



Mühlbauer

High Tech International

PROJEKTDOKUMENTATION ZUR ABSCHLUSSPRÜFUNG SOMMER 2025

BEWEGUNGSERKENNUNG UND BILDFILTERUNG ZUR
VERBESSERUNG VON OBJEKT- UND PERSONENERKENNUNG

Prüfling: Claus Krzyzek

Fachinformatiker für Anwendungsentwicklung

Prüflingsnummer: 95081

E-Mail: claus.krzyzek@muehlbauer.de

Telefon: +49 9461 952 18013

Projektbetreuer: Salih Uzundal

E-Mail: salih.uzundal@muehlbauer.de

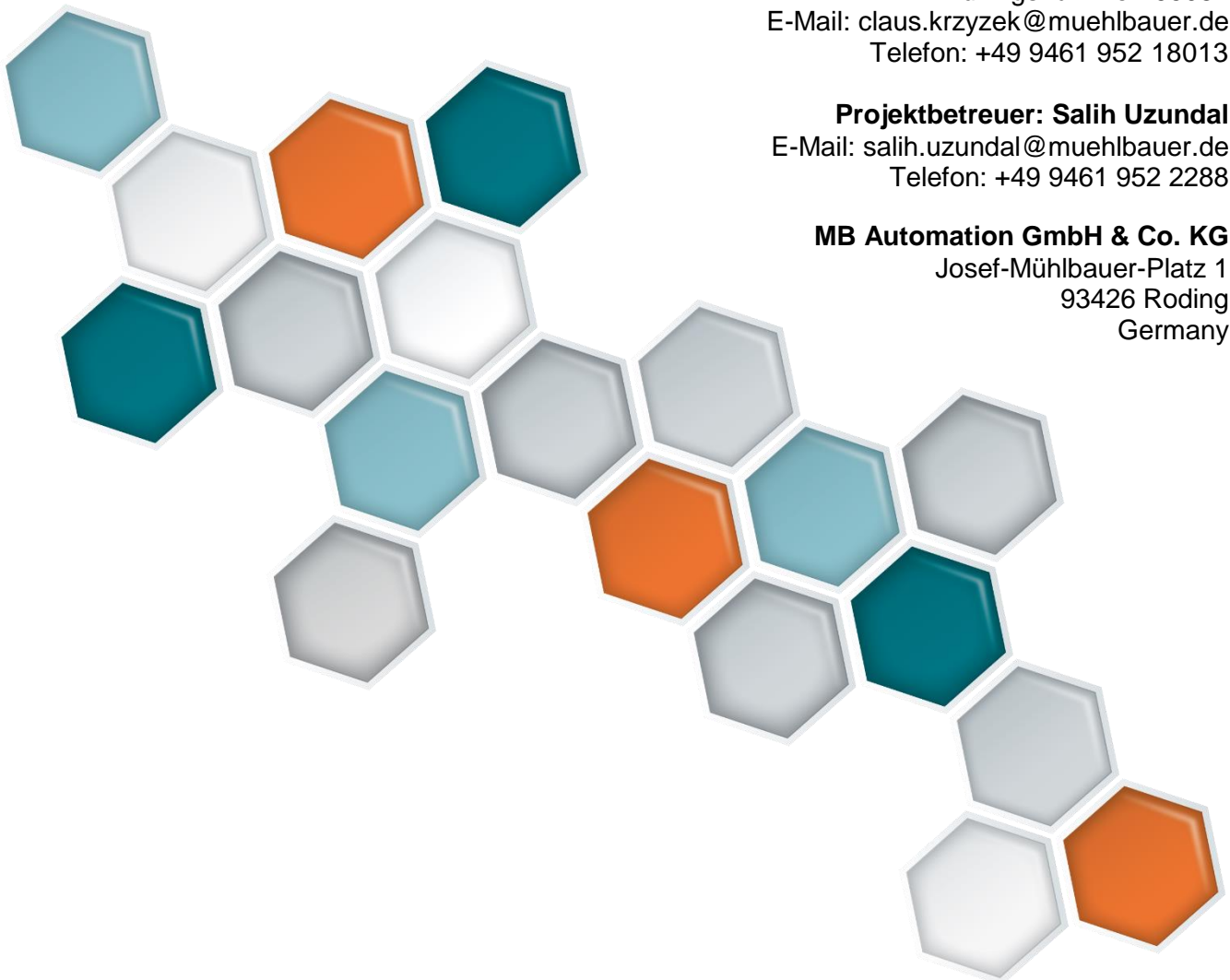
Telefon: +49 9461 952 2288

MB Automation GmbH & Co. KG

Josef-Mühlbauer-Platz 1

93426 Roding

Germany



Copyright © 2025 Mühlbauer Group
All rights reserved

No part of this document may be reproduced or transmitted in any form or by any means, mechanical or electronic, including recording, photocopying, storing in a retrieval system, for any purpose other than the personal use, without the prior written permission of Mühlbauer Group. The information given in this document is subject to change without notice. Any software described in this document is subject to license agreement or NDA (non-disclosure agreement) and may be copied and/or used only in accordance with the conditions of the agreement.

Brand and product names and marks are included in this document and are trademarks or registered trademarks of their respective holders.

The authors of this document and the publisher have taken care in the preparation of this document. The accuracy, completeness or timeliness of such information cannot be guaranteed. Mühlbauer Group does not make expressed or implied warranty of any kind and assumes no responsibility for errors or omissions. No guarantee and no liability are assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs given in this document.

