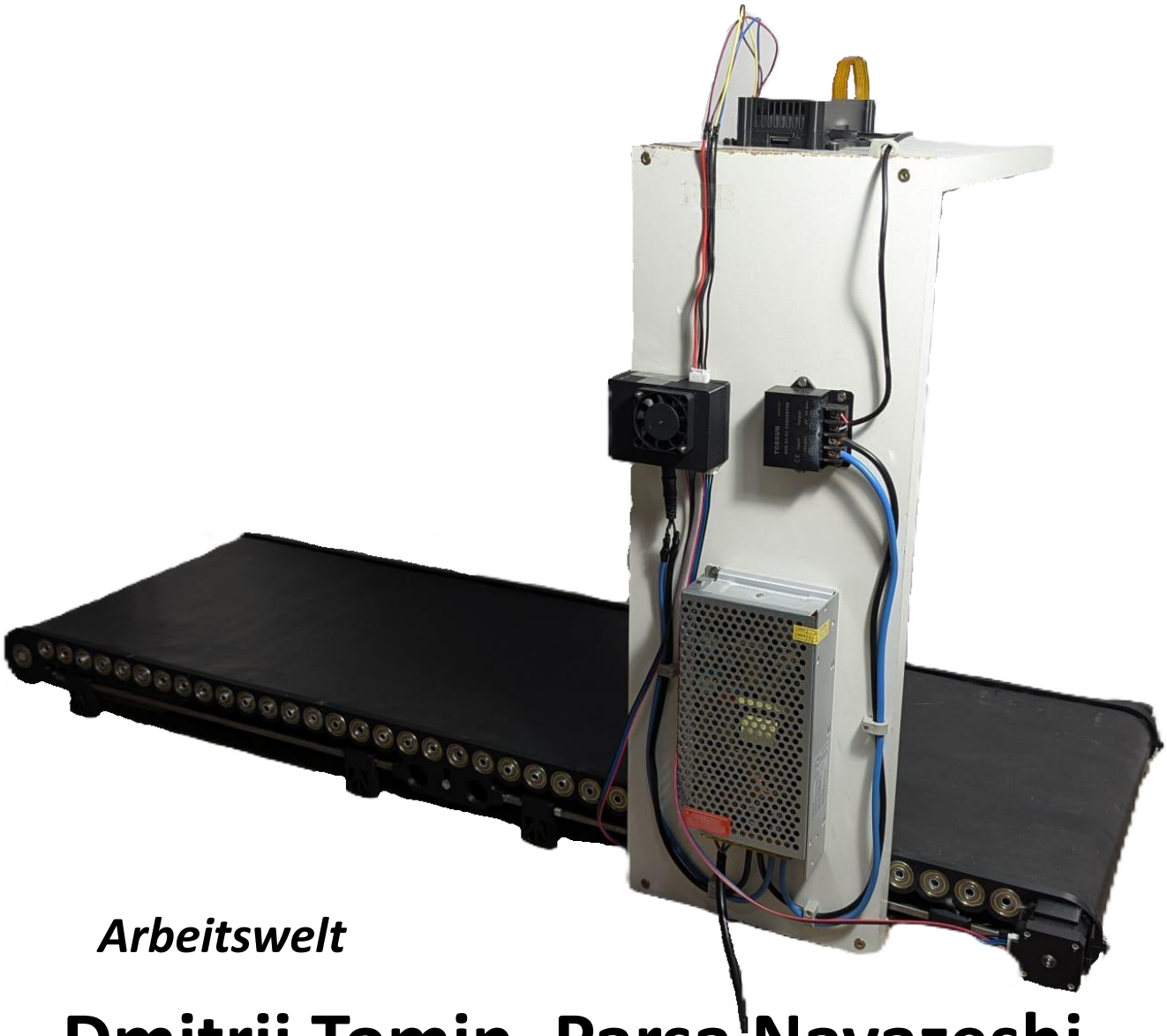
SmartScan

Ein innovatives Selbstbedienungskassensystem



Arbeitswelt

Dmitrii Tomin, Parsa Navazeshi

Kurzfassung

Unser Ziel war es, eine Selbst-Checkout-Kasse zu entwickeln, die mithilfe von Computer Vision auf Kassierer sowie herkömmliche Barcodes verzichten kann.

