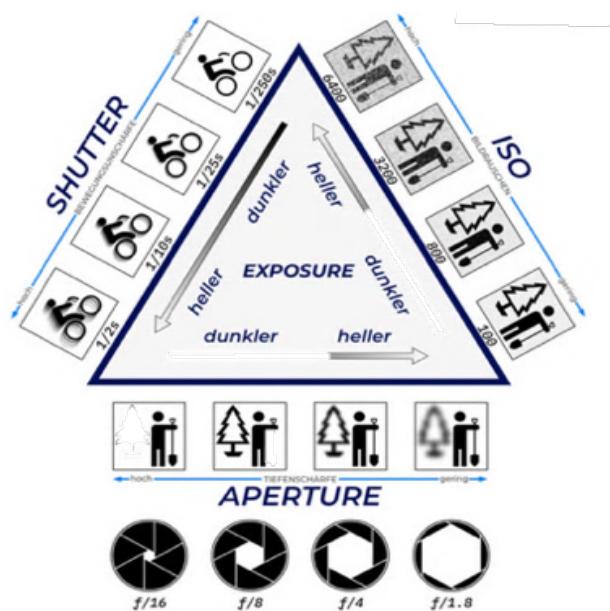


Abb. 4: Zeigt das Belichtungsdreieck (ertraeume-deine-welt, 2024)

2.3 Motorenauswahl für den Tischkicker

Für die Bewegung der Stangen eines Tischkickers sind Motoren notwendig. Jede Stange erfordert dabei zwei Motoren: einen für die **lineare Bewegung** und einen für die **Rotationsbewegung**. Um die optimalen Motoren für das Projekt zu finden, wurde eine ausführliche Recherche durchgeführt. Ziel war es, passenden Komponenten zu finden, die gut zusammenarbeiten und den automatisierten Tischkicker zuverlässig zum Laufen bringen. Angesichts der Vielzahl an verfügbaren Motoren, die sich in Preis und Anwendung unterscheiden, wird in diesem Abschnitt untersucht, welche Motoren am besten für das Projekt geeignet sind.

2.3.1 Nutzwertanalyse Motoren

In dieser Nutzwertanalyse werden fünf Motoren anhand der Kriterien Preis, Geschwindigkeit, Bedienung und Installationsaufwand verglichen. Jedes Kriterium wird gewichtet und jeder Motor erhält eine Bewertung. Der Motor mit der besten Gesamtbewertung wird als der geeignete für den Tischkicker ausgewählt.



Abb. 7: Zeigt einen modernen BLDC-Motor (Bodin, 2024).

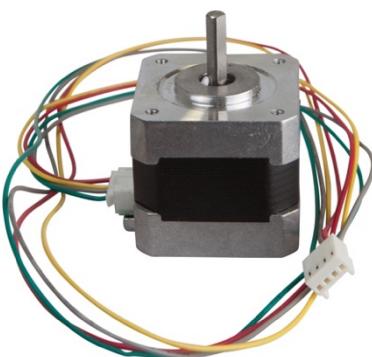


Abb. 6: NEMA 17 Schrittmotor (3DWare, 2024).



Abb. 5: NEMA 23 Schrittmotor (Bastelgarage, 2024).



Abb. 8: MG 996 Servomotor (Makerselectronics, 2024).



Abb. 9: Gleichstrommotor (Kem Industrie, 2024).

Die folgenden Kriterien – Preis, Geschwindigkeit, Bedienbarkeit und Zeitaufwand – wurden für die Analyse festgelegt und nach ihrer Bedeutung gewichtet. (Die Gewichtung basiert auf einer subjektiven Einschätzung.) Die Bewertung erfolgte anhand der PDF im Anhang zu den Motoren. In der Tabelle sind die Motoren gemäss der obigen Abbildung dargestellt.

Nutzenwertanalyse für die Motoren												
G = Gewichtung [%]	B = Bewertung	N = Nutzwert	Nema 17		Servomotor		Gleichstrom		Nema 23		BLDC-Motor	
			G	B	N	B	N	B	N	B	N	
Preis	30		4	1.2	2	0.6	4	1.2	2	0.6	2	0.6
Geschwindigkeit	20		3	0.6	5	1	3	0.6	4	0.8	4	0.8
Bedienbarkeit	30		4	1.2	4	1.2	3	0.9	4	1.2	3	0.9
Zeitaufwand	20		3	0.6	3	0.6	3	0.6	3	0.6	3	0.6
Total	100											
Resultat				3.6		3.4		3.3	3.2		2.9	
Rangfolge				1. Rang		2. Rang		3.Rang	4.Rang		5.Rang	

Die Bewertung erfolgt von 1-5 Punkten (1=sehr schlecht, 5= sehr gut)

2.3.2 Ergebnisse der Motorenbewertung

Aus der Recherche heraus konnte ein Gewinner bestimmt werden, der **Nema 17 Schrittmotor**. Das Balkendiagramm welches im Anhang ersichtlich ist veranschaulicht, welche Motoren die höchsten Punktzahlen mit Einbezug der Wertung in %, in den verschiedenen Bewertungskriterien, sowie insgesamt erzielt haben. Im Anhang sind Links zu Datenblätter der untersuchten Motoren sowie den Treibern vorhanden.

2.4 Eigenschaften NEMA 17 Schrittmotor

Schrittmotoren wie der NEMA 17 setzen Stromimpulse vom Steuergerät (siehe Abb. 10) in präzise Schritte um, die eine Rotation erzeugen. NEMA 17 bezeichnet Motoren mit einem Gehäusemass von 1,7 Zoll (ca. 4,3 cm) und einem Schrittwinkel von $1,8^\circ$, was 200 Schritte für eine volle Umdrehung bedeutet. Diese Motoren ziehen pro Phase etwa 0,25 bis 2 Ampere, wobei die beiden Phasen nie gleichzeitig den maximalen Strom ziehen. NEMA 17 Motoren sind präzise, kostengünstig und für die Anforderungen dieses Projekts ideal. Zur Steuerung wird der bewährte A4988-Treiber (siehe Abb. 10) verwendet, der eine sichere Kompatibilität mit den Komponenten gewährleistet. Ein 12 V, 10 A Netzteil sorgt für die nötige Stromversorgung, da der Arduino allein nicht ausreichend Strom liefern kann (Douglas 2024).



Abb. 10: A4988 Treiber

2.5 Ähnliche Projekte

Es gibt mehrere Projekte, die automatisierte Tischkicker realisiert haben. Im Folgenden werden drei davon kompakt dargestellt, um deren Ansätze sowie die Unterschiede zu diesem Projekt zu verdeutlichen.

2.5.1 Bosch

2017 entwickelte Bosch einen automatisierten Tischkicker mit Standard-Webkamera, leistungsfähigem Computer und grossen Gleichstrommotoren. Statt eine feste Spiellogik zu programmieren, verfolgte Bosch einen interessanten Ansatz: Sie trainierten ein neuronales Netz, das mithilfe eines Belohnungssystems eigenständig Strategien entwickelte (Krause 2017).

Das Projekt wurde jedoch 2019 eingestellt, da das Training aufgrund der enormen Datenanforderungen zu aufwendig war. Auch der Versuch, das Training mithilfe von Simulationen durchzuführen, scheiterte, da die Simulation die Realität nicht präzise genug widerspiegeln konnte (Alba 2022).



Abb. 11: Kick it like Bosch – Automatisierter Tischfussball von Bosch (Krause, 2017).

2.5.2 EPFL

2016 entwickelten Studierende der EPFL ein ähnliches Projekt. Sie verwendeten eine Hochgeschwindigkeitskamera (500 fps), die unter einem transparenten Plexiglas-Tisch montiert war, um den Ball anhand seiner Farbe in 2D zu verfolgen. Für die Bewegungen des Tischkickers setzten sie leistungsstarke Gleichstrommotoren ein, die sowohl lineare als auch rotatorische Bewegungen mit hoher Beschleunigung ermöglichten. Der Tisch war stabil konstruiert, und zusätzliche Sensoren erfassten die Position des gegnerischen Spielers. Diese Daten wurden genutzt, um die Spielstrategie dynamisch anzupassen und zu optimieren (Ximea 2020).

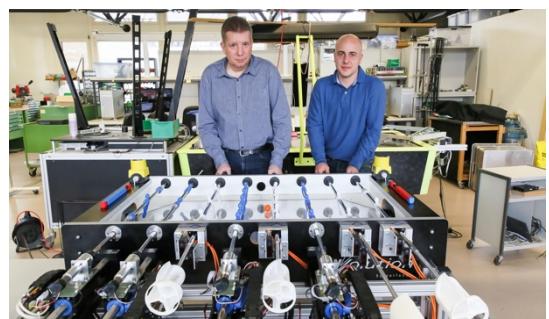


Abb. 12: EPFL-Student und Professor spielen gegen den Tischfussballtisch (Ximea, 2020).

2.5.3 From Scratch

2024 präsentierte Xander Naumenko auf seinem Kanal ein Video zu seinem automatisierten Tischkicker, in dem er 500 Stunden Entwicklungszeit investierte. Für die Steuerung setzte er auf hochwertige Schrittmotoren und nutzte fünf teure Infrarotkameras, um den Ball in 3D zu verfolgen, da Farberkennung allein für präzises Tracking unzureichend war. Er programmierte die Logik selbst und legte den Fokus auf Präzision sowie strategische Spielzüge (Naumenko 2024).

Nach einigen Testspielen zeigte sich jedoch, dass das System Schwächen hatte: Bei chaotischen Ballbewegungen reagierte es zu langsam, was menschliche Spieler ausnutzen konnten. Zusätzlich verblasste die Markierungsfarbe des Balls im Laufe des Spiels, was die Verfolgung erschwerte. Allerdings überzeugte das System bei Ballbesitz durch eine sehr präzise Steuerung und effektive Strategien (Bünte 2024).

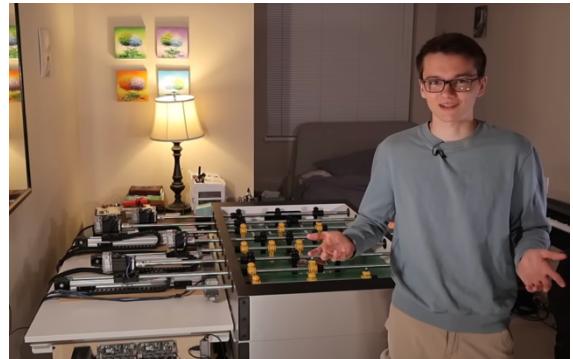


Abb. 13: Screenshot aus dem Video des YouTube Kanals „From Scratch“ (Naumenko, 2024).

2.6 Unterschied zu diesem Tischkicker

Die drei Projekte verfolgen unterschiedliche Ansätze: Bosch nutzt maschinelles Lernen, EPFL setzt auf Sensoren und 2D-Tracking, und „From Scratch“ verwendet eine Infrarotkamera mit 3D-Tracking. Alle haben jedoch Einschränkungen in Präzision und Strategie, insbesondere in unkontrollierten Spielsituationen, sowie hohe Kosten.

Der Ansatz unterscheidet sich durch den Fokus auf Effizienz und Kosteneffektivität. Mit einem begrenzten Budget wurde ein kompakter Prototyp entwickelt, der eine Standardkamera für die 2D-Ballerkennung nutzt. Die Bildverarbeitung erfolgt durch ein KI-Modell, das eine deutlich höhere Präzision als die klassische Farberkennung bietet (siehe Abb. 17 im Anhang, mit einem Confidence-Score von 96% in verschiedenen Szenarien). Für die Steuerung kommen vorprogrammierte Motoren zum Einsatz, ohne den Einsatz von maschinellem Lernen. Das Ziel ist es, ein kostengünstiges System zu entwickeln, das mit hochpreisigen Lösungen konkurrieren kann.

3 Planung

Im Abschnitt Planung, werden die Ideen für die Montage beziehungsweise die Auswahl der technischen Komponenten, sowie die Konzepte der Logik der Programmierung erläutert.

3.1 Konstruktion Konzept

Einer der ersten Gedanken war, den Linearmotor mit einer Gewindestange umzusetzen. Dabei sollte der Motor die Gewindestange drehen, die sich in ein auf einer Schiene fixiertes Wägelchen einschraubt. Dieses Wägelchen, dessen Drehung verhindert wird, wandelt die Rotationsbewegung in eine Linearbewegung um. In Abb. 14 sieht man einen solchen Motor.



Abb. 14. Zeigt einen linear ball Screw actuator (Daniel, Kowol und Sciuto 2024).



Abb. 15: Zeigt einen Linearmotor mit Riemenantrieb.