INHALTSVERZEICHNIS

1. Einleitung	5
1.1 Anmerkung	5
1.2 Mühlbauer Group	5
1.3 MB AI for Image Enhancement	5
2. Projektdefinition	6
2.1 Ist-Analyse	6
2.2 Soll-Konzept	6
3. Projektbegründung	7
3.1 Zeitlicher Rahmen	7
3.2 Wirtschaftlichkeit	8
3.2.1 Kostenplanung	8
3.2.2 Amortisierung	8
3.3 Ressourcenplanung	9
4. Planung und Ablauf des Projekts	9
4.1 Einarbeitung in das bestehende Projekt	9
4.2 Recherche zu verfügbaren Technologien	9
4.3 Erstellen eines Pflichtenheftes	10
4.4 Entwurf der Architektur	10
4.5 Entwicklungsumgebung	11
5. Projektdurchführung	11
5.1 Entwicklung der Bilddifferenzierung (Bildsubtraktion)	11
5.2 Implementierung der Bildfilterung zur Rauschreduzierung	11
5.3 Integration von Clustering-Algorithmen	13
5.4 Einbindung in die KI	14
6. Qualitätssicherung	14
6.1 Whitebox- Tests	14
6.2 Blackbox- Tests	14
7. Projektabschluss	16
7.1 Dokumentation	16
7.2 Übergabe	16
7.3 Schulung	16
7.4 Soll- Ist- Vergleich	17
7.5 Fazit	17
7.6 Ausblick	18
8. Verzeichnisse	18
8.1 Glossar	18

8.2 Abbildungsverzeichnis

8.3 Tabellenverzeichnis

19

19

1. Einleitung

1.1 Anmerkung

Die folgende Projektdokumentation wurde im Rahmen des IHK-Abschlussprojektes für die Ausbildung zum *Fachinformatiker für Anwendungsentwicklung* erstellt und schildert den Ablauf des Abschlussprojektes. Die Gegebenheit des Abschlussprojekts bezieht den Gebrauch von Fachsprache ein. Zusätzlich werden einige Begriffe genannt, die spezifisch zum Ablauf von Handlungsschritten dienen. Alle Begriffe können in einem angehängten Glossar (siehe 8.1) eingesehen werden. Beschriebene Wörter werden im Text mit einem fetten Schriftschnitt gekennzeichnet.

1.2 Mühlbauer Group

Die Mühlbauer Group hat sich seit ihrer Gründung im Jahr 1981 zu einem weltweit führenden Hightech-Unternehmen entwickelt. Als globaler Technologiepartner bietet Mühlbauer umfassende Lösungen in der Herstellung und Personalisierung von hochsicheren ID-Dokumenten wie ePässen, ID-Karten und Führerscheinen sowie in den Bereichen der RFID-Technologien, Smart Cards, im Solar-Backend und der Halbleiterindustrie. Ergänzt wird das Portfolio durch moderne Grenzkontrollsysteme und biometrische Sicherheitslösungen, die weltweit zur Anwendung kommen. Bis 2025 konnte die Mühlbauer Group eine zunehmende Marktakzeptanz und steigende Marktausweitung verzeichnen. Mit circa 3500 Mitarbeitern an global 35 Standorten bietet das Unternehmen ein weitreichendes Vertriebs- und Servicenetzwerk.

1.3 MB AI for Image Enhancement

Die Abteilung „MB AI for Image Enhancement“ der Mühlbauer ID Services GmbH ist auf Bildverarbeitung sowie die Lokalisierung und die Identifizierung von Personen mithilfe Künstlicher Intelligenz in Echtzeit spezialisiert. Das Aufgabengebiet erstreckt sich von der Neu- und Weiterentwicklung bis hin zur Wartung von bestehenden Lösungen im Produktportfolio. Produkte werden hauptsächlich in **C++** und **Python** für **Windows** entwickelt. Zurzeit ist die Hauptaufgabe der Abteilung die Entwicklung des „Smart Seamless Corridor“-Projektes. Mit diesem soll eine automatische Personenkontrolle ohne manuelle Überprüfung von Ausweis- oder Reisedokumenten an Kontrollpunkten wie dem Check-Out an Flughäfen ermöglicht werden. Das Alleinstellungsmerkmal dieses Produktes ist es, dass Reisende hierfür nicht stehen bleiben müssen, sondern einfach durch die Kontrolle laufen. Hierfür durchqueren Reisende beziehungsweise zu kontrollierende Personen einen speziellen Korridor. Mithilfe mehrerer Kameras werden Personen in diesem Bereich erfasst und durch eine künstliche Intelligenz geortet und mit hinterlegten Daten abgeglichen. Identifizierte und zugangsberechtigte Personen müssen so weder anhalten noch Ausweis- oder Reisedokumente vorzeigen während sie durch diese Kontrolle gehen. Dies führt zu einer erheblichen Einsparung an Reisezeit, Sicherheitspersonal und damit verbundenen Kosten. Bei nicht identifizierten Personen oder sogenannten „Persons of Interest“ wird menschliches Personal informiert, das angemessene Maßnahmen ergreifen kann.

2. Projektdefinition

2.1 Ist-Analyse

Objekt- und Personenerkennung geht mit einer Vielzahl an technischen Herausforderungen und gerade zu dem genannten Einsatzzweck mit speziellen Sicherheitsanforderungen einher. Im „**Smart Seamless Corridor**“ ist unter anderem eine Unterscheidung von stationären als auch sich durch den Korridor bewegendenden Objekten notwendig. Nach aktuellem Stand des Produktes kann beispielsweise nicht zwischen einem Rucksack, der von einem Reisenden durch den Korridor getragen wird und einem Rucksack, der auf dem Boden liegen gelassen wurde, unterschieden werden. Eine solche Funktionalität ist aber gerade bei einem Einsatz in sicherheitskritischen Bereichen wie Flughäfen von großer Bedeutung. Weiterhin überprüft das aktuelle System sämtliche in der Videoübertragung enthaltenen Bildinformationen, was sich negativ auf die Effizienz als auch auf die Qualität der Erkennung von Objekten und Personen auswirkt.