Dafür nutzten wir YOLOv8 (You only look once), eine Architektur für Echtzeit-Objekterkennung, um die KI mithilfe eines Datensets der Produkte zu trainieren, wobei deren visuelle Merkmale identifiziert werden. Anschließend haben wir ein eigenes Förderband gebaut, auf dem die Produkte erfasst werden, und ließen das trainierte Modell auf einem Raspberry Pi 5, einem Mikrocomputer, laufen, der die Produktbilder, die, mithilfe einer Kamera aufgenommen werden, verarbeitet und die Erkennung durchführt.

Unser Ergebnis zeigt, dass unsere Kasse die Produkte zuverlässig erkennt und den Kassiervorgang automatisch durchführt. So haben wir eine einfache und effiziente Lösung entwickelt, die im Einzelhandel genutzt werden könnte.

Inhaltsverzeichnis

Einleitung

Unser Projekt entstand aus der Idee, eine Selbst-Checkout-Kasse zu entwickeln, die ohne Kassierer und Barcodes auskommt. Die Digitalisierung und der Wunsch nach effizienteren Einkaufserlebnissen treiben den Einzelhandel zunehmend zu innovativen Lösungen. Ein Beispiel, das uns inspiriert hat, ist Amazon Go. Diese Geschäfte setzen auf Computer Vision, um den gesamten Kassiervorgang zu automatisieren. Kunden können einfach einkaufen und gehen, während die Technologie automatisch erkennt, welche Produkte sie nehmen. In der Praxis zeigte sich jedoch, dass Amazon Go in den Anfangsjahren auf menschliche Arbeitskräfte angewiesen war, die das System überwachten und bei Problemen halfen, etwa bei der fehlerhaften Identifikation von Produkten. So entstand ein Kompromiss zwischen der fortschrittlichen Technologie und den traditionellen Kassierern, die nach wie vor eine Rolle spielten.

Unser Ziel war es, dieses Konzept weiterzuentwickeln und ein System zu schaffen, das ohne Barcodes und Kassierer funktioniert. Wir haben YOLOv8, eine Technologie für Echtzeit-Objekterkennung, genutzt, um die Produkte zu erkennen. Unser Ansatz war es, ein System zu entwickeln, das in einem realen Einzelhandelsumfeld zuverlässig und schnell funktioniert, ohne dass man auf die traditionellen Kassen angewiesen ist.

Zu Beginn unseres Projekts wollten wir eine Lösung finden, die den Kassiervorgang schneller und effizienter macht, indem sie die Notwendigkeit für Barcodes und Kassierer überflüssig macht. Wir hofften, dass unser System die Produkte unter verschiedenen Bedingungen genau erkennen und den gesamten Einkauf automatisch abwickeln könnte. Dabei war unser Ziel, eine einfache und praktische Lösung für den „Markt der Zukunft“ zu entwickeln, die mit moderner Technologie arbeitet und den Einkauf für die Kunden erleichtert.