Das schliesslich gewählte Konzept ist mit einem Riemenantrieb ist in Abb. 15 zu sehen und bietet mehrere Vorteile. Hier bewegt eine am Motor befestigte Antriebsrolle einen Riemen, der über eine Umlenkrolle am Ende der Schiene geführt wird. Ein Schlitten, der fest mit dem Riemen verbunden ist, wird so entlang der Schiene bewegt. Dieses System ist nicht nur kostengünstiger als das Ball-Screw-System, sondern auch deutlich einfacher zu konstruieren.

3.2 Regelkreissystem Konzept

Dies ist ein geschlossenes Regelkreissystem, bestehend aus mehreren Komponenten, wie in Abb. 16 dargestellt. Die Kamera erkennt den Ball sowie das Spielfeld. Der erfasste Frame wird an die Software gesendet, wo mithilfe des YOLO-Modells der Ball und relevante Objekte erkannt werden. Weitere Berechnungen erfolgen, wie beispielsweise die Umwandlung von Pixelkoordinaten in reale Masseneinheiten. Diese Daten werden über USB-C an einen Arduino übertragen. Der Arduino interpretiert die Koordinaten und steuert daraufhin die entsprechenden Komponenten, sodass die Bewegung des Balls verarbeitet und gesteuert wird. Der Zyklus wiederholt sich kontinuierlich,

was das System in einem fortlaufenden Regelkreis hält.

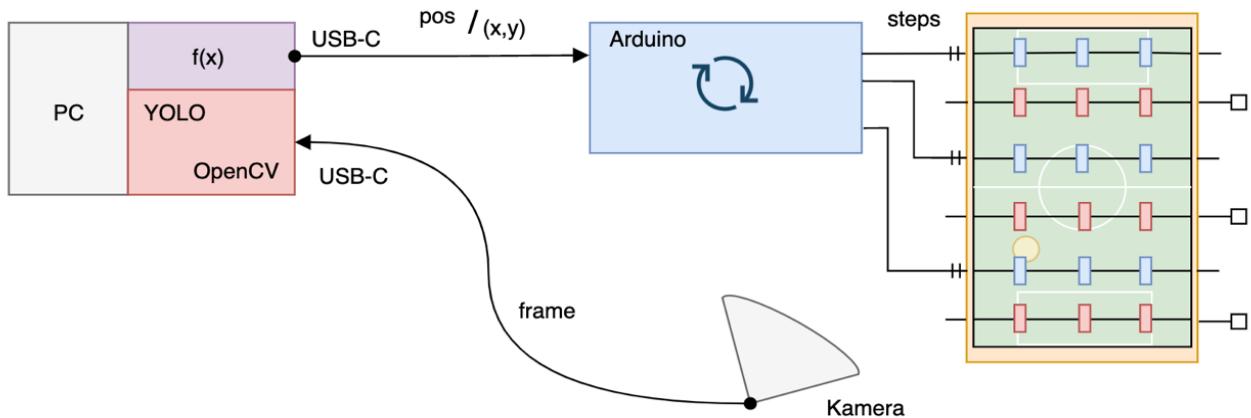


Abb. 16: Aufbau des Regelkreises für einen automatisierten Tischkicker mit Kamera, PC und Arduino.

Ein zentraler Aspekt eines Regelkreissystems ist die reibungslose Kommunikation zwischen den Komponenten. Engpässe („Bottlenecks“) können Fehler oder Leistungseinbussen verursachen. Abb. 17 zeigt: In Beispiel 1) erfolgt die Übermittlung regelmäßig und mit gleichmässigen Abständen, was einen stabilen Informationsfluss gewährleistet. Im Gegensatz dazu zeigt Beispiel 2) unregelmässige Übertragungen mit Engpässen, die den Regelkreis stören und das System negativ beeinflussen können.

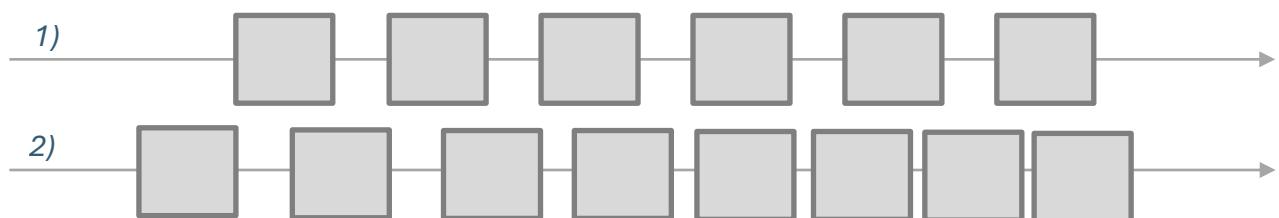


Abb. 17: Darstellung des Informationsflusses in einem Regelkreis: geordnet und regelmässig (Beispiel 1) vs. unregelmässig mit Engpässen (Beispiel 2).

3.3 Kamera Auswahl

Im Rahmen des Bilderkennungssystems wird untersucht, welche Kamera mit dem YOLOv11-Modell verwendet werden sollte, um die Effizienz und Leistungsfähigkeit des Systems zu maximieren. Dazu wird berechnet, wie lange die Kamera benötigt, um ein einzelnes Bild aufzunehmen und wie diese Zeit mit der Verarbeitungskapazität des YOLOv11-Modells abgestimmt werden kann.

Die Berechnung erfolgt anhand der Formel:

$$\text{Zeit pro Bild (in ms)} = \frac{1000}{\text{FPS}} \text{ ms}$$

Für 60 FPS:

$$\text{Zeit pro Bild (in ms)} = \frac{1000}{360} \approx 16,67 \text{ ms}$$

Für 120 FPS:

$$\text{Zeit pro Bild (in ms)} = \frac{1000}{360} \approx 8,33 \text{ ms}$$

Für 240 FPS:

$$\text{Zeit pro Bild (in ms)} = \frac{1000}{360} \approx 4,17 \text{ ms}$$

Für 360 FPS:

$$\text{Zeit pro Bild (in ms)} = \frac{1000}{360} \approx 2,78 \text{ ms}$$

3.4 Analyse der FPS-Kompatibilität

Es wurde entschieden, die neueste YOLO-Version 11 zu verwenden, da sie laut Analyse (siehe Abschnitt "Grundlage YOLO") 2024 die leistungsstärkste Version ist. In Abb. 18 wird gezeigt, welche Kameras und FPS-Raten optimal mit YOLOv11 zusammenarbeiten. Wenn ein Modell oberhalb der grauen Linie liegt, ist die Latenz zu hoch, um die FPS-Rate effizient zu unterstützen. Das Modell benötigt mehr Zeit für die Bildverarbeitung als die Kamera für die Aufnahme eines neuen Bildes, wodurch nicht alle Bilder rechtzeitig verarbeitet werden können.

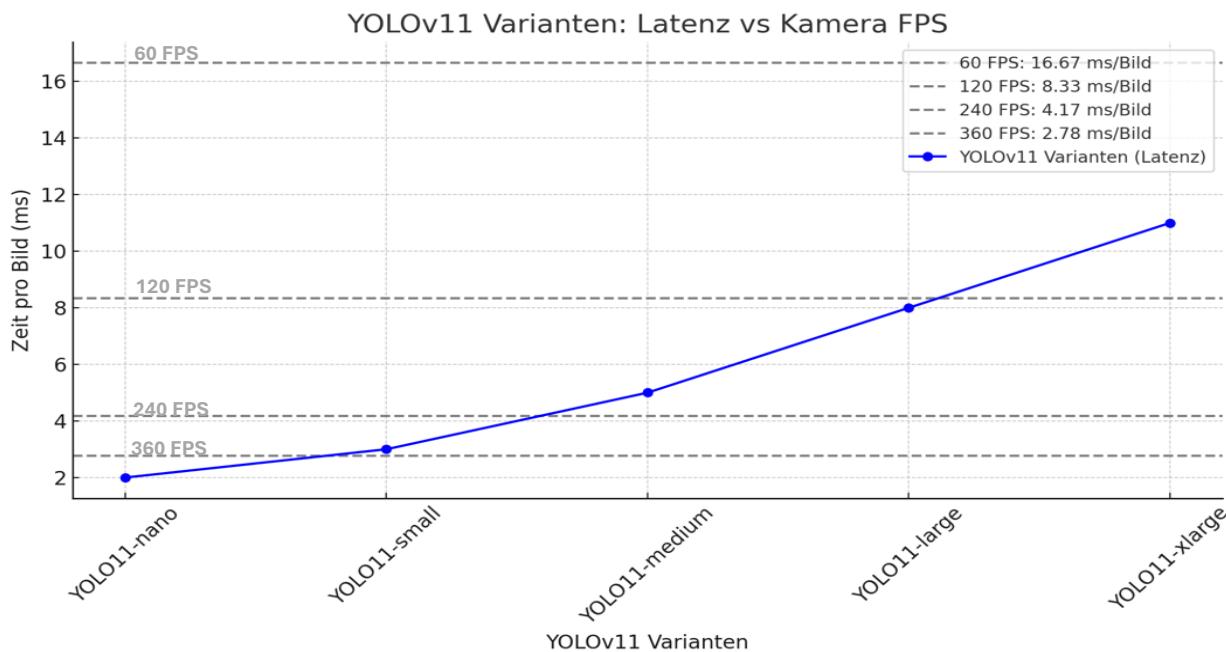


Abb. 18: YOLO-Modell 11 in verschiedenen Versionen, verglichen mit einer Kamera mit 60 bis 360 FPS.

3.4.1 Wahl des YOLOv11 Nano Modells

Für dieses Projekt wurde das YOLOv11-Nano-Modell gewählt, die kleinste Variante der Version 11. Eine FPS-Rate von 120 bis 360 wäre theoretisch ideal, um das Modell optimal zu nutzen. Die Daten basieren jedoch auf einem High-End-System mit leistungsstarker Grafikkarte (siehe Abschnitt Grundlagenbeschreibung YOLO). Diese Ergebnisse spiegeln nicht die Leistung des MacBook mit M1-Chip wider, das für das Projekt verwendet wird. Auf diesem System benötigt die Bildverarbeitung 0,56 bis 0,98 ms pro Bild, sodass eine FPS-Rate von 60 ausreichend ist. Höhere Werte wären hier überflüssig.

3.5 Nutzwertanalyse zur Auswahl der Technik

Die Nutzwertanalyse ist ein Verfahren, mit dem verschiedene Optionen anhand festgelegter Kriterien bewertet werden, um die bestmögliche Entscheidung zu treffen. Daten wie Kosten wurden aus dem Internet recherchiert, während subjektive Einschätzungen zu Zeitaufwand, Risiko und Bedienbarkeit projektspezifisch vorgenommen wurden.

Kriterien

- Kosten: Klare, angemessene Anschaffungskosten bei begrenztem Budget.
- Qualität: Zuverlässige, leistungsstarke Motoren
- Flexibilität: Kompatibel mit anderen Geräten, erweiterbar für Upgrades.
- Risiko: Sichere Lieferung mit Garantie- und Rückgaberegelungen.
- Zeitaufwand: Schnelle Lieferung, einfacher Aufbau.
- Bedienbarkeit: Leichte Handhabung und Installation, auch ohne Vorkenntnisse.

Nutzenwertanalyse für die Auswahl von Technikteilen für einen automatisierten Tischkicker							
G	=	Gewichtung Fertig gekaufte Tech-	Selbstbau mit hoch-	Selbstbau mit chinesi-			
B	=	Bewertung nik	wertigen Teilen	schen Teilen			
N = Nutzwert							
G	B	N	B	N	B	N	
1. Kosten	30	1	30	3	90	6	180
2. Qualität	15	6	90	4	60	3	45
3. Flexibilität	10	5	50	4	40	4	40
4. Zeitaufwand	20	5	100	3	60	3	60
5. Risiko	15	4	60	3	45	2	30
6. Bedienbarkeit	10	4	40	4	40	4	40
Total	100						
Resultat		370		335		390	
Rangfolge		2. Rang		3. Rang		1.Rang	

*Die Bewertung geht von 1 bis 6 (1 = sehr schlecht, 6 = sehr gut). Die Summe aller Gewichtungen muss 100 Punkte ergeben. Gewichtung x Bewertung = Nutzwert.

3.6 Sankey Diagramm

Das Sankey Diagramm ist ein Flussdiagramm, das die Aufteilung der Kosten für das Produkt darstellt. Es veranschaulicht, wie das Gesamtbudget auf die verschiedenen Komponenten verteilt ist. Das Zielbudget für dieses Projekt liegt unter 300 CHF, um das MVP kosteneffizient aufzubauen. Die Daten basieren auf dem Preisstand von 2024.