2.2 Soll-Konzept

Ziel des Projektes ist es, das **Backend** dieses Systems um ein Tool zu erweitern, das aufeinanderfolgende Bilder einer Videoübertragung durch gegenseitige **Subtraktion** auf Unterschiede und damit Bewegungen überprüfen und diese darstellen soll. Ein weiteres Ziel des Projektes ist es, Bildrauschen in diesen so entstandenen Bildern, das zum Beispiel durch sich verändernde Lichtverhältnisse entstehen kann, zu reduzieren. Weiterhin sollen zueinander gehörende Pixel durch verschiedene Bildbearbeitungsschritte einander zugeordnet (**geclustert**) werden. Es sollen während der Entwicklung verschiedene Algorithmen bezüglich der Bildqualität getestet werden. Das betrifft sowohl die Bildbereinigung als auch das Clustering. Die so bearbeiteten Bilder sollen einer Künstlichen Intelligenz im Zuge von Objekterkennungen übergeben werden. Die Auswirkung auf die Qualität der Objekterkennungen ist sowohl bei sich bewegendenden als auch bei stationären Objekten zu testen. Einzubeziehen sind verschiedene Lichtverhältnisse sowie sich bewegendende Objekte und Personen in unterschiedlicher Anzahl und Entfernung zueinander.

3. Projektbegründung

3.1 Zeitlicher Rahmen

Die Soll-Stunden wurden durch eine Analyse der Aufgabenstellung ermittelt. Die tatsächlich aufgewendeten Ist-Stunden wurden während der Durchführung des Projekts erfasst und anschließend miteinander verglichen.

Vorgang	Stunden
Planungs- und Analysephase	9
Analyse des Ist-Zustands	4
Stakeholder Kommunikation	1
Entwurf des Soll-Konzepts	3
Analyse der Wirtschaftlichkeit	1
Entwurfsphase	16
Einarbeitung in das bestehende Projekt	6
Recherche zu verfügbaren Technologien	5
Erstellen eines Pflichtenheftes	2
Entwurf der Architektur	3
Implementierungsphase	37
Entwicklung der Bilddifferenzierung (Bildsubtraktion)	8
Implementierung der Bildfilterung zur Rauschreduzierung	9
Integration von Clustering-Algorithmen	10
Einbindung in die KI	6
Debugging und Fehlerbehebung	4
Qualitätsmanagement	9
Durchführen von Whitebox-Tests	5
Blackbox-Tests an der Maschine	4
Dokumentation und Schulung	8
Erstellen einer Dokumentation	6
Schulung von Mitarbeitenden	2
Übergabe	1
Gesamt	80

Tabelle 1 - Zeitlicher Rahmen

3.2 Wirtschaftlichkeit

Um die wirtschaftliche Rentabilität des Abschlussprojekts zu analysieren, müssen sowohl die entstandenen Kosten als auch die durch das Projekt gewonnenen Einnahmen beachtet werden. Es ist jedoch zu beachten, dass die im Nachfolgenden genauen Kosten nur als Richtwerte zu verstehen sind, was dem Zweck dienen soll, Betriebsgeheimnisse zu schützen. Aufgrund dessen kann es zu Abweichungen gegenüber den tatsächlichen Werten kommen.

3.2.1 Kostenplanung

Um die durch das Projekt verursachten Fixkosten übersichtlich darzustellen, wurde eine detaillierte Kalkulation erstellt, welche die direkt mit dem Abschlussprojekt in Zusammenhang stehenden Personalkosten auflistet.

Vorgang	Mitarbeiter	Zeit	Stundensatz	Kosten
Umsetzung	1x Auszubildender	80h	20,00 €	1600,00 €
Meeting	2x Mitarbeiter	2h	100,00 €	400,00 €
Testen	2x Mitarbeiter	1h	100,00 €	200,00 €
Schulung	1x Mitarbeiter	2h	100,00 €	200,00 €
Gesamt				2400,00 €

Tabelle 2 - Personalkosten

3.2.2 Amortisierung

Externe Lösungen von Drittanbietern werden in diesem Bereich für gewöhnlich als „Software pay per transaction“ abgerechnet. Sollte die Mühlbauer Group einen vergleichbaren Dienst bei einem Drittanbieter buchen müssen wären folgende Kosten zu erwarten. Bei einer ganzjährigen 24/7-Nutzung, einem Betrag von 0,01 Eurocent pro Transaktion und 30 Bildern pro Sekunde ergibt sich eine Amortisierung nach 24.000.000 Transaktionen beziehungsweise nach 9,26 Tagen ab Inbetriebnahme des ersten Systems.

$$\frac{\text{Gesamtprojektkosten}}{\text{Kostensparnis pro Zeiteinheit}} = \text{Amortisierungsdauer} \rightarrow \frac{2400}{0,0001 \cdot 30 \cdot 60 \cdot 60 \cdot 24} = 9,26$$

$$\frac{\text{Gesamtprojektkosten}}{\text{Kostensparnis pro Zeiteinheit}} = \text{Amortisierungsdauer} \rightarrow \frac{2400}{0,0001 \cdot 30 \cdot 60 \cdot 60 \cdot 24} = 9,26$$

Abbildung 1 - Amortisierungsdauer

Ressourcenplanung

Die nachfolgende Tabelle beinhaltet eine umfassende Auflistung aller erforderlichen Elemente, welche zur Durchführung des Projekts verwendet wurden.