Vorgehensweise, Materialien und Methode

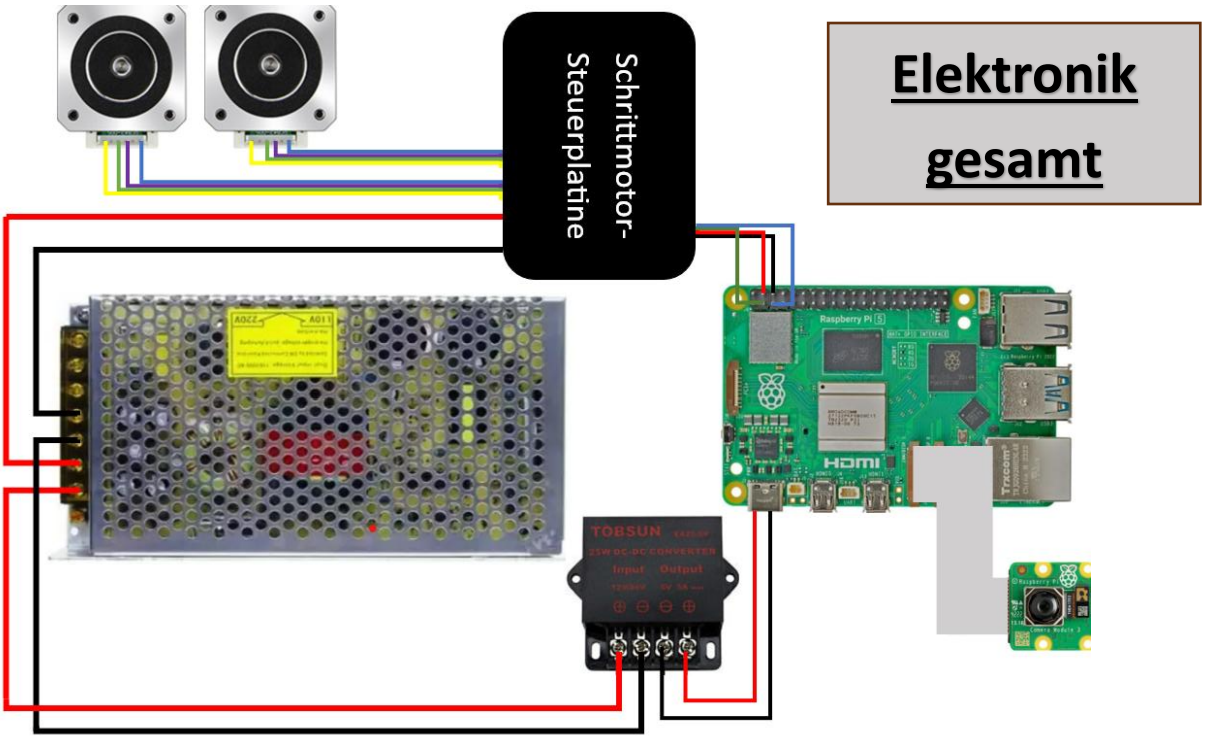
KI

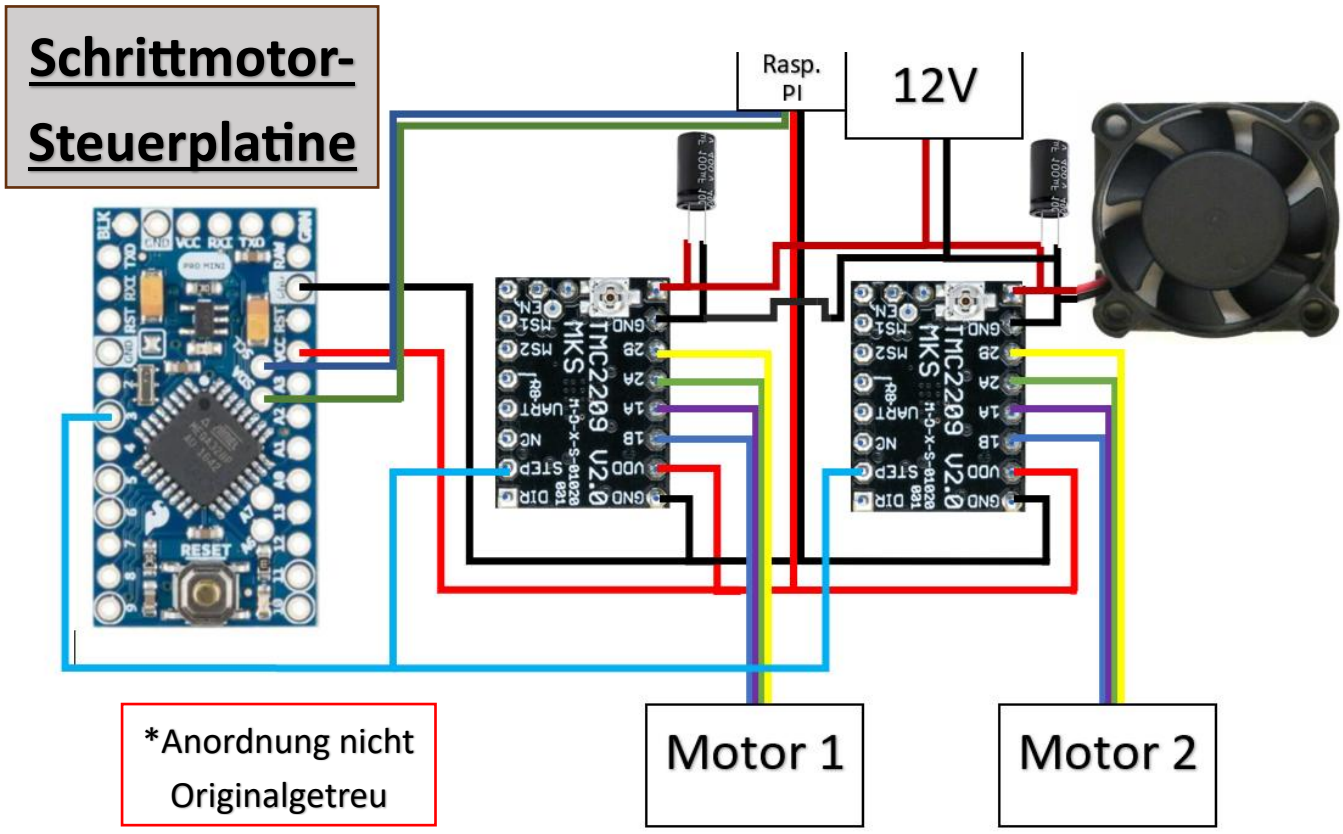
Ein technisches Ziel unseres Projekts war die Entwicklung einer Selbst-Checkout-Kasse, die ohne Kassierer und herkömmliche Barcodes auskommt und sich für den Einsatz im „Markt der Zukunft“ eignet. Dazu setzten wir YOLOv8 ein, eine Architektur für Echtzeit-Objekterkennung, die auf den Prinzipien eines Convolutional Neural Networks (CNN) basiert. Ein CNN analysiert ein Bild und extrahiert automatisch relevante Merkmale. In den Faltungsschichten (Convolution Layers) werden kleine Filter, auch Kernel genannt (z. B. 3x3 oder 5x5), über das Bild geschoben, um Merkmale wie Kanten, Formen oder Texturen zu erkennen. Diese Informationen werden in sogenannten Feature Maps gespeichert. Anschließend reduzieren Pooling-Schichten die Datenmenge, indem sie die wichtigsten Merkmale beibehalten. Dies macht das Netzwerk effizienter und robuster. Am Ende werden die extrahierten Merkmale durch vollständig verbundene Schichten verarbeitet, die das Bild in Klassen wie „Milch“ oder „Eier“ einteilen.