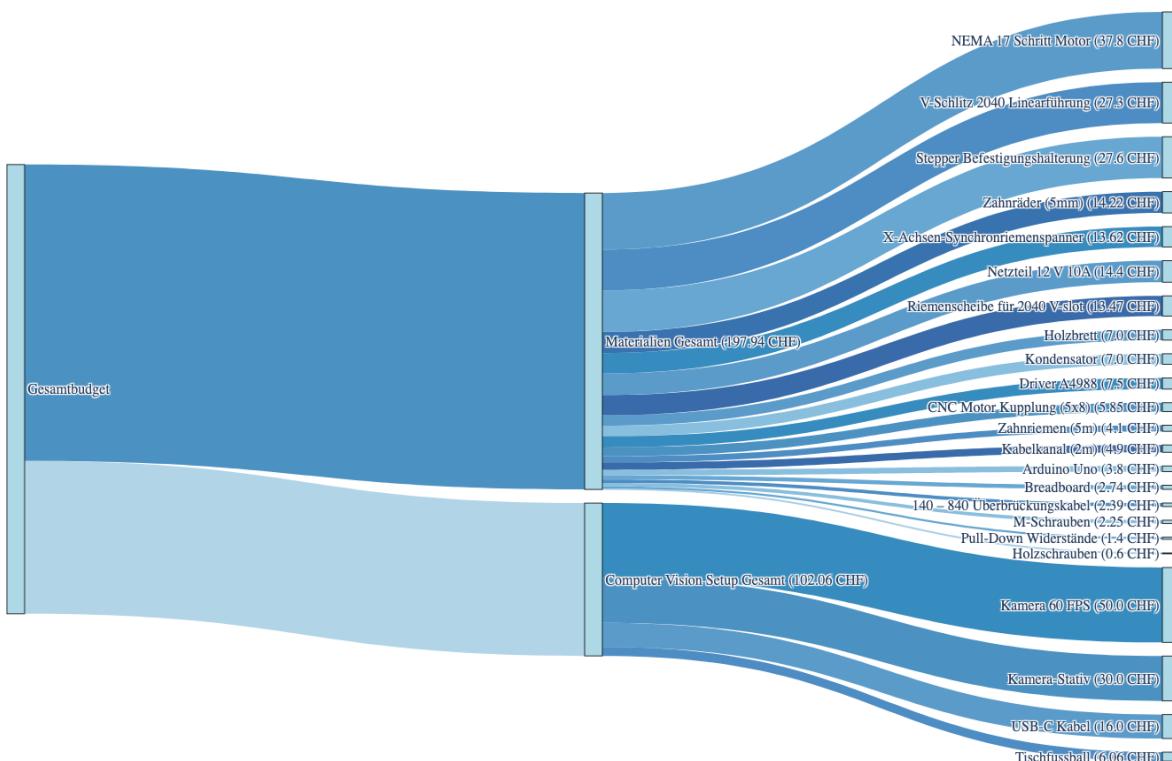


Abb. 19: Das Sankey-Diagramm veranschaulicht die Kostenaufteilung aller Komponenten des automatisierten Tischkickers.

Alle Kostendaten wurden aus Bestellungen bei Temu und Amazon entnommen. Eine detaillierte Auflistung der Kosten sowie die Bestellhistorie sind im Anhang hinterlegt.

3.6.1 Vergleich der Kostenanteile

Die Kostenaufteilung zeigt, dass die Kamera den grössten Anteil am Budget hat, da sie für die Ball-Erkennung essenziell ist. Der Rest des Budgets entfällt überwiegend auf die Motorsteuerung und den Aufbau der Konstruktion. Insgesamt machen diese beiden Bereiche etwa zwei Drittel des Budgets aus, während das Kamera-Setup ein Drittel des Budgets beansprucht. Innerhalb des Kamera-Setups hat die Kamera selbst den grössten Kostenanteil. Siehe Abb. 19.

4 Umsetzung

Im dritten und wichtigsten Abschnitt, der Umsetzung, werden die geleistete Arbeit, wichtige Erkenntnisse sowie weitere Erfahrungen festgehalten.

4.1 Konstruktion

Die Stangen wurden auf einem Holzbrett montiert, um Vibrationen des Kastens zu minimieren. Die Motoren sind so angeordnet, dass der unabhängige oder parallele Betrieb von Linearmotoren und Rotationsmotoren möglich ist. Für die Bewegungsübertragung wurde eine präzise Riemenkonstruktion gewählt (siehe Abschnitt "Planung"). Alle Bauteile wurden gemäss den technischen Spezifikationen montiert und verschraubt (siehe Abb. 20).

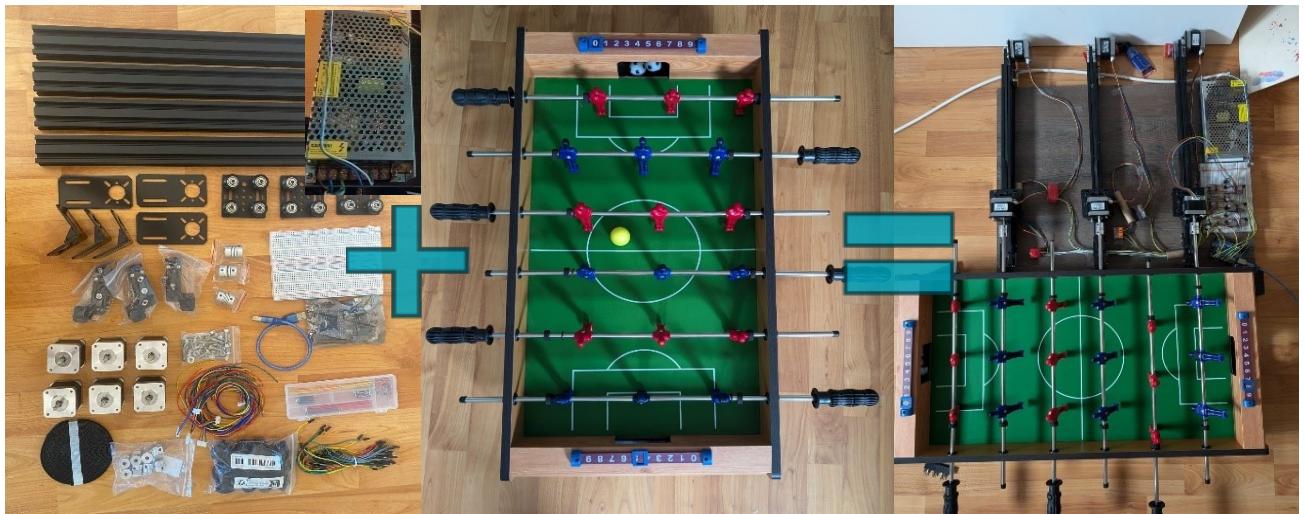


Abb. 20: Übersicht der Konstruktion von Anfang bis Ende.

Das Netzteil wandelt die Netzspannung (230 V AC) in 12 V DC um und liefert bis zu 10 A. Die Nema17-Motoren werden über A4988-Treiber gesteuert, die auf Arduino-Signale reagieren und den Strom schrittweise leiten. Der Stromfluss ist über ein Potentiometer einstellbar. Zur besseren Ballerkennung wurden die Spielfiguren des Tischfussballkastens mit Acrylfarbe versehen (siehe Abb. 20).

Zur Vermeidung von Überhitzung der Treiber wurde ein Ventilator integriert. Ein Kabelkanal aus Kunststoff sorgt zudem für eine ordentliche Kabelverlegung.

4.2 Verdrahtung

Für die Verdrahtung der elektronischen Komponenten wurde ein Breadboard verwendet, da es eine effiziente und übersichtliche Möglichkeit bietet, Schaltungen parallel aufzubauen. Eine zentrale Massnahme war die Sicherstellung einer gemeinsamen Masseverbindung (GND), indem alle GND-Leitungen über das Breadboard miteinander verbunden wurden. Die Spannungsversorgung wurde so organisiert, dass die 12-V-Spannung vom Netzteil leistungsstarke Komponenten versorgt, während die 5-V-Steuerspannung vom Arduino die Logik- und Steuerkreise betreibt. Diese Aufteilung gewährleistet eine stabile und zuverlässige Stromversorgung für die verschiedenen Schaltungsteile.

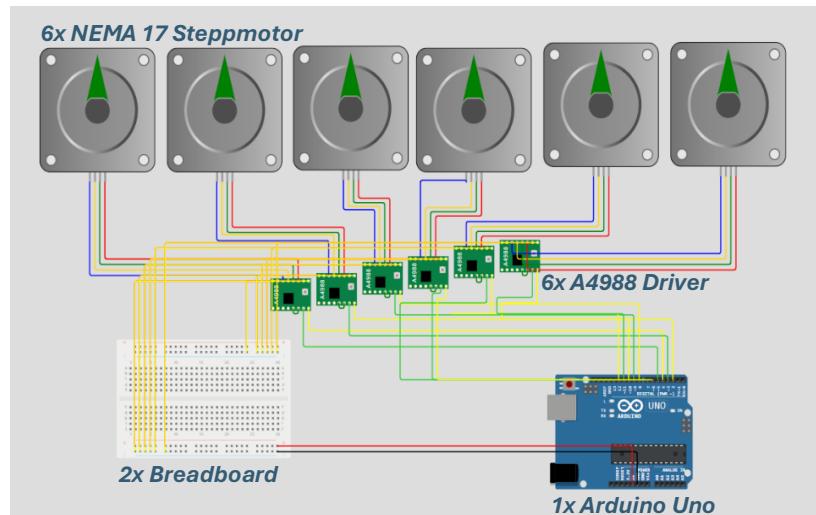


Abb. 21: Die Schaltung zeigt einen Schrittmotor mit Arduino und Treiber, jedoch ohne Kondensatoren, Pull-Down-Widerstände und Netzteil, erstellt im Wokwi Simulator.

In der Abb. 21 fehlen zusätzliche Elemente wie Kondensatoren zur Spannungsstabilisierung, Pull-Down-Widerstände und das Netzteil, die in einem physischen Aufbau für eine verbesserte Stabilität notwendig wären. Diese Details wurden jedoch im Wokwi-Simulator nicht dargestellt.

Die A4988-Treiber wurden mit 12 V, 5 V und GND verbunden. Über die Pins „DIR“ und „STEP“ sind sie mit dem Arduino verknüpft, während die Motorenspulen an die Ausgänge A1, B1, A2 und B2 angeschlossen wurden. Dieses Schema ermöglicht eine präzise Motorsteuerung.

4.2.1 Fehlerquellen bei der Verdrahtung

Bei der Verdrahtung von Schrittmotortreibern wie dem A4988 treten häufig Probleme auf, die die Funktionsweise des Systems beeinträchtigen können. Einige Erfahrungen werden hier nähergelegt.

Gemeinsame Masse (GND): Alle Komponenten wie Treiber, Netzteil und Arduino müssen über eine gemeinsame Masse verbunden sein. Fehlt diese Verbindung, entstehen Potenzialunterschiede, die zu Fehlfunktionen, ungenauer Steuerung oder Schäden führen können.

Spannungsspitzen: Netzteile können Störimpulse erzeugen, die die Stabilität der Schaltung beeinträchtigen. Kondensatoren nahe den Treibern filtern diese Impulse, glätten die Spannung und schützen so die Elektronik.

Sleep- und Reset-Pins: Werden diese Pins nicht verbunden, könnte der Treiber in den Schlafmodus wechseln oder instabil werden. Durch das Verbinden der Pins bleibt der Treiber kontinuierlich betriebsbereit.

Überhitzung: Der Treiber kann bei hohen Belastungen oder langem Betrieb überhitzten und abschalten oder beschädigt werden. Kühlkörper und Ventilatoren helfen, die Wärme abzuleiten und die Lebensdauer zu erhöhen.

Stromeinstellung: Es ist wichtig die Strombegrenzung am Potentiometer exakt auf den Nennstrom des Motors abzustimmen. Falsche Einstellungen können zu unzureichendem Drehmoment oder Überhitzung führen.

Pull-Down-Widerstände: Ohne Pull-Down-Widerstände können ungenutzte Pins undefinierte Zustände annehmen, was zu ungewolltem Verhalten führt. Widerstände setzen diese Pins auf LOW und gewährleisten eine stabile Steuerung.

4.3 Maschinelles Lernen

Um ein benutzerdefiniertes Bildklassifikationsmodell zu erstellen, wird ein geeignetes Datenset benötigt. Für dieses Modell wurde ein Datenset aus 1.000 Bildern eines gelben Tischtennisballs verwendet. Die Bilder wurden unter verschiedenen Bedingungen aufgenommen, einschliesslich variierender Aufnahmewinkel, unterschiedlicher Beleuchtung und schneller Belichtungszeiten, um eine möglichst hohe Diversität und Robustheit des Modells zu gewährleisten (siehe Abb. 22 für Beispiele des Datensets).