Ressource	Kategorie	Beschreibung
Visual Studio Professional 2019	Software	Entwicklungsumgebung für die Erstellung des Quellcodes der Webanwendung für die Verwaltung der Geräte
Microsoft Word 2019	Software	Textverarbeitungsprogramm zur Erstellung der Dokumentation
Seamless Corridor	Hardware	Metallrahmen mit eingebautem Kamera- und Bildverarbeitungssystem
KI-PC	Hardware	Leistungsstarker PC zum Training von KI-Modellen
Gitlab Server	Hardware	Firmeninterner Gitserver
Claus Krzyzek	Personal	Durchführung des Projekts
Uzundal Hilmi Salih	Personal	Beratung (Ausbilder, Projektbetreuer)
Kästl Nicole	Personal	Beratung & Testen
Gabriel Mercado Rivera	Personal	Beratung & Testen

Tabelle 3 - Ressourcen

4. Planung und Ablauf des Projekts

4.1 Einarbeitung in das bestehende Projekt

Das Seamless-Corridor-Projekt besteht aus mehreren modular erstellten Softwarekomponenten das von einem Kamerasystem übermittelte Bilddaten verarbeitet. Softwareseitig handelt es sich hierbei um ein Produkt, das aus mehreren unabhängig voneinander agierenden Modulen besteht. Bei diesen Modulen handelt es sich um **Microservices** welche untereinander kommunizieren. Das Modul „Camtracker“ ist für die Erfassung und Verfolgung von Personen und Objekten zuständig. Weiter zeigt es die Identität erfasster Personen im Videofeed an. In dieses Modul soll das Projekt letztendlich integriert werden.

4.2 Recherche zu verfügbaren Technologien

Als Programmiersprache für die Entwicklung der Bildfilterung und der Bewegungserkennung wurde sich für **C++** entschieden. Dieser Entschluss fiel aufgrund der Hardwarenähe, der Effizienz und Wartbarkeit der Programmiersprache. Weiterhin ist ausschlaggebend, dass ein Großteil der bildverarbeitenden Funktionen des **Seamless Corridors** bereits in **C++** vorhanden sind und daher abteilungsintern ein großer Erfahrungsschatz diesbezüglich vorliegt. Als **Library** bezüglich Bildverarbeitung wurde sich für **OpenCV** entschieden. Grund hierfür waren die Breite des Spektrums an Funktionen, die umfangreiche Dokumentation und die Optimierung auf Echtzeitanwendungen. Auch mit OpenCV wurde abteilungsintern bereits viele Erfahrungen gesammelt, wodurch zukünftige Einbindungen weiterer Mitarbeiter vereinfacht werden.

Die Wahl von **DBSCAN** als Algorithmus gegenüber anderen Algorithmen für die Clusteranalyse fiel aufgrund mehrerer Vorteile gegenüber den Alternativen. Gerade bei unregelmäßigen Bewegungsmustern bietet DBSCAN im Vergleich zu anderen Verfahren wie K-Means große Vorteile, da es sich besonders gut für nichtlineare und beliebig geformte

Cluster eignet. Die Vor- und Nachteile der verschiedenen Algorithmen zeigt die folgende Tabelle auf.

Algorithmus	Vorteile	Nachteile
DBSCAN	Erkennt beliebig geformte Cluster	Schwierigkeiten bei hochdimensionalen Daten
	Robust gegenüber Rauschen und Ausreißern	Wahl geeigneter Parameter kann herausfordernd sein
	Keine vorherige Angabe der Clusteranzahl erforderlich	
K-Means	Einfach zu implementieren	Clusteranzahl muss im Voraus angegeben werden
		Kann nur kugelförmige Cluster erkennen
Hierarchisches Clustering	Beliebige Clusterformen möglich	Hoher Rechenaufwand bei größeren Datensätzen

Tabelle 4 - Algorithmusvergleich

4.3 Erstellen eines Pflichtenheftes

Das Pflichtenheft wurde unter Berücksichtigung der erforderlichen Anforderungen erstellt. Es bietet detaillierte Informationen darüber, wie der Auftragnehmer beabsichtigt, das Projekt durchzuführen und alle Anforderungen zu erfüllen. Das Pflichtenheft ist als separates Dokument im Anschluss zu finden.

4.4 Entwurf der Architektur

Es wurde ein objektbasierter Architekturentwurf des Projektes angestrebt, die es ermöglichen soll, verschiedene Komponenten der Bewegungserkennung unabhängig voneinander zu entwickeln und zu testen. Durch eine Trennung der Verantwortlichkeiten zwischen diesen sollen außerdem einfachere Wartungen, Anpassungen und Erweiterungen zukünftig ermöglicht werden. Diese Hauptkomponenten umfassen die Erfassung von Bildbeziehungsweise Videodaten, die Bildverarbeitung, die Clusteranalyse und die Visualisierung der Ergebnisse.

Bilderfassung und Vorverarbeitung

Dieses Modul ist verantwortlich für die kontinuierliche Aufnahme eines Videostreams über **OpenCV** und die Vorbereitung dieser Bilddaten für die spätere Analyse. Dazu gehören Konvertierungsprozesse, Bildsubtraktion sowie Rauschunterdrückung durch **morphologische Operationen**.

Merkmalsextraktion

Hier werden die nach der Vorverarbeitung verbliebenen Konturen extrahiert, die als Ausgangspunkt für das Clustering dienen. Weiterhin werden diese Konturen in Vektoren aus Punkten umgewandelt, um eine Bearbeitung durch den Clusteringalgorithmus zu ermöglichen.