Als ersten Schritt erstellten wir ein Datenset mit Produktbildern, wobei wir die Produkte aus verschiedenen Blickwinkeln fotografierten. Dadurch wurde sichergestellt, dass die Position der Produkte auf dem Förderband, welches wir zum Imitieren von echten Bedingungen später gebaut haben, keinen Einfluss auf die Erkennung hat und der Scanprozess reibungslos verläuft. Danach trainierten wir das neuronale Netzwerk mit YOLOv8. Dabei nutzten wir das vortrainierte Modell „yolov8n“, wobei das „n“ für Nano steht, die kleinste und effizienteste Variante. Diese Wahl reduzierte die benötigte Rechenleistung und steigerte die Geschwindigkeit der Erkennung. Um die durch das kleinere Modell möglicherweise verringerte Genauigkeit auszugleichen, erhöhten wir die Anzahl der Trainingsdurchläufe auf 100. Anschließend exportierten wir das Modell und konvertierten es ins NCNN-Format, um die maximale Leistung auf dem Raspberry Pi 5 zu gewährleisten.

Prüfstand

Als nächstes bauten wir ein Prüfstand, das Förderband, um sicherzustellen, dass unsere Idee auch unter Echt-Welt-Bedingungen funktioniert. Dafür haben wir ein Förderband in Fusion 360, einem CAD-Programm, konzipiert und es anschließend 3D-gedruckt. Nachdem wir das Förderband zusammengebaut haben, haben wir uns um die Elektronik gekümmert.