4.3.1 Bilder Labeln

Die Bildbearbeitung und -labeling erfolgte mit der Software **Roboflow**, einer Plattform, die Entwicklern hilft, benutzerdefinierte Modelle für maschinelles Lernen zu erstellen, indem sie Bilder labeln, bearbeiten und optimieren. Der gelbe Ball wurde in jedem Bild manuell mit der Roboflow-Oberfläche markiert (siehe Abb. 23). Dieser Prozess wurde für jedes Bild einzeln durchgeführt. Anschliessend wurde das Datenset in Trainings-, Test- und Validierungsdaten aufgeteilt, wobei 70 % der Bilder für das Training, 15 % für die Validierung und 15 % für den Test verwendet wurden. (Nicolai 2024)

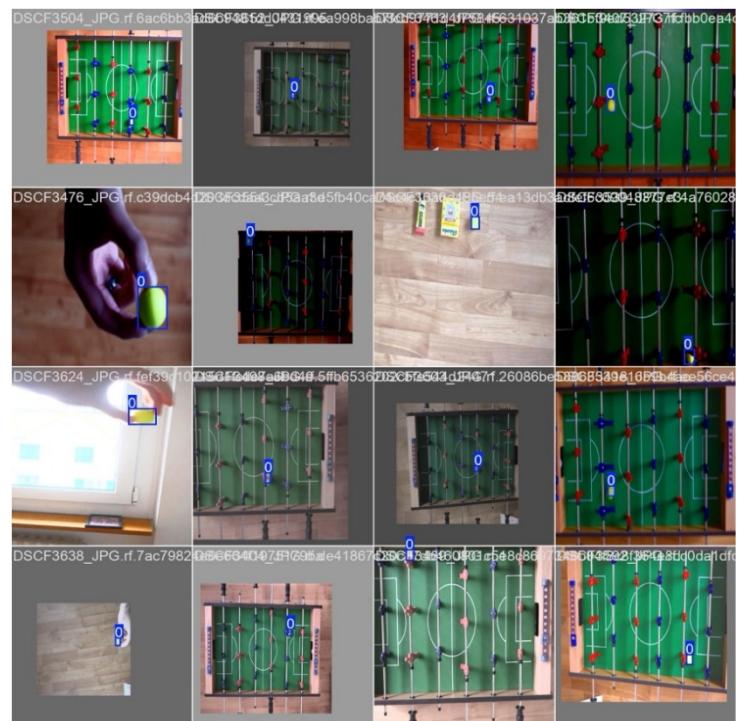


Abb. 22: Datenset zum Trainieren eines Bildklassifikationsmodells für die Erkennung eines Balls.

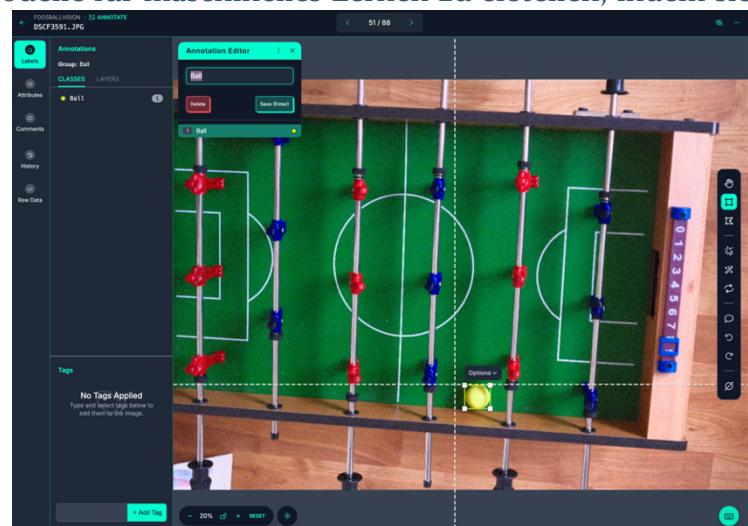


Abb. 23: Manuelle Annotation eines gelben Balls in Roboflow mittels Bounding Box zur Vorbereitung des Datensets für das Modelltraining.

4.3.2 Training eines YOLOv11 Nano Modells

Sobald alle Daten vorliegen, wird das YOLO-Modell mit den eigenen, erstellten Daten trainiert. Zur Durchführung des Trainings wurde **GoogleColab** verwendet, da es eine kostenlose Umgebung mit **CUDA-fähigen NVIDIA-Grafikkarten** bietet (**Nicolai 2024**). Eine leistungsstarke Grafikkarte ist für das Training entscheidend, da sie die Trainingszeit erheblich verkürzt. Der Quellcode für das Training sowie eine detaillierte Beschreibung des Ablaufs sind auf Englisch im entsprechenden GitHub-Repository verfügbar.

Im Folgenden wird der Trainingsprozess mit dem YOLOv11 Nano-Modell erklärt, unterstützt durch ein Code-Snippet.

```
results = model.train(data="/content/FossballVision/data.yaml", epochs=100, imgsz=640, batch=21)
```

Das "Nano"-Modell wurde gewählt, da es auf einem MacBook mit einem weniger leistungsstarken Prozessor effizient läuft, da kleinere Modelle in diesem Fall effektiver sind. Der Trainingsprozess erfolgt über mehrere **Epochen**, wobei eine Epoche angibt, wie oft das Modell den gesamten Datensatz durchläuft. Die Bildgrösse (imgsz) wird auf 640x640 Pixel festgelegt, um die Bilder für das Training zu skalieren. Die **Batch** Size bestimmt, wie viele Bilder gleichzeitig in einem Schritt verarbeitet werden, bevor die Modellgewichte aktualisiert werden. Eine grössere Batch-Grösse kann das Training schneller machen, erfordert jedoch mehr Arbeitsspeicher. In diesem Beispiel beträgt die Batch-Grösse 21.

Beim Training eines YOLO-Modells werden Bilder mit markierten Objekten (Bounding Boxes) verwendet, damit das Modell lernt, ähnliche Muster zu erkennen. Das Modell versucht, selbstständig eine Bounding Box um das erkannte Objekt zu zeichnen. Diese Vorhersage wird mit der echten Bounding Box aus den Trainingsdaten verglichen. Dabei wird die Überlappung mit der Methode „Intersection over Union“ (IoU) berechnet. Ein hoher IoU-Wert zeigt, dass das Modell das Objekt gut erkannt hat. In der Abb. 24 ist eine Überlappung von 60 % dargestellt. Das Modell passt seine internen Neuronen an, um bei jedem Durchlauf bessere Vorhersagen zu machen. Am Ende des Modelltrainings, wenn das Training als abgeschlossen betrachtet wird, wird die Leistung des Modells mithilfe von Validierungsdaten überprüft, um zu beurteilen, wie zuverlässig es Objekte erkennt. (Weiss 2024)

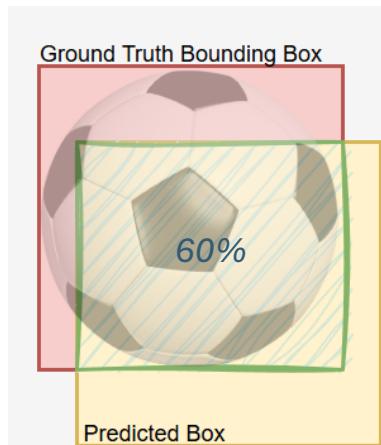


Abb. 24: Erkennung eines Balls durch das Modell. Die rote Bounding Box zeigt die tatsächliche Position, die gelbe die vorhergesagte Position.

4.3.3 Wichtige Erkenntnis

Je mehr Durchgänge das Modell durch den gesamten Datensatz macht, desto besser wird es. Ein Beispiel aus der realen Welt: Ein Schüler muss seinen Wortschatz mehrmals wiederholen, um ihn wirklich zu verstehen. Ein einmaliges Durchgehen reicht nicht – es muss immer wieder geübt werden.

Achtung:

Zu viele Wiederholungen können jedoch dazu führen, dass das Modell den Datensatz auswendig lernt, anstatt allgemeinere Muster zu erkennen. Es könnte dann Schwierigkeiten haben, auf neuen, unbekannten Szenarien zu erkennen, ähnlich wie ein Schüler, der nur Aufgaben auswendig lernt, ohne das Konzept zu verstehen. Deshalb ist es wichtig, das richtige Gleichgewicht beim Training zu finden – weder zu wenig noch zu oft trainieren. Die angehängte Grafik zeigt, wie sich der Trainingsfortschritt in den verschiedenen Phasen entwickelt.

4.4 Aufbau des Bilderkennungssystems

Die Verarbeitung der Spielfeld-Erkennung erfolgt vollständig auf einem Laptop, in diesem Fall einem MacBook mit M1-Chip. Ein leistungsstarker Computer ist erforderlich, um die notwendigen Berechnungen effizient durchzuführen.

4.4.1 Spielfeld-Erkennung

Die Software erkennt das Spielfeld durch Farberkennung und erstellt eine Maske, die die rechteckige Spielform sowie die Ecken mit eindeutig markierten Punkten definiert. Die Abstände zwischen diesen Punkten werden berechnet, um Pixelkoordinaten in reale Masse umzuwandeln. Störende Objekte, die das Spielfeld teilweise verdecken, werden herausgefiltert, sodass die rechteckige Form erhalten bleibt.



Abb. 25: Die Spielfeld-Erkennung wird durch die grüne Farbe angezeigt, woraufhin eine Maske erstellt wird.

Diese Methode wurde gewählt, da sie ohne komplexe KI-Modelle auskommt und durch einfache Farberkennung effizient ist. Die Kameraposition beeinflusst die Erkennung nicht, jedoch kann schlechte Beleuchtung die Zuverlässigkeit mindern. Abb. 25 zeigt die Maskenerstellung (weisses Feld) und das virtuelle Gitter zur Umrechnung der Koordinaten in reale Masse.

4.4.2 Ball-Erkennung

Das entwickelte Modell zur Ball-Erkennung arbeitet effizient und mit minimalem Code. Mit Hilfe des Frameworks Ultralytics war es möglich, schnell eine funktionierende Lösung zu erstellen. Die Hauptaufgabe bestand darin, die vom Modell gelieferten Ballkoordinaten, die zunächst in Pixeln angegeben sind, in reale Masseneinheiten umzurechnen.

Nach der Umrechnung der Pixelkoordinaten in reale Masse übermittelt das System die Ballposition an die Motorsteuerung. Die Koordinaten haben eine Toleranz von ± 1 cm, da die Bounding Box des Modells nicht immer exakt auf dem Ball liegt. Dies kann durch weiteres Training optimiert werden. Damit ist das System einsatzbereit und ermöglicht präzise Bewegungssteuerung basierend auf der Ballposition.

4.5 Optimierung durch Ballbewegungsvorhersage

Zur Verbesserung des Systems wurde eine Methode implementiert, die es ermöglicht, die Flugrichtung des Balls vorherzusagen. Dies hilft, den Punkt zu bestimmen, an dem der Ball voraussichtlich landen wird, sodass die Spielfiguren frühzeitig in Position gebracht werden können. Zudem kann das System vorübergehende Koordinaten berechnen, falls der Ball in einem Frame nicht vom YOLO-Modell erkannt wird.

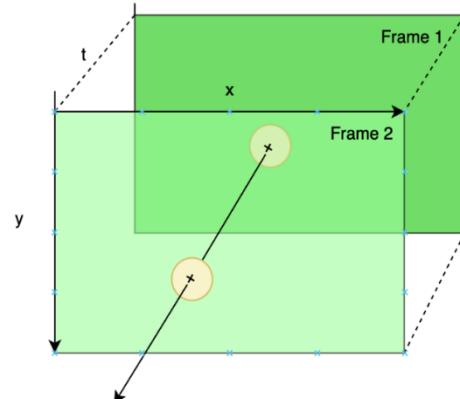


Abb. 26: Zeigt den Vektor zwischen Frames

4.5.1 Funktionsweise der Richtungsvektor-Methode

Die Vorhersage basiert auf einem Bewegungsvektor des Balls, der aus den Positionsänderungen in aufeinanderfolgenden Frames berechnet wird, wie in der Abb. 26 dargestellt.

Wesentliche Parameter für diese Berechnung sind:

- **Delta-Zeit (Δt):** Der Zeitabstand zwischen zwei aufeinanderfolgenden Frames.
- **Delta-X (Δx) und Delta-Y (Δy):** Die Positionsänderungen des Balls entlang der x- und y-Koordinaten des Spielfelds.

Der Bewegungsvektor wird durch folgende Formel berechnet:

$$\vec{v} = \begin{pmatrix} \frac{\Delta x}{\Delta t} \\ \frac{\Delta y}{\Delta t} \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

Mit diesem Vektor kann die zukünftige Ballposition vorhergesagt werden.