Clusteranalyse

Mithilfe des DBSCAN-Algorithmus werden die in den Vektoren enthaltenen Punkte in Cluster eingeteilt.

Visualisierung

Mithilfe von OpenCV-Funktionen werden erkannte Cluster über das Originalbild gelegt, farblich markiert und in Echtzeit ausgegeben.

4.5 Entwicklungsumgebung

Die Bildfilterung und die Bewegungserkennung wurde mithilfe der Entwicklungsumgebung Visual Studio Professional 2019 entwickelt. Als Versionsverwaltung wurde **Git** verwendet. Die Repositories werden auf dem firmeneigenen **Gitlab** Server gehostet.

5. Projektdurchführung

Während der Projektdurchführungsphase wurden die vorher festgelegten Anforderungen praktisch umgesetzt.

5.1 Entwicklung der Bilddifferenzierung (Bildsubtraktion)

Um Bewegungserkennung zu ermöglichen, wurde Bildsubtraktion angewendet. Hierbei werden zunächst der aktuelle und der vorlaufende **Frame** eines Videostreams jeweils in Graustufen umgewandelt. Diese Bilder werden nun voneinander subtrahiert. Das hieraus entstandene Bild zeigt die Helligkeitsunterschiede zwischen dem ersten und zweiten Graustufenbild an. Da hier zwei aufeinander folgende Bilder aus einem Videostream miteinander verglichen werden, kann man bei diesen Unterschieden davon ausgehen, dass es sich hierbei um Bewegungen handelt. Aus Performancegründen wurde sich hier für eine absolute Subtraktion und für Graustufen entschieden. Außerdem wird bei diesem Vorgang nur die Information benötigt, ob und wo Bewegung stattfindet. Weitere Farbinformationen sind hier folglich nicht nötig. Unterschiede zwischen den beiden Bildern bei einer Aufnahme Frequenz von 30 Bildern pro Sekunde sind für das bloße Auge nicht zu erkennen. Daher wurde im folgenden Beispiel auf das zweite Originalbild verzichtet.



Abbildung 2 - Original zu Subtraktion

5.2 Implementierung der Bildfilterung zur Rauschreduzierung

Während der Entwicklung ergaben sich mehrere Faktoren, die für ungewünschtes Bildrauschen in subtrahierten Bildern sorgten. Zu diesen Faktoren gehörten unter anderem sich ändernde Lichtverhältnisse, minimale Neufokussierungen und leichte Erschütterungen der Kamera. Dieses Bildrauschen zeigt sich deutlicher bei den nächsten für die weitere Bildverarbeitung nötigen Schritten.

Für die weiteren Bearbeitungsschritte wurden binäre Bilder benötigt. Für diesen Zweck wurde **Thresholding** angewendet.

Hierbei werden sämtliche Pixel auf ihre Helligkeit überprüft. Pixel die dabei einen bestimmten Schwellenwert unterschreiten, werden aus dem Bild entfernt. Pixel die den Schwellenwert erreichen oder überschreiten werden auf maximale Helligkeit gesetzt. Hierdurch entstehen binäre Bilder. Durch die damit verbundene Verdeutlichung von Pixeln werden Umrisse von sich bewegenden Personen und Objekten zwar deutlicher erkennbar, allerdings werden auch für das bloße Auge vorher nicht sichtbare Bildstörungen hervorgehoben.



Abbildung 3 – Links: Graustufenbild Rechts: Nach Thresholding mit Bildrauschen

Um dieses Bildrauschen zu entfernen und weiterhin die gewünschten Konturen zu verdeutlichen, wurden in der Folge verschiedene Kombinationen aus **Erosion** und **Dilatation** implementiert.

Bei **Erosion** eines Bildes durchläuft ein Strukturelement (zum Beispiel ein zweidimensionales **Array**), der sogenannte **Kernel**, das zu bearbeitende Bild. Trifft dieser **Kernel** auf weiße Pixel, die nicht den gesamten Kernel ausfüllen, werden diese aus dem Bild gelöscht. Hierdurch kann Bildrauschen, das aus einzelnen oder kleineren Gruppen von Pixeln besteht, erheblich reduziert werden. Allerdings werden gewünschte Konturen auf diese Art auch reduziert oder gar gelöscht.

Umgekehrt verhält es sich bei **Dilatation**. Hierbei durchläuft ebenfalls ein **Kernel** ein Bild. Trifft hier allerdings ein **Kernel** auf einen oder mehrere Pixel, die den Kernel nicht in seiner Gänze ausfüllen, wird der gesamte Bereich mit weißen Pixeln gefüllt. Gewünschte Konturen können so verdeutlicht, ungewünschte Pixelgruppen allerdings auch vergrößert werden. Verschiedene Kombinationen an **Erosion** und **Dilatation** werden in der Bildverarbeitung Öffnen und Schließen von Bildern genannt. Alle diese Vorgänge fallen in den Bereich der **morphologischen Operationen**.

Durch einsetzen verschiedener Kernelgrößen in den jeweiligen Vorgängen und dem mehrmaligen Kombinieren von Erosion und Dilatation in verschiedenen Reihenfolgen konnte Bildrauschen weitgehend entfernt werden, während gewünschte Konturen im nötigen Maße beibehalten wurden.



Abbildung 4 – Links: Thresholding mit Bildrauschen Rechts: Nach morphologischen Operationen

5.3 Integration von Clustering-Algorithmen

Um diese Konturen in einen logischen Zusammenhang zueinander setzen zu können wurden die Koordinaten der verbliebenen Pixel in als Punktkoordinaten in einem **Vektor** abgespeichert und per **DBSCAN** verarbeitet.