Python script

Als letztes haben wir ein Python script geschrieben, mithilfe dessen jedes Produkt gezählt und der dazugehörige Preis zugeordnet wird. Dabei mussten wir ein Problem lösen und zwar sollten die Produkte nur einmal gezählt und danach ignoriert werden. Das haben wir gelöst, indem wir das erkannte Produkt einfach nicht mitzählten, wenn die X-Koordinaten dessen hochgehen, d.h., wenn das Produkt im nächsten Bild weiter ist, als im vorherigen, wird es ignoriert. Wenn aber das Produkt im nächsten Bild sich „zurückbewegt“ hat, was passiert, wenn ein neues ins Bild kommt, wird es gezählt.

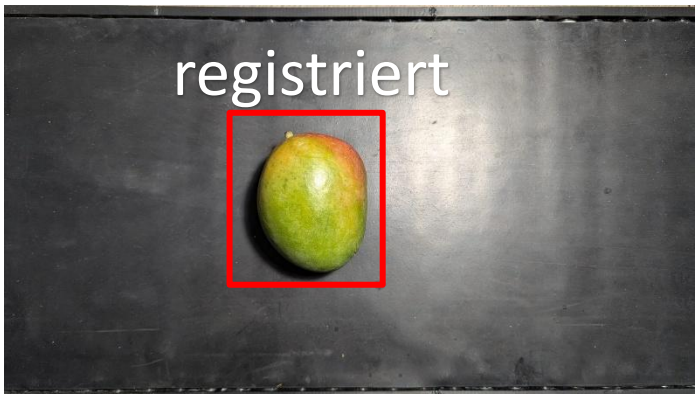


Bild 1

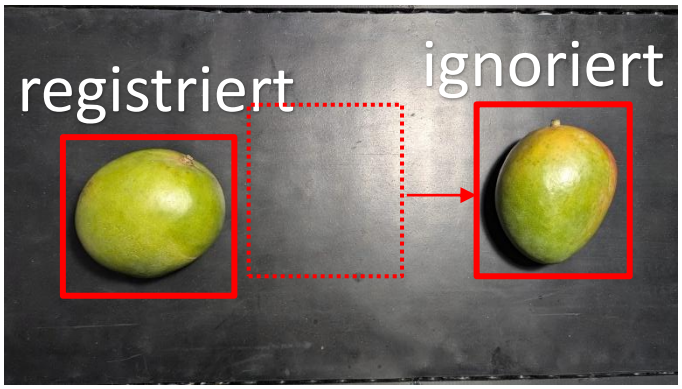


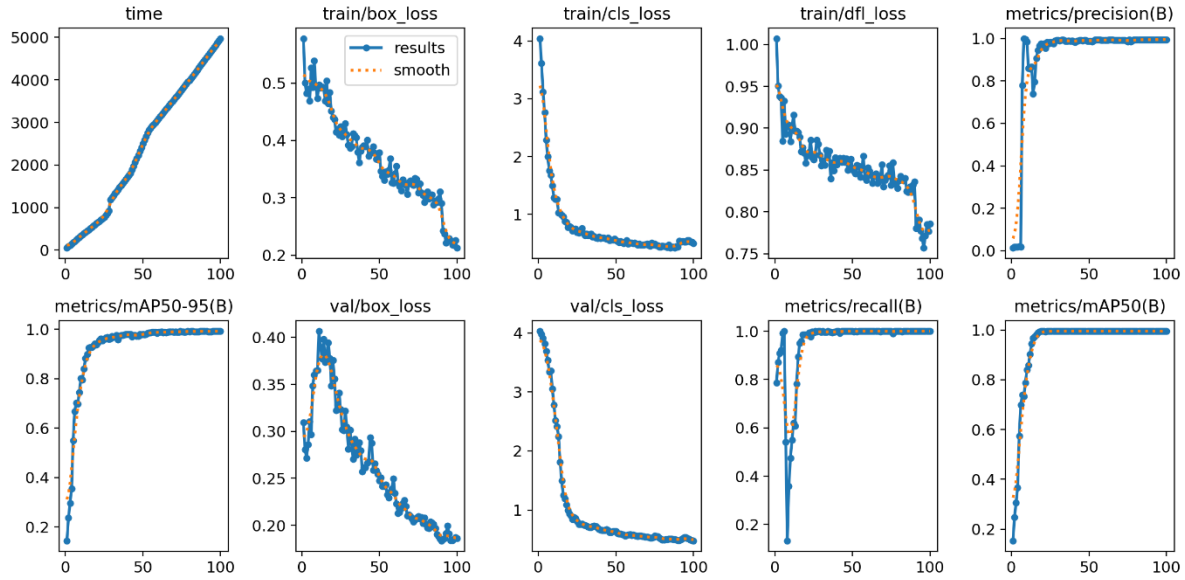
Bild 2

Damit es zu keinen Fehlerkennungen kommt, wenn das Produkt nur zur Hälfte im Bild ist, werden die Produkte erst ab einer virtueller Linie gezählt.

Wir haben keine externe Unterstützung erhalten. Unsere Arbeit haben wir hauptsächlich zu Hause durchgeführt, und zwar an jedem Wochenende sowie während der Ferien, jeweils in kleineren Einheiten.

Ergebnisse

Trainingsergebnisse



Das sind die Ergebnisse des trainierten Modells. Die train/loss Graphen zeigen, wie gut das Modell die Trainingsdaten lernt, während die val/loss Graphen veranschaulichen, wie gut das Modell auf unbekannten Daten performt. Ein sinkender train/loss Graph bedeutet, dass das Modell seine Parameter an die Trainingsdaten anpasst, während ein sinkender val/loss anzeigt, dass das Modell auch auf neuen Daten gut funktioniert. Da die train/loss und val/loss Graphen keine großen aufweisen, liegt kein Problem des Overfittings vor. Das heißt, dass das Modell generalisiert und nicht auf die Trainingsdaten abgestimmt ist.