Für den Frame $n+1$ gilt:

$$x_{n+1} = x_n + v_x \times \Delta t, \quad y_{n+1} = y_n + v_y \times \Delta t$$

Die Berechnung wurde von Alexander Körner, BMZ-Physiklehrer, überprüft und validiert.

4.6 Motor Logik

Die Funktionslogik der Motoren basiert auf einem Rastersystem, wie in Abb. 27 dargestellt. Das Spielfeld ist in ein 6x6-Raster unterteilt, wobei jeder Motor einer Zelle (0 bis 5 in x- und y-Richtung) zugeordnet ist. Dies ermöglicht es der Kamera, den Ball präzise zu lokalisieren und nur die Motoren zu aktivieren, die sich in der Nähe des Balls befinden, wodurch unnötige Bewegungen vermieden und die Effizienz gesteigert werden.

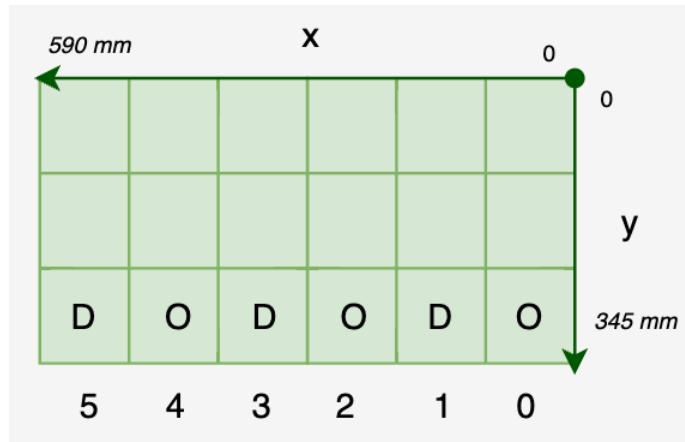


Abb. 27: Zeigt die Unterteilung des Rasters
(O= Offense, D= Defense)

Es gibt zwei Modi: **Offense** (Angriff) und **Defense** (Verteidigung). Im Offense-Modus läuft ein anderer Teil des Programmes als im Defense-Modus, was eine optimale Anpassung an die Spielsituation ermöglicht.

- **Angriff**

Im Angriffsmodus führt der nächste Spieler am Ball einen konstanten Kick aus, während sich andere Spieler aus dem Weg nach oben bewegen, um den Ball ungehindert aufs Tor zu bringen.

- **Verteidigung**

Im Verteidigungsmodus versuchen alle Spieler, den Ball zu blockieren und ein Tor des Gegners zu verhindern.

Weitere Spiellogik und der Quellcode für die Motorsteuerung sind im GitHub-Repository „[FoosballVision](#)“ zu finden.

5 Feldtest

Ein Spiel geht auf 5 Tore. Wer zuerst 5 Tore erzielt gewinnt. In Abb. 28 sieht man die Tore die in den 48 Spielen von Maschine / Mensch geschossen wurden.

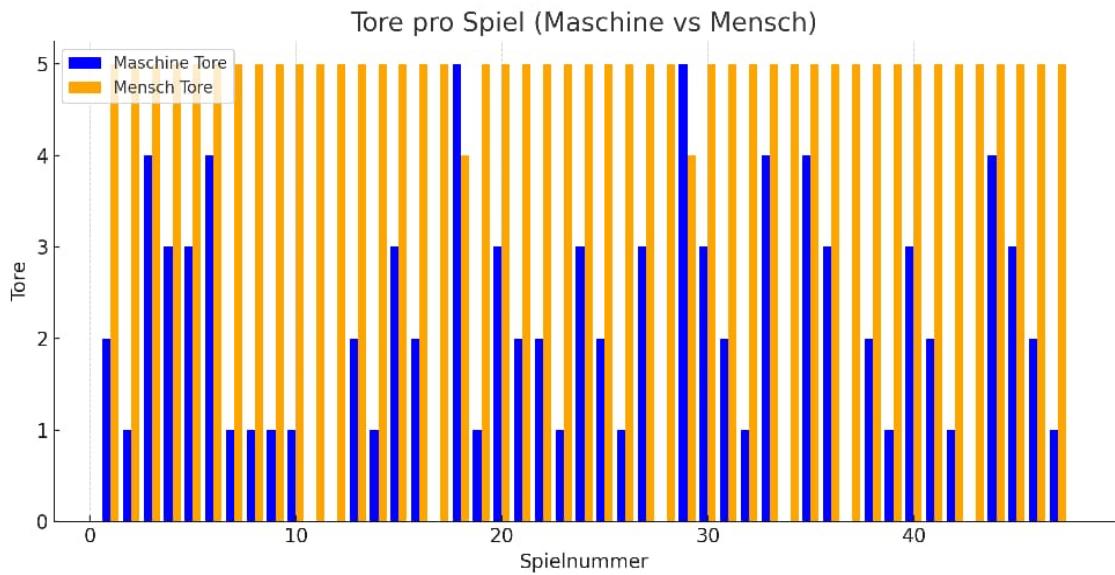


Abb. 28: Zeigt alle insgesamt geschossenen Tore.

Das Diagramm zeigt, dass ein Freizeitspieler dem Tischkicker-System deutlich überlegen ist. Dennoch erzielt der Tischkicker konstant einige Tore gegen seine menschlichen Gegenspieler. Der durchschnittliche Torwert des autonomen Kickers berechnet sich wie folgt:

$$\text{Durchschnitt} = \text{Tore System} \div \text{Anzahl Spiele} = \\ 93 \div 48 \approx 1,94 \text{Tore/Spiel}$$

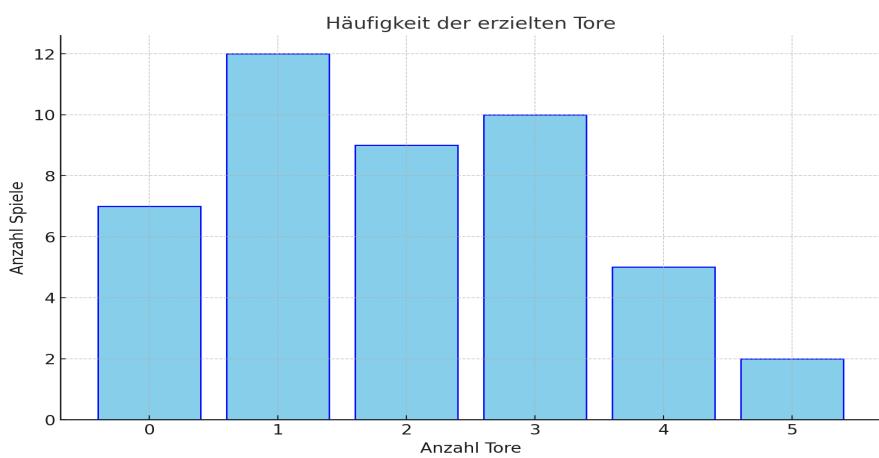


Abb. 29: Zeigt die Anzahl Spiele, bei denen eine gewisse Punktzahl erreicht wurde.

Abb. 29 verdeutlicht, dass 1 Tor die häufigste Trefferquote darstellt.

Im Anhang finden Sie eine Excel-Datei mit den tabellarischen Daten aller Spiele.

5.1 Echtzeit Analyse Tischkicker

Um zu prüfen, ob das System wie in der Hypothese in Echtzeit arbeitet, muss es genauer untersucht werden. Echtzeit bedeutet, dass Aktionen nahezu verzögerungsfrei ablaufen und Informationen in Sekundenbruchteilen verarbeitet werden.

Echtzeit = vorgegebene Zeit, die bestimmte Prozesse einer elektronischen Rechenanlage in der Realität verbrauchen dürfen (Duden, 21.11.2024)

```
0: 480x640 1 Ball, 54.0ms
Speed: 1.5ms preprocess, 54.0ms inference, 0.6ms postprocess per image at shape (1, 3, 480, 640)
Data has been sent
Ball position in court coordinates: (355.01, 331.24)

0: 480x640 1 Ball, 51.7ms
Speed: 1.6ms preprocess, 51.7ms inference, 0.6ms postprocess per image at shape (1, 3, 480, 640)
Data has been sent
Ball position in court coordinates: (357.70, 325.67)

0: 480x640 1 Ball, 53.5ms
Speed: 1.5ms preprocess, 53.5ms inference, 0.7ms postprocess per image at shape (1, 3, 480, 640)
Data has been sent
Ball position in court coordinates: (358.40, 325.00)

0: 480x640 1 Ball, 61.1ms
Speed: 1.6ms preprocess, 61.1ms inference, 0.7ms postprocess per image at shape (1, 3, 480, 640)
Data has been sent
Ball position in court coordinates: (358.32, 322.81)
```

Abb. 30: Zeigt die Verarbeitungszeit der Ballerkennung und die Prozessdauer.

Der grösste Teil der „Reaktionszeit“ des Systems entfällt auf die eigentliche Bildverarbeitung, insbesondere auf die Objekterkennung. Laut Terminal-Daten variiert die Interferenzzeit – also die Zeit zwischen der Verarbeitung einzelner Bilder- zwischen 50 und 100 Millisekunden. Was letztlich etwa 10 FPS entspricht (wie in Abb. 30 ersichtlich)

Es kann argumentiert werden, dass, solange die Reaktionszeit des Systems schneller ist als die durchschnittliche Menschliche Reaktionszeit eine Kommunikation in „Echtzeit“ stattfindet. Wie gut ist also die durchschnittliche Menschliche Reaktionszeit in Bezug auf einen Tischkicker?

5.2 Menschliche Reaktionszeit

Es gibt vergleichbare Studien zur menschlichen Reaktionszeit auf visuelle Reize, wie z.B. Licht. Diese belaufen sich auf eine durchschnittliche Reaktionszeit von etwa 200–250 ms ((Castro-Palacio, et al. 2021); (Jain, et al. 2015)). Dazu kommt noch die eher komplizierte motorische Reaktion (Reaktion des Muskels), die weitere 100–150 ms in Anspruch nimmt. (Wikipedia, „Mental chronometry“, 2024.10.23). Für eine persönliche Prüfung kann selbstständig ein Test der eigenen Reaktionszeit über humanbenchmark.com durchgeführt werden.

Daraus ergibt sich eine geschätzte menschliche Reaktionszeit von etwa 300–400 ms für das Wegkicken eines Balls beim Tischfussball. Somit ist die Verarbeitung unseres Systems theoretisch tatsächlich schneller, als die physiologische Reaktionszeit eines Menschen.

Allerdings spielt die Intuition des Menschen eine wichtige Rolle. Unsere Sinne arbeiten zusammen und helfen uns, die Position des Balls im Voraus zu erahnen und so das Spiel zu lesen. Wir können dadurch schneller reagieren als es die reine physiologische Reaktionszeit vermuten lässt. Ein automatisiertes System kann den Ball auch nicht mit hundertprozentiger Gewissheit erkennen. Hinzu kommen Physische Grenzen der Motoren. Die physischen Grenzen der Motoren spielen eine Rolle. Mit besserer Hardware, insbesondere Grafikkarten, könnte das-selbe YOLO-Modell deutlich bessere Ergebnisse liefern und die menschliche Spielleistung näher erreichen. Ideal wäre eine Bildverarbeitung in unter 2 ms (siehe Abb. 1).

6 Schlusswort

Unser technischer Beitrag bestand in der Entwicklung eines Systems, das mittels eines trainierten YOLO-Computer-Vision-Modells die Bewegungen von Motoren über Richtungsvektoren steuert. Dies umfasste den Aufbau eines Regelkreises zur Analyse und Reaktion auf Ballpositionen und -geschwindigkeiten, die Integration der Hardwarekomponenten sowie die Logik für die Ballvorhersage unter den begrenzten Bedingungen der verfügbaren Hardware.

Die Hypothese, dass der Tischkicker 50 % der Spiele gewinnen kann, wurde widerlegt: Im Feldtest lag die Erfolgsrate bei durchschnittlich 4 % mit zwei Toren pro Spiel. Trotz theoretisch schnellerer Verarbeitungszeit als ein Mensch fehlt es dem System an der Fähigkeit, komplexe Spielsituationen wie Randbälle oder strategische Winkelberechnungen zu erkennen und entsprechend zu handeln. Diese Einschränkungen sind auf die Hardware und die begrenzten Vorhersagefähigkeiten der Algorithmen zurückzuführen.

Trotzdem bietet das Projekt vielversprechende Ansätze für die Weiterentwicklung. Durch maschinelles Lernen könnten Ballverläufe präziser vorhergesagt und komplexe Spielsituationen besser erkannt werden. Die Integration zusätzlicher Sensoren und leistungsfähigerer Hardware könnte die Funktionalität weiter verbessern.