Die Aufgabe dieses Algorithmus ist es, zusammenhängende Punktmengen zu identifizieren, die vermutlich zur gleichen Bewegungsfläche gehören und diese zu **Clustern** zusammenzufassen. **DBSCAN** arbeitet auf Basis von zwei zentralen Parametern: **eps** (Epsilon) und **min_samples**. Der Epsilon-Wert definiert den maximalen Abstand zwischen zwei Punkten, damit diese als „**Neighbours**“ gelten. Der Parameter **min_samples** legt fest, wie viele **Neighbours** ein Punkt mindestens haben muss, um als „Kernpunkt“ eines **Clusters** angesehen zu werden.

Ein wichtiger Aspekt der Implementierung ist die Verwendung eines Spatial indexing-Verfahrens. Dabei wird der zweidimensionale Raum in ein Gitter (Raster) eingeteilt. Jeder erkannte Punkt wird einer bestimmten Rasterzelle, basierend auf seiner Position und der vorher definierten Zellgröße (welche wiederum auf dem Epsilon-Wert beruht) zugeordnet. Dadurch kann die Suche nach **Neighbours** während des **Clustering**-Prozesses erheblich beschleunigt werden, da nur Punkte aus derselben Zelle und den benachbarten Zellen betrachtet werden. Dadurch wird die Anzahl der notwendigen Distanzberechnungen deutlich reduziert und die Performance bei größeren Punktmengen gesteigert.

Im eigentlichen Clustering-Prozess wird zunächst bei jedem Punkt überprüft, ob er bereits untersucht wurde. Wenn nicht, wird er auf seine **Neighbours** mittels der Funktion `regionQuery()` untersucht.

Falls genügend **Neighbours** vorhanden sind, wird ein neuer **Cluster** erstellt. Die Methode `expandCluster()` fügt in der Folge angrenzende Punkte, die ebenfalls genügend **Neighbours** haben, dem **Cluster** hinzu, woraufhin deren angrenzende Punkte weiter untersucht werden.

Die Ergebnisdaten des **DBSCAN**-Algorithmus bestehen aus einem Vector von **Labels**. Diese **Labels** sind dafür zuständig für jeden einzelnen Punkt die Zugehörigkeit zu einem Cluster festzulegen. Gehören zum Beispiel Punkt A, B, und C zum Cluster 1, wird ihnen jeweiligen Labels ein Wert von 1 zugeordnet. Diese Labels werden anschließend genutzt, um mithilfe der Funktion `createClustersFromLabels()` neue Gruppen von Punkten zu bilden. Diese Gruppen entsprechen den erkannten Bewegungsclustern und werden zur weiteren Verarbeitung beziehungsweise Darstellung genutzt.

Punkte, die zu keinem Cluster gehören (also weniger als **min_samples Neighbours** haben), werden mit dem Wert -1 markiert und gelten als Rauschen.

Durch diese Form des Clusterings wurde eine robuste und flexible Segmentierung von Bewegungsbereichen im Videostream ermöglicht, ohne die Anzahl oder Form der Cluster

vorher kennen zu müssen. Die hieraus resultierenden Cluster wurden als Information zur Zeichnung von Konturen einzelner Objekte genutzt. Dadurch, dass diese Konturen ein Objekt komplett umschließen, konnten diese über das aktuelle Bild des Videostreams gelegt und ausgegeben werden.

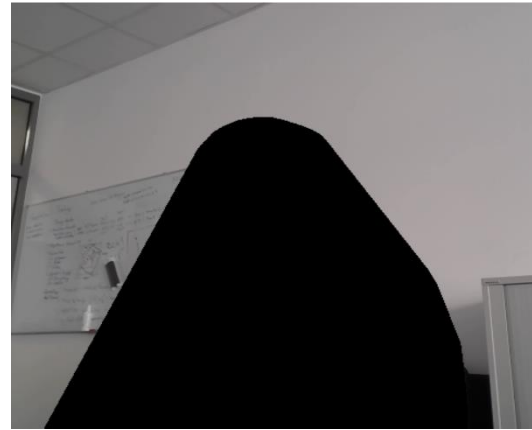
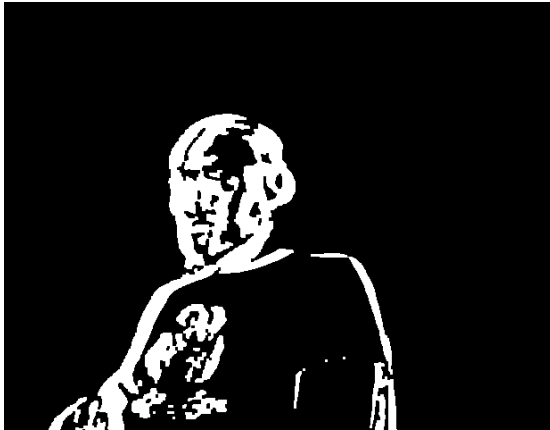


Abbildung 5 – Links: Zur Übergabe an das Clustering bereite Bild. Rechts: Originalbild, über das ein Cluster als ausgefüllte Kontur gelegt wurde

5.4 Einbindung in die KI

Zum Zweck der Gepäckerkennung wurde das **YOLO-X** Objekterkennungsmodell mit einem vom **COCO-Datensatz** abgeleiteten Datensatz trainiert. Hierfür wurden mit einem betriebseigenen Python-Script nur **Annotationen** aus den Kategorien Rucksäcke, Handtaschen und ähnlichen zu einer gemeinsamen Kategorie „Luggage“ zusammengeführt. Mit diesem neuen Datensatz wurde das KI-Modell trainiert, um es die Videoausgabe der Projektarbeit analysieren zu lassen.