Der mAP50-95 Graph zeigt die Präzision des Modells und wie zu erkennen flacht der Graph bei nahezu 1.0 ab, was auf einer sehr hohe Genauigkeit des Modells deutet. Der Endwert des recall Graphen liegt ebenfalls bei 1.0, was auf 0 übersehene Objekte hindeutet.
Diese Schlüsse aus den Graphen lassen sich auch bei Praxistests bestätigen.

Wie man erkennen kann, kann das Modell sogar nur anhand von Farben die Produkte mit einer Mindestsicherheit von 90% erkennen.

Mögliche Problembehebungen

KI-Beschleuniger

Das System ist stark durch den Raspberry Pi limitiert, da obwohl es ein sehr leitungsstarker, günstiger und zugleich kleiner Mikrocomputer ist, ist dieser nicht auf KI und Computer Vision spezialisiert. Ein KI-Beschleuniger oder auch als NPU (neuronale Verarbeitungseinheit) bekannt wäre eine passende Lösung, aufgrund von folgenden Punkten:

Geschwindigkeit

KI-Beschleuniger sind aufgrund ihrer deutlich geringeren Latenz, einem Maß für Verzögerungen in einem System, viel schneller als herkömmliche CPUs. Eine geringe Latenz ist besonders wichtig bei z.B. der Entwicklung von Computer Vision-Anwendungen

Effizienz

KI-Beschleuniger können hundert- bis tausendmal effizienter sein als andere, eher standardmäßige Rechensysteme. Sowohl die großen KI-Beschleunigerchips, die in Rechenzentren verwendet werden, als auch die kleineren, die typischerweise in Edge-Geräten zum Einsatz kommen, verbrauchen weniger Strom und geben weniger Wärme ab als ihre Vorgänger.

Design

KI-Beschleuniger verfügen über eine sogenannte heterogene Architektur, die es mehreren Prozessoren ermöglicht, separate Aufgaben zu unterstützen. Diese Fähigkeit erhöht die Rechenleistung auf das für Computer Vision-Anwendungen erforderliche Niveau.

Der einzige Nachteil ist der Preis. In unserem Falle würde ein KI-beschleuniger mehr als der Raspberry Pi kosten.