Abschliessend zeigt unsere Arbeit, dass die Grundlagen für ein funktionierendes System gelegt wurden, auch wenn es noch an Präzision und Effizienz fehlt, um mit menschlichen Spielern zu konkurrieren. Die Frage, wie weit technische Systeme den Menschen in dynamischen Umgebungen ersetzen oder sinnvoll ergänzen können, bleibt eine spannende Herausforderung für die Zukunft.

7 Quellenverzeichnis

- [1] Glenn, Jocher. 2024. 01. Oktober. Zugriff am 21. Oktober 2024.
<https://docs.ultralytics.com/solutions/>.
- [2] Bodine. 2024. *Bodine*. 20. 11. Zugriff am 2024. 11 2024. <https://www.bodine-electric.com/products/brushless-dc-motors/34b-series-blcd-motor/3406/>.
- [3] 3DWare. 2024. *3DWare*. 20. 11. Zugriff am 20. 11 2024.
<https://www.3dware.ch/de/zubehor/motoren/3dware-ch-handelsware-01900502-40-nema-17-schrittmotor-1-8-grad-1-2a>.
- [4] Bastelgarage. 2024. *Bastelgarage*. 20. 11. Zugriff am 10. 11 2024.
<https://www.bastelgarage.ch/56x56x76mm-schrittmotor-nema-23-v-4-2a-1-8nm>.
- [5] Makerselectronics. 2024. *Makerselectronics*. 10. 11. Zugriff am 20. 11 2024.
<https://makerselectronics.com/product/servo-motor-half-metal-gear-mg995-towerpro-continuous-rotation>.
- [6] Industrie, Kem. 2024. *Kem Industrie*. 20. 11. Zugriff am 11. 11 2024.
<https://kem.industrie.de/elektromotoren/maxon-motor-verbessert-gleichstrommotoren/>.
- [7] Ultralytics. 2024. *Models Supported by Ultralytics*. 20. 11. Zugriff am 20. 11 2024.
<https://docs.ultralytics.com/models/>.
- [8] Alba, Michael. 2022. *engineering.com*. 22. 10. Zugriff am 20. 11 2024.
<https://www.engineering.com/the-kicker-story-foosball-and-deep-reinforcement-learning/>.
- [9] Krause, Hans Michael. 2017. *Bosch*. Zugriff am 20. 11 2024.
<https://www.bosch.com/stories/bend-it-like-bosch/>.
- [10] Bünte, Oliver. 2024. *Foosbar: Autonomous table soccer robot shoots almost unstoppably*. 11. 6. Zugriff am 20. 11 2024. <https://www.heise.de/en/news/Foosbar-Autonomous-table-soccer-robot-shoots-almost-unstoppably-9758099.html>.
- [11] Naumenko, Xander. 2024. *I Made the World's Best Foosball Robot!* 5. 6. Zugriff am 3. 10 2024. <https://www.youtube.com/watch?v=xrwXZXGiP1w>.
- [12] Ximea. 2020. *Ximea*. 15. 10. Zugriff am 11. 10 2024.
<https://www.ximea.com/news/case-studies/case-study-robot-foosball-table-high-speed-camera>.
- [13] Douglas, W. Jones. 2024. *THE UNIVERSITY OF IOWA Department of Computer Science*. 20. 11. <https://homepage.divms.uiowa.edu/~jones/step/>.

- [14] Castro-Palacio, Juan Carlos, Pedro Fernández-de-Córdoba, J. M. Isidro, Sarira Sahu, und Esperanza Navarro-Pardo. 2021. *Linking Individual and Collective Behaviour Through Physics Modeling*. 10. 3. <https://www.mdpi.com/2073-8994/13/3/451>.
- [15] Jain, Aditya, Ramta Bansal, Avnish Kumar, und KD Singh. 2015. *PubMedCentral*. 2. 5. <https://pmc.ncbi.nlm.nih.gov/articles/PMC4456887/>.
- [16] Daniel, Kamil Andrzej, Paweł Kowol, und Grazia Lo Sciuto. 2024. „Linear Actuators in a Haptic Feedback Joystick System for Electric Vehicles.“ *MDPI journal* 4.
- [18] Weiss, Manuel, Interview geführt von Sy Viet Dao. 2024. *Yolo / Maschinelles Lernen* (4. 11).
- [19] Nicolai. 2024. *YouTube*. 26. 7. Zugriff am 1. 10 2024. <https://www.youtube.com/watch?v=LNwODJXcvt4>.

8 Abbildungsverzeichnis

- 1 Zeigt eine Übersicht der YOLO-Modelle bis Version 11 und veranschaulicht die Leistung jedes Modells in Millisekunden bei einem spezifischen Prozess (Ultralytics, 2024).
- 2 YOLO-Erkennung von Menschen in Echtzeit.
- 3 *Ball mit Motion Blur, aufgenommen mit einer 60-fps Kamera ohne spezifische Einstellung.*
- 4 Zeigt das Belichtungsdreieck (ertraeume-deine-welt, 2024)
- 5 NEMA 23 Schrittmotor (Bastelgarage, 2024).
- 6 NEMA 17 Schrittmotor (3DWare, 2024).
- 7 Zeigt einen modernen BLDC-Motor (Bodin, 2024).
- 8 MG 996 Servomotor (Makerselectronics, 2024).
- 9 Gleichstrommotor (Kem Industrie, 2024).
- 10 A4988 Treiber
- 11 Kick it like Bosch – Automatisierter Tischfussball von Bosch (Krause, 2017).
- 12 EPFL-Student und Professor spielen gegen den Tischfussballtisch (Ximea, 2020).
- 13 Screenshot aus dem Video des YouTube Kanals „From Scratch“ (Naumenko, 2024).
- 14 Zeigt einen linear ball Screw actuator (Daniel, Kowol und Sciuto 2024).
- 15 Zeigt einen Linearmotor mit Riemenantrieb.
- 16 Aufbau des Regelkreises für einen automatisierten Tischkicker mit Kamera, PC und Arduino.
- 17 Darstellung des Informationsflusses in einem Regelkreis: geordnet und regelmässig (Beispiel 1) vs. unregelmässig mit Engpässen (Beispiel 2).
- 18 YOLO-Modell 11 in verschiedenen Versionen, verglichen mit einer Kamera mit 60 bis 360 FPS.
- 19 Das Sankey-Diagramm veranschaulicht die Kostenaufteilung aller Komponenten des automatisierten Tischkickers.
- 20 Übersicht der Konstruktion von Anfang bis Ende.
- 21 Die Schaltung zeigt einen Schrittmotor mit Arduino und Treiber, jedoch ohne Kondensatoren, Pull-Down-Widerstände und Netzteil, erstellt im Wokwi Simulator.
- 22 Datenset zum Trainieren eines Bildklassifikationsmodells für die Erkennung eines Balls.

- 23 Manuelle Annotation eines gelben Balls in Roboflow mittels Bounding Box zur Vorbereitung des Datensets für das Modelltraining.
- 24 Erkennung eines Balls durch das Modell. Die rote Bounding Box zeigt die tatsächliche Position, die gelbe die vorhergesagte Position.
- 25 Die Spielfeld-Erkennung wird durch die grüne Farbe angezeigt, woraufhin eine Maske erstellt wird.
- 26 Zeigt den Vektor zwischen Frames
- 27 Zeigt die Unterteilung des Rasters (O= Offense, D= Defense)
- 28 Zeigt alle insgesamt geschossenen Tore
- 29 Zeigt die Verarbeitungszeit der Ballerkennung und die Prozessdauer.
- 30 Zeigt die Anzahl Spiele, bei denen eine gewisse Punktzahl erreicht wurde
- 1A Ergebnis eines selbst trainierten YOLO-Modells zur Ballerkennung mit einer Genauigkeit von 96 %.
- 2A Ergebnis des YOLO-Modelltrainings nach 100 Epochen.
- 3A Zeigt die Bewertung der Motoren.
- 4A Skizze von YOLO und maschinellem Lernen: Das ideale Modelltraining zielt darauf ab, den globalen Tiefpunkt der Verlustfunktion (Loss-Funktion) zu finden, anstatt in lokalen Minima stecken zu bleiben (Manuel, 2024).
- 5A Grundlage, wie ein neuronales Netzwerk bei YOLO aussehen kann (Manuel, 2024).
- 6A Grundlage, wie ein neuronales Netzwerk bei YOLO aussehen kann (Manuel, 2024).
- 7A Ansicht, wie die Neuronenparameter nach jedem Epoche-Durchlauf angepasst werden, um die Modellleistung zu optimieren. Dieser Prozess erfolgt durch die Anpassung der Gewichte basierend auf dem Fehler, der während des Trainings berechnet wird (Manuel, 2024).
- 8A Tabellarisch aufgelistete Werte des Feldtestes.

9 Anhang

9.1 Glossar

In diesem Glossar werden Fachbegriffe und Abkürzungen erklärt, die in dieser Arbeit verwendet werden und für das Verständnis der Inhalte eine wichtige Rolle spielen.

YOLO (You Only Look Once): Ein neuronales Netzwerk-Algorithmus zur schnellen und präzisen Objekterkennung in Bildern und Videos.

NEMA (National Electrical Manufacturers Association): Eine Organisation in den USA, die Standards für elektrische Geräte und Komponenten entwickelt und festlegt.

BLDC (Brushless Direct Current Motor): Ein bürstenloser Gleichstrommotor, der durch seine höhere Effizienz und Langlebigkeit für Anwendungen in der Automatisierungstechnik und Robotik geschätzt wird.

Breadboard: Eine Steckplatine zur Entwicklung und Testen elektronischer Schaltungen, bei der Bauteile ohne Löten verbunden werden können.

OpenCV (Open Source Computer Vision Library): Eine freie Programmmbibliothek mit zahlreichen Algorithmen für Bildverarbeitung und Computer Vision, häufig genutzt für Echtzeitanwendungen in der Robotik und Automatisierung.

MVP (Minimum Viable Product): Eine Version eines Produkts mit den minimal erforderlichen Funktionen, um den Hauptnutzen für den Nutzer zu bieten und das Konzept in der Praxis zu testen.

Modell: Ein KI-Modell ist eine algorithmische Struktur, die aus Daten lernt, um Muster zu erkennen und Vorhersagen zu treffen.

Data: Das erstellte Datenset mit Labels wird dem Modell zur Verfügung gestellt.

Epochs: Die Anzahl der Epochen gibt an, wie oft das Modell den gesamten Datensatz durchläuft. In diesem Fall sind es 100 Epochen.

Imgsz (Image Size): Die Bildgrösse für das Training wird z.b auf 640x640 Pixel festgelegt.

Linear actuator: Ein Linear Aktor ist ein Gerät, das eine rotierende Bewegung z. B. von einem Motor in eine lineare Bewegung umwandelt.

Ball screw Motor: Ein Ball Screw Motor ist ein Motor, der mit einer Kugelumlaufspindel kombiniert ist. Die Kugelumlaufspindel wandelt die rotierende Bewegung des Motors präzise in eine lineare Bewegung um. Sie verwendet Kugeln zwischen der Spindel und der Mutter, um Reibung zu reduzieren und hohe Effizienz, Präzision sowie Tragfähigkeit zu gewährleisten

Batch Size: Die Batch-Grösse gibt an, wie viele Bilder gleichzeitig trainiert werden. In diesem Fall sind es 16 Bilder pro Batch, bevor die Modellgewichte aktualisiert werden.

9.2 Quellcode

Der vollständige Quellcode für die Ball Erkennung und Motorsteuerung ist im GitHub Repository verfügbar. Die Programmdateien und Dokumentation zur Implementierung finden Sie dort unter folgendem Link: https://github.com/Azukio4/foosball_vision

9.3 Videos, Bilder, Feldtest

Hier sind einige Videos zur Umsetzung und Fertigstellung des automatisierten Tischkickers zu finden, sowie den Zeitplan, den Feldtest und weitere Dokumente. Der Link zum Google Drive lautet:

<https://drive.google.com/drive/folders/1316r4ddjdSs3syisFmXPIUNULWxpFtFC?usp=sharing>

Nutzwertanalyse

Unter diesem link kann eine Preisliste für die Auswahl von Technikteilen abgerufen werden.

<https://docs.google.com/document/d/1YIaUKSXwjzeI5SPqjptCRsO3Si4PyCSz/edit?usp=sharing&ouid=108069173819656800954&rtpof=true&sd=true>

Vergleich Motoren

Treiber	Datenblatt
https://www.pololu.com/file/oj450/a4988_dmos_microstepping_driver_with_translator.pdf	

Stepper	Motor
Stepper System Overview.	