6. Qualitätssicherung

Um dem hohen Qualitätsstandard der Mühlbauer Group gerecht zu werden, wurde sich sowohl während als auch nach der Implementierung unterschiedlicher Testverfahren beholfen, welche im Nachfolgenden genauer erläutert werden. Dies geschah um sicherzustellen, dass die im Verlauf des Projekts entstanden Softwareanwendungen den zuvor festgelegten Anforderungen genügen und zudem auch die Qualitätsmerkmale von Software im Allgemeinen erfüllen. Darunter fällt beispielsweise die Zuverlässigkeit, die Benutzbarkeit, die Effizienz, die Wartbarkeit oder auch die Änderbarkeit.

6.1 Whitebox- Tests

Der Quellcode wurde während und nach der Entwicklung auf Funktionalität und Fehlerfreiheit überprüft. Dadurch konnten Fehler bereits frühzeitig erkannt und behoben werden. So wurde hierbei auf logische Fehler und Syntaxfehler geachtet. Um Fehler schon frühzeitig erkennen und beheben zu können, wurden firmeninterne Tools für **Unittests** angepasst, die nach jeder Änderung ausgeführt wurden um die jeweiligen Komponenten auf ihre Funktionsfähigkeit zu überprüfen.

6.2 Blackbox- Tests

Neben den zuvor bereits genannten Whitebox Tests sind auch sogenannte Blackbox Tests eine tragende Säule der Qualitätssicherung. Im Zuge dessen wurde in einer Testumgebung, der Betrieb eines Smart Corridors simuliert, indem mehrere Personen mit und ohne Gepäck sich durch das Sichtfeld einer Kamera bewegten. Hierbei wurde besonderes Augenmerk auf

die Qualität der Bildfilterung bei sich ändernden Lichtverhältnissen und die korrekte Überdeckung sich bewegender Objekte und Personen gegenüber stationären gelegt. Die genaue interne Funktionsweise wurde hierbei nicht berücksichtigt. Um dies zu gewährleisten, wurde der Blackbox Test von einer Person ausgeführt, die nicht unmittelbar an der Implementierung beteiligt war und dem zufolge keine genauen Kenntnisse über den Quellcode verfügte.

Zu sehen im Beispielbild der Testumgebung ist die gewünschte Funktionsweise des Tools. Im linken oberen Bereich wird eine gehende Person, die einen Rucksack trägt, vollständig verdeckt. Dadurch findet keine Identifizierung des getragenen Rucksacks statt. Mitte rechts wird eine Person größtenteils verdeckt, die zwar steht, allerdings ihren Oberkörper bewegt. Die Tasche, die vor ihre Brust geschnallt ist, wird aufgrund der Bewegung verdeckt. Minimale Bewegungen der absichtlich sehr stillstehenden Person in der rechten Bildhälfte (Fingerbewegung und Augenzwinkern) werden erkannt. Die einzigen Gepäckstücke, die der K.I. in der Übertragung angezeigt werden, sind die stationären unten rechts und rechts oben im Bild. Diese Gepäckstücke werden auch erfolgreich von der K.I. geortet und als solche identifiziert. Dies ist durch die rechteckige Einrahmung erkennbar.



Abbildung 6 - Testumgebung Beispielbild

Testszenario	Ergebnis
Verschiedene Beleuchtungen	Bestanden
Leichte Erschütterungen des kamerahaltenden Rahmens	Bestanden
Langsame Bewegungen	Bestanden
Stationäres Gepäck teilweise verdeckt durch sich bewegende Personen	Bestanden
Gar keine Bewegungen	Bestanden
Überschneidung gleichfarbiger Gegenstände	Bestanden
Reflexionen	Bestanden

Tabelle 5 – Testszenarien

7. Projektabschluss

7.1 Dokumentation

Im Anschluss an dieses Dokument finden sich das Pflichtenheft und eine Entwicklerdokumentation für die Bildfilterung und die Bewegungserkennung. Aufgrund der vielen internationalen Mitarbeitenden des Unternehmens wurde die Entwicklerdokumentation auf Englisch erstellt.

7.2 Übergabe

Nach Abschluss der Implementierung und erfolgreichem Durchlaufen umfangreicher Tests wurde das Tool in den Seamless Corridor integriert. Anschließend wurde mit dem zuständigen Personal ein Termin für eine Schulung vereinbart, um alle Funktionalitäten und die optimale Benutzung/Kalibrierung vorzustellen.

7.3 Schulung

Um eine möglichst effektive Nutzung des entwickelten Tools zu gewährleisten, wurde eine Mitarbeiterschulung durchgeführt. In dieser wurde auf die verschiedenen Einzelheiten der Bildfilterung und der Bewegungserkennung eingegangen. Darunter waren beispielsweise Hinweise zur grundlegenden Kalibrierung der Bildfilterung und des **Clusterings**. All dies trägt dazu bei, die Produktivität bei der Einrichtung zu steigern, mögliche Fehler während der Kalibrierung zu verhindern und somit auch die Zuverlässigkeit des Produktes zu erhöhen.

7.4 Soll- Ist- Vergleich

Aufgrund des softwareseitigen modularen Aufbaus und des Seamless Corridors war eine schnellere Einarbeitung in das bestehende Projekt möglich. Dadurch konnten zwei Stunden eingespart werden. Da sich die Implementierung des Clusterings als aufwändiger als erwartet herausstellte, mussten zwei Stunden mehr als erwartet aufgewendet werden. Der zeitliche Rahmen des Projektes von 80 Stunden konnte damit eingehalten werden.