Durch einen KI-beschleuniger wäre es möglich, mehr FPS zu erreichen, wodurch zu einem die Geschwindigkeit des Förderbands erhöht werden könnte, was zum schnellerem Abkassieren führen würde. Außerdem könnte man ein komplexeres und schwereres Modell verwenden (anstatt das nano z.B. das large), wodurch auch die kleinsten Merkmale identifiziert werden könnten.

Einschränkung des Konzepts

Die größte Einschränkung ist aber auch das, was dieses System ausmacht, und zwar die Erkennung der Gegenstände anhand visueller Merkmale. Das bedeutet, dass unser System 2 fast identisch aussehende Produkte nicht zuverlässig voneinander unterscheiden kann, wie es beispielsweise bei Obst der gleichen Sorte der Fall ist. Eine einfache Lösung hierfür wäre es den Kunden, beim Beispiel Apfel, die Auswahl zwischen den im Supermarkt erhältlichen Sorten zu geben, was aber ein bisschen den Sinn des Konzepts zerstört, da eigentlich keine bis fast keine menschliche Interaktion erforderlich sein soll.

Große Produkte

Bei großen Gegenständen, die über das Sichtfeld der Kamera gehen, würde man bei dem Trainieren der KI ein herausstechendes Merkmal aussuchen und anhand dessen das Modell trainieren.

Ergebnisdiskussion

Abgesehen von den Problemlösungen haben wir noch ein paar Ideen, wie man auf der Idee weiter aufbauen kann.

Eine Wägezelle in das Förderband wäre eine davon. Durch eine Wägezelle könnte man nicht nur den Stückpreis zählen, sondern auch den Preis pro Kilogramm, was den Anwendungsbereich erweitert. Zusätzlich könnte eine eingebaute Waage Diebstahl vorbeugen, indem das Programm den Messwert mit dem bekannten Produktgewicht vergleicht und wenn es deutlich abweicht, ein Mitarbeiter ruft, dadurch wäre so eine Kasse wohlmöglich diebstahlsicherer als eine herkömmliche Selbstbedingungskasse.

Eine weitere Idee ist es, eine abgewandelte Version dieses Systems als Hilfe und nicht als Ersatz für Kassierer zu verwenden.
Durch den Einbau einer kleinen Kamera neben den Barcode-Scannern könnte man das Scannen von Obst, Backwaren und anderen Produkten ohne Etikett produktiver gestalten. Man würde z. B. Äpfel auf die bereits eingebaute Waage vor dem Kassierer stellen, die Kamera würde das Produkt erfassen und zusammen mit dem Gewicht den Preis berechnen.

Zusammenfassung

Unser Ziel, eine Selbst-Checkout-Kasse zu entwickeln, die ohne Kassierer und Barcodes auskommt, haben wir erfolgreich erreicht.

Das System erkennt die Produkte zuverlässig und ermöglicht einen automatisierten Kassiervorgang, jedoch nicht ohne Herausforderungen, wie die begrenzte Rechenleistung des Raspberry Pi, und Nachteile, wie die begrenzte Einsetzbarkeit, aufgrund der Schwierigkeit, sehr ähnliche Produkte zuverlässig zu unterscheiden, etwa bei Obstsorten oder ähnlichen Verpackungen.

Trotz dieser Herausforderungen zeigt das Projekt, dass die Idee eines automatisierten, barcodefreien Selbst-Checkout-Systems grundsätzlich funktioniert und großes Potenzial für die Zukunft des Einzelhandels bietet.

Quellen-und Literaturverzeichnis

https://www.amazon.com/b?ie=UTF8&node=16008589011, 3.02.2025, Amazon.com, Inc., Amazon Go

https://docs.ultralytics.com/de/guides/raspberry-pi/#comparison-chart, 11.02.2025, Ultralytics Team, Yolov8 mit Raspberry Pi

https://docs.ultralytics.com/de/models/yolov8/#performance-metrics, 11.02.2025, Ultralytics Team, Was ist Yolov8

https://www.ibm.com/de-de/think/topics/ai-accelerator, 13.02.2025, IBM Corporation, Vorteile von KI-Beschleunigern

https://www.ibm.com/think/topics/convolutional-neural-networks, 13.02.2025, IBM Corporation, Was ist ein CNN