[Control of Stepping Motors - A Tutorial - Douglas W. Jones, The University of Iowa](#)

DC-Motor	
https://www.transmotec.de/produkt-kategori/dc-motoren/	
Servomotoren	
https://mall.industry.siemens.com/mall/de/de/Catalog/Products/10551699?tree=CatalogTree	

BLDC-Motor

<https://www.power-tronic.com/produkte/bldc-motoren/motoren-ohne-getriebe/bldc-bl80s/>

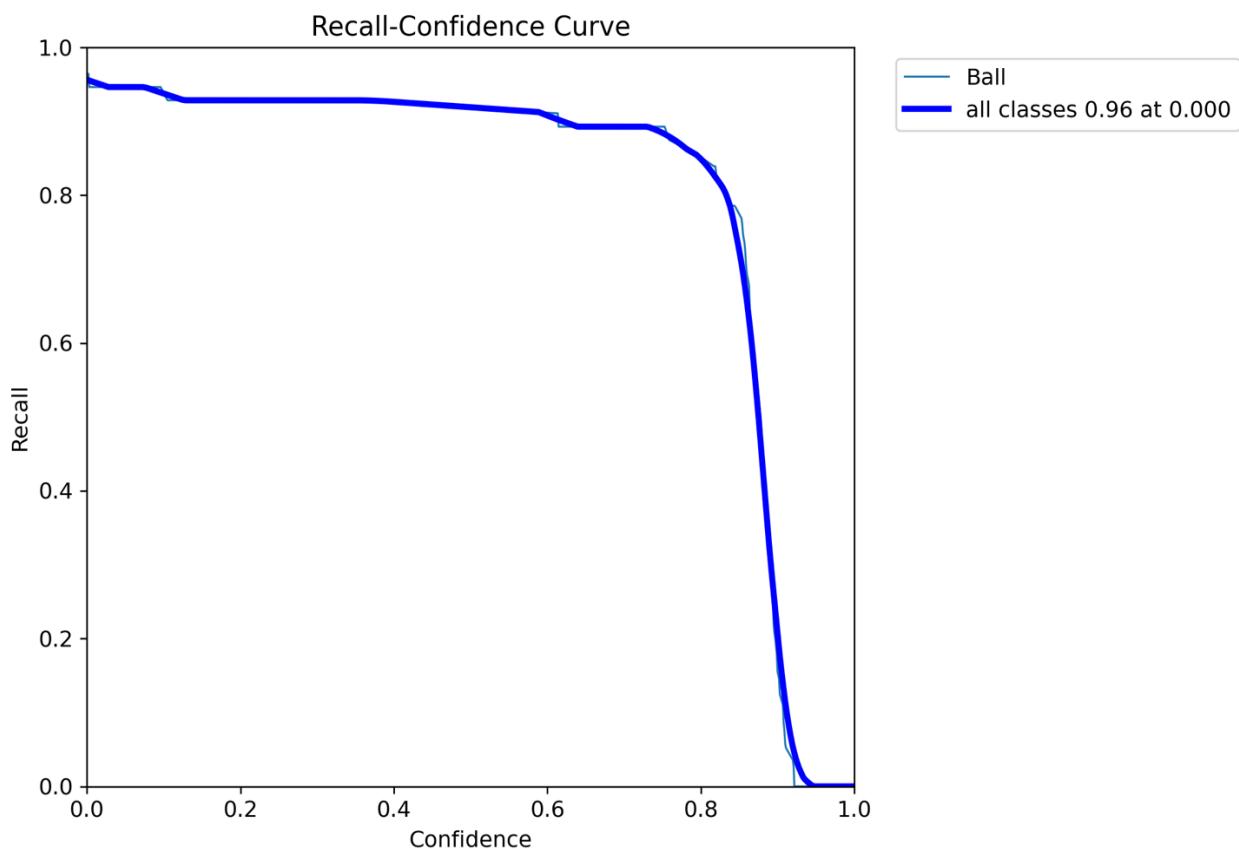


Abb. 1A: Ergebnis eines selbst trainierten YOLO-Modells zur Ballerkennung mit einer Genauigkeit von 96 %.

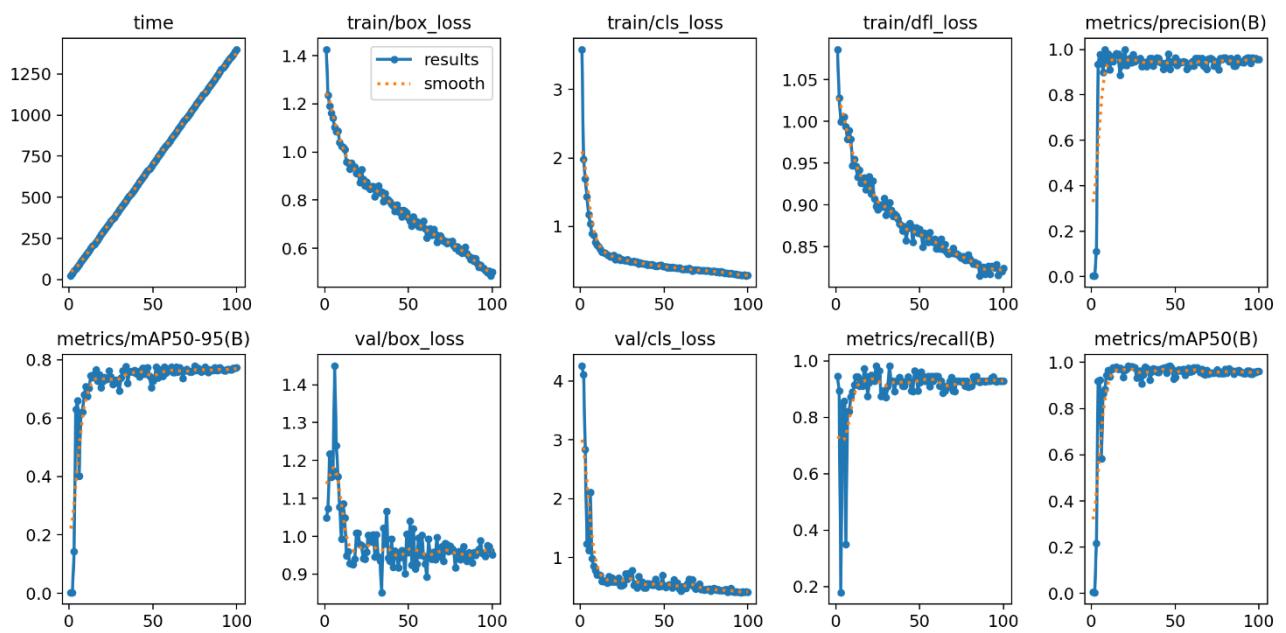


Abb. 2A: Ergebnis des YOLO-Modelltrainings nach 100 Epochen.

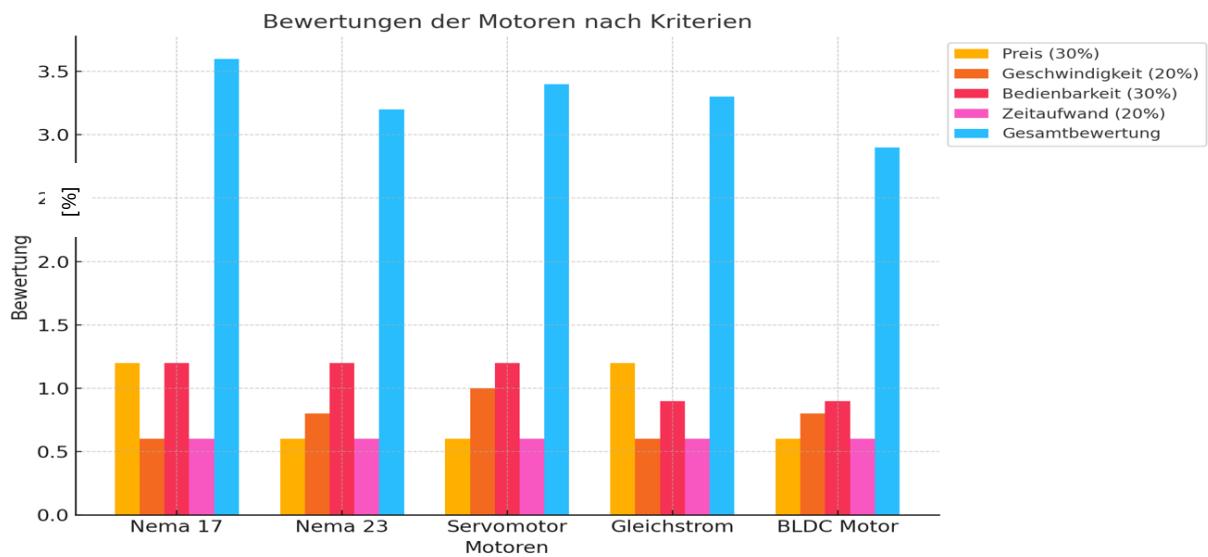


Abb.3A: Zeigt die Bewertung der Motoren.

Interview-Bescheinigung

Name: Manuel Weiss

Hiermit bestätige ich, dass ein Interview sowie eine Erklärung zu den Themen Computer Vision und YOLO mit Sy Viet Dao durchgeführt wurden. Dieses Interview kann für wissenschaftliche Arbeiten verwendet werden.

Die Inhalte des Interviews umfassen:

- Erklärungen zu YOLO und maschinellem Lernen.
- Verständnishilfen durch Diagramme.
- Ansätze und Konzepte zur Verbesserung von Modellen.

Ort und Datum: St. Gallen, 25.11.24

Unterschrift: M. Weiss

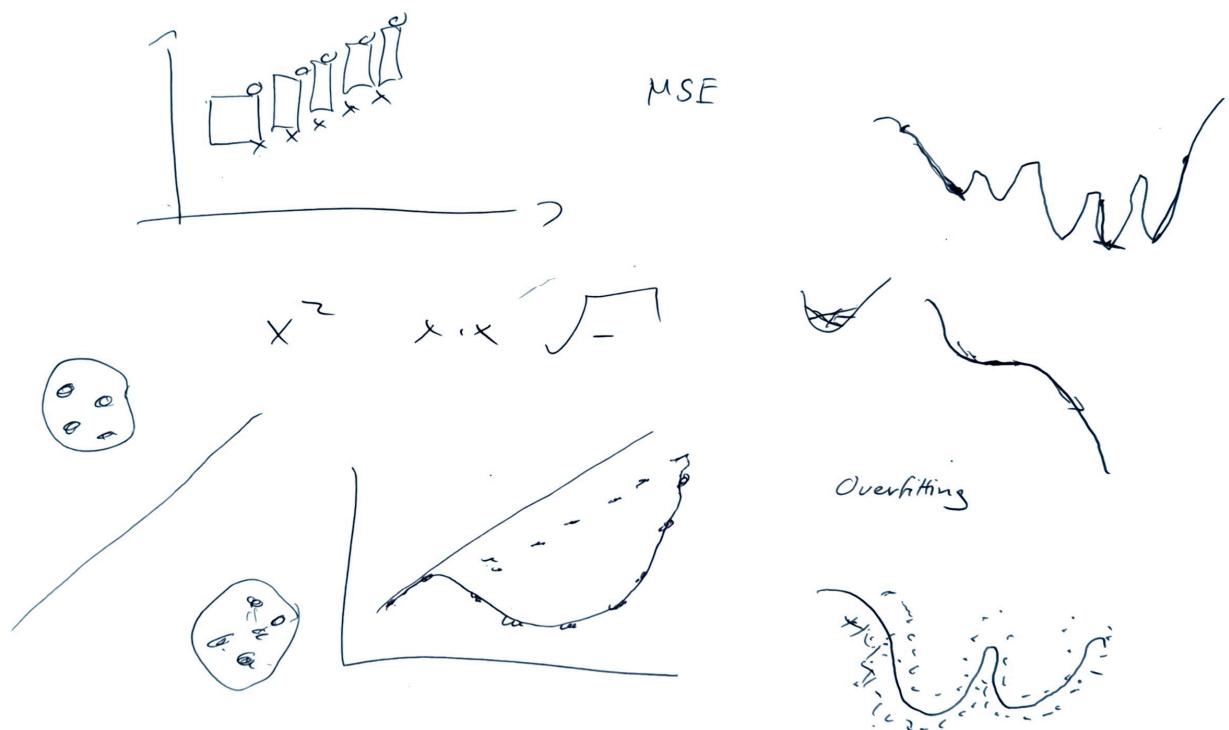


Abb. 4A: Skizze von YOLO und maschinellem Lernen: Das ideale Modelltraining zielt darauf ab, den globalen Tiefpunkt der Verlustfunktion (Loss-Funktion) zu finden, anstatt in lokalen Minima stecken zu bleiben (Manuel, 2024).

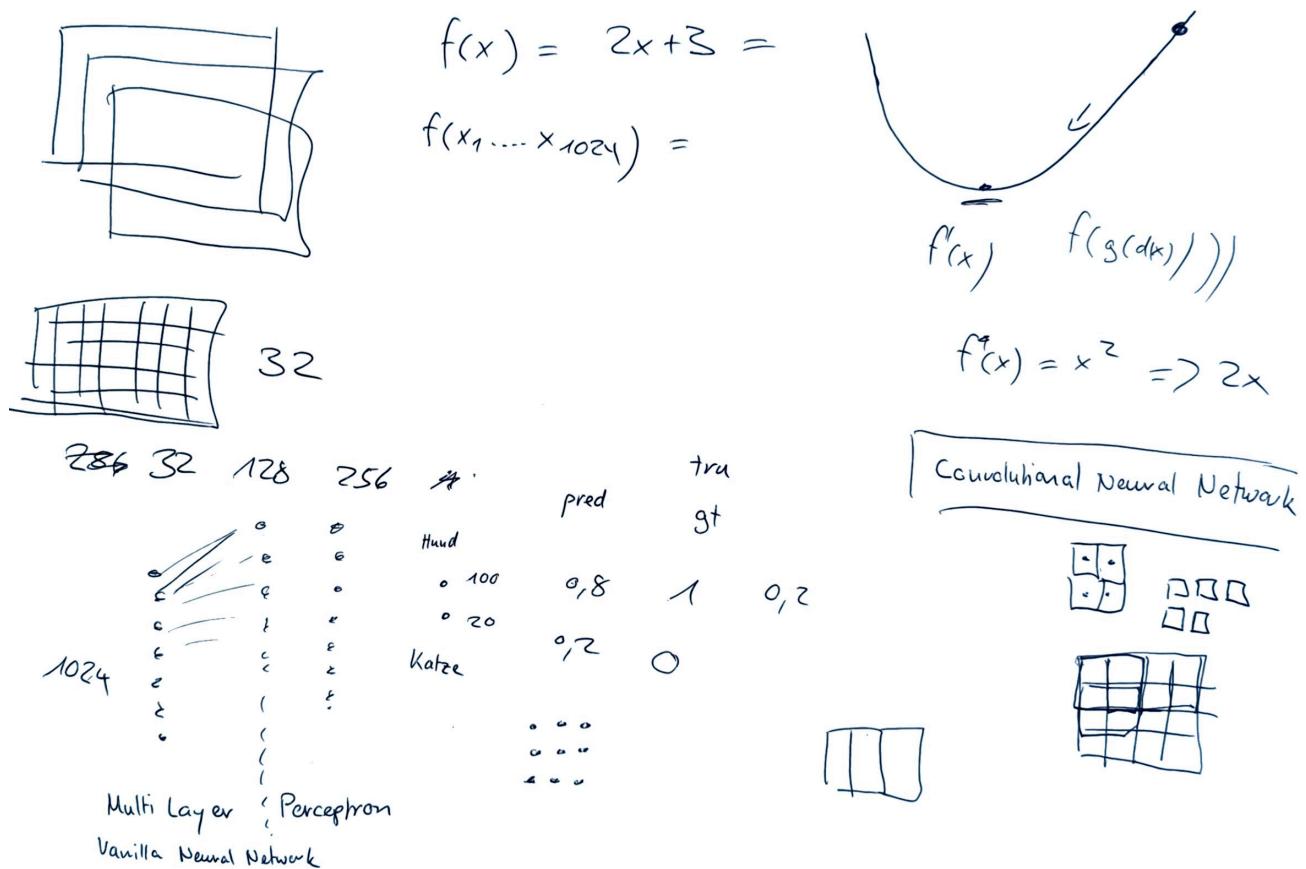


Abb. 5A: Grundlage, wie ein neuronales Netzwerk bei YOLO aussehen kann (Manuel, 2024).

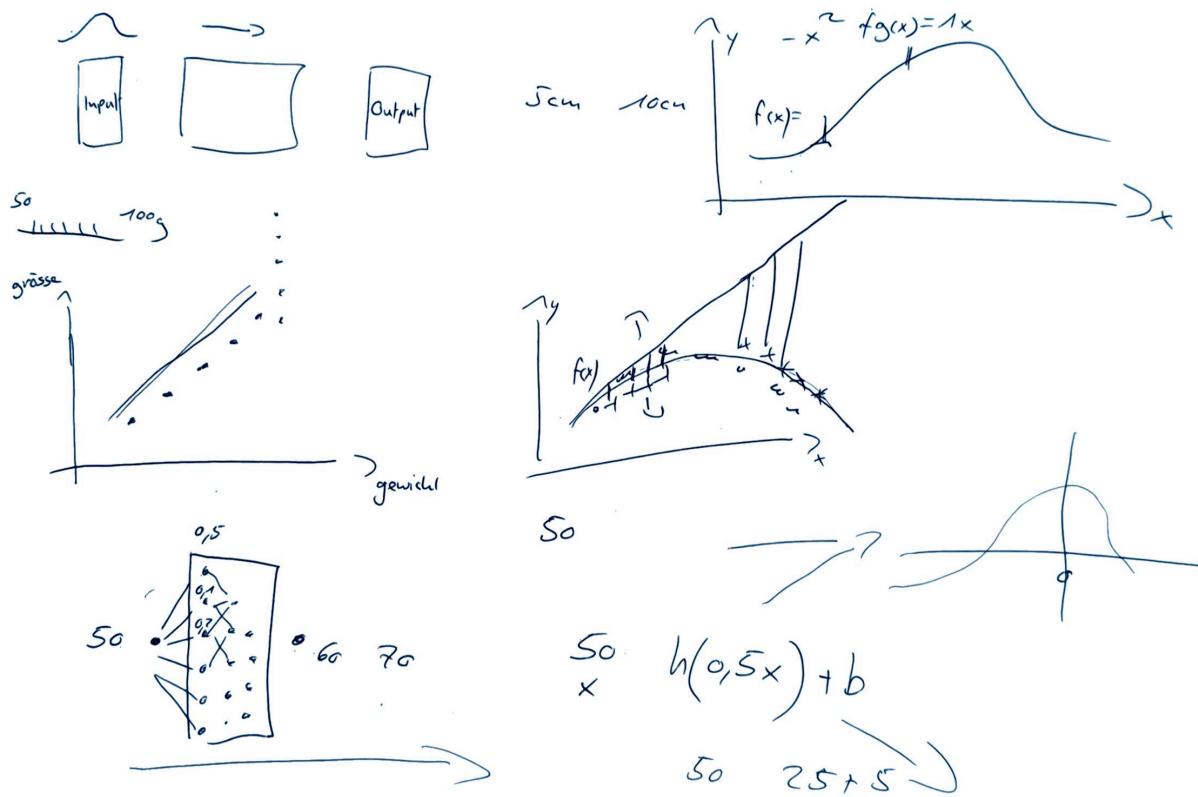


Abb. 6A: Grundlage, wie ein neuronales Netzwerk bei YOLO aussehen kann (Manuel, 2024).

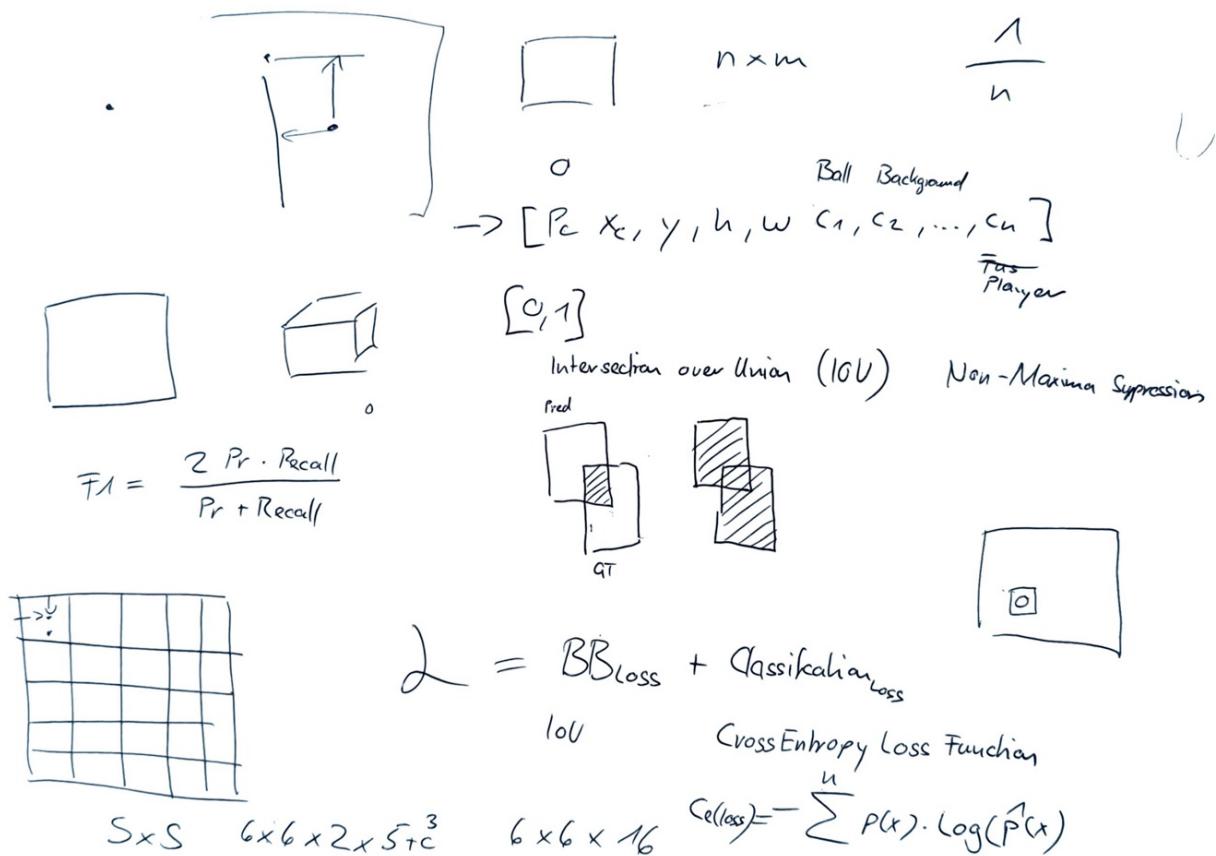


Abb. 7A: Ansicht, wie die Neuronenparameter nach jedem Epoche-Durchlauf angepasst werden, um die Modellleistung zu optimieren. Dieser Prozess erfolgt durch die Anpassung der Gewichte basierend auf dem Fehler, der während des Trainings berechnet wird (Manuel, 2024).

Tabellarische Liste des durchgeführten Feldtestes

Spielnummer	Maschine Tore	Menschliche Tore	Ergebnis	Gewinnchance (%)	Tor-Ratio (Maschine:Mensch)
	2	5	Verloren	4,255319149	0,4
	1	5	Verloren		0,2
	4	5	Verloren		0,8
	3	5	Verloren		0,6
	3	5	Verloren		0,6
	4	5	Verloren		0,8
	1	5	Verloren		0,2
	1	5	Verloren		0,2
	1	5	Verloren		0,2
	1	5	Verloren		0,2
	0	5	Verloren		0
	0	5	Verloren		0
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	3	5	Verloren		0,6
	2	5	Verloren		0,4
	0	5	Verloren		0
	5	4	Gewonnen		1,25
	1	5	Verloren		0,2
	3	5	Verloren		0,6
	2	5	Verloren		0,4
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	3	5	Verloren		0,6
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	3	5	Verloren		0,6
	0	5	Verloren		0
	5	4	Gewonnen		1,25
	3	5	Verloren		0,6
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	4	5	Verloren		0,8
	0	5	Verloren		0
	4	5	Verloren		0,8
	3	5	Verloren		0,6
	0	5	Verloren		0
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	3	5	Verloren		0,6
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	0	5	Verloren		0
	4	5	Verloren		0,8
	3	5	Verloren		0,6
	2	5	Verloren		0,4
	1	5	Verloren		0,2
	1	5	Verloren		0,2

Abb.8A: Tabellarisch aufgelistete Werte des Feldtestes.

Bescheinigung

Name: Sy Viet Dao, Timoniel Perez Vargas _____ Klasse: BGB21c _____

Hiermit bestätige ich, die vorliegende Berufsmaturitätsarbeit mit dem Titel
«Automatisierter Tischkicker»

selbst verfasst zu haben. Informationen aus fremden Quellen sind stets durch die entsprechenden Angaben (Zitate, Quellenverzeichnis) gekennzeichnet.

Ort und Datum: _____

Unterschrift: _____