Vorgang	Soll	Ist
Planungs- und Analysephase	9	9
Analyse des Ist-Zustands	4	4
Stakeholder Kommunikation	1	1
Entwurf des Soll-Konzepts	3	3
Analyse der Wirtschaftlichkeit	1	1
Entwurfsphase	16	14
Einarbeitung in das bestehende Projekt	6	4
Recherche zu verfügbaren Technologien	5	5
Erstellen eines Pflichtenheftes	2	2
Entwurf der Architektur	3	3
Implementierungsphase	37	39
Entwicklung der Bilddifferenzierung (Bildsubtraktion)	8	8
Implementierung der Bildfilterung zur Rauschreduzierung	9	9
Integration von Clustering-Algorithmen	10	12
Einbindung in die KI	6	6
Debugging und Fehlerbehebung	4	4
Qualitätsmanagement	9	9
Durchführen von Whitebox-Tests	4	4
Blackbox-Tests an der Maschine	5	5
Dokumentation und Schulung	8	8
Erstellen einer Dokumentation	6	6
Schulung von Mitarbeitenden	2	2
Übergabe	1	1
Gesamt	80h	80h

Tabelle 6 - Soll-Ist Vergleich

7.5 Fazit

Das Projekt konnte innerhalb der vorgegebenen Zeitrahmens umgesetzt werden und alle Vorgaben des Soll-Konzeptes umsetzen. Es erwies sich damit als voller Erfolg. Die neuen Funktionen ergänzen das Produkt funktional sinnvoll und bieten einen echten Mehrwert, der internationalen Sicherheitsanforderungen entspricht.

7.6 Ausblick

Für die Zukunft ist geplant, die entwickelten Funktionen in weitere Mühlbauer-Produkte zu integrieren. Weiterhin wurde mit der Implementierung des Projektes der technische Grundstein für Funktionserweiterungen wie Detektion von **Tailgating** und damit eine Verbesserung der Personenerkennung gelegt.

8. Verzeichnisse

8.1 Glossar

Begrifflichkeit	Erklärung
Annotation	Manuelle Markierung von Objekten in Bildern zur Trainingsvorbereitung für Objekterkennungsmodelle
Array	Datenstruktur, die eine feste Anzahl gleichartiger Elemente speichert, meist mit Indexzugriff
Backend	Teil einer Anwendung, der für Datenverarbeitung, Logik und Schnittstellen zuständig ist.
Bildsubtraktion	Verfahren der Bildverarbeitung, bei dem die Differenz zweier Bilder berechnet wird., um Änderungen zu erkennen
C++	Kompilierte leistungsstarke Programmiersprache
Cluster	Gruppe ähnlicher Datenpunkte, die bei Clustering-Algorithmen gemeinsam klassifiziert werden
Clustering	Einteilung ähnlicher Datenobjekte in Gruppen (Cluster)
COCO-Datensatz	Standardisierter Datensatz für Objekterkennung
Dilatation	Morphologische Operation, die Bildbereiche (z.B. weiße Pixel) ausweitet und Löcher schließt.
DBSCAN	Dichtebasierter Clusteringalgorithmus, der Cluster durch die lokale Dichte von Datenpunkten definiert
eps (DBSCAN)	Radius um einen Datenpunkt, innerhalb dessen Neighbours für diesen bestimmt werden
Erosion (Bildverarbeitung)	Morphologische Operation, die Bildbereiche schrumpft und kleine Objekte entfernt
Frame	Einzelbild in einer Sequenz von Bildern, z.B. aus einem Video
Hierarchisches Clustering	Clusteringverfahren, das eine Baumstruktur für die Gruppierung von Datenpunkten erstellt
Kernel (Bildverarbeitung)	Matrix zur Bildfilterung z.B. bei Weichzeichnung
K-Means	Clusteringverfahren, das Daten in Gruppen aufteilt, basierend auf der minimalen Distanz zum Clusterzentrum
Library	Sammlung vorgefertigter Funktionen oder Klassen, die zur Wiederverwendung in Programmen dient
Min_samples (DBSCAN)	Minimale Anzahl von Nachbarn (inkl. Punkt selbst), die ein Punkt haben muss, um als Kernpunkt zu gelten
Morphologische Operationen	Bildverarbeitungsmethoden zur Formveränderung von Objekten, z.B. Erosion oder Dilatation
Neighbours (DBSCAN)	Datenpunkte im Umkreis von eps, die zur Bestimmung eines Clusters herangezogen werden
OpenCV	Open-Source-Bibliothek für Computervision und Bildverarbeitung in C++ und Python

Python	Programmiersprache, bekannt für Lesbarkeit, zugängliche Syntax und breite Bibliotheksunterstützung
Smart Seamless Corridor	System zur Personenerkennung für Zutrittskontrolle
Tailgating	Sicherheitsrisiko, bei dem eine unauthorisierte Person einer berechtigten Person sehr nah folgt
Thresholding (Bildverarbeitung)	Segmentierungsverfahren, bei dem Pixel nach einem bestimmten Schwellenwert aussortiert werden
Unittest	Automatisierter Softwaretest, der einzelne Komponenten auf korrektes Verhalten überprüft
Vektor (C++)	Dynamische Datenstruktur in C++, die Elemente desselben Typs speichert
Whiteboxtest	Testverfahren, bei dem die interne Struktur der Software bekannt ist und gezielt getestet wird
YOLO-X	Echtzeitobjekterkennungsmodell der YOLO-Familie, kombiniert hohe Genauigkeit mit Geschwindigkeit

Tabelle 7 – Glossar

8.2 Abbildungsverzeichnis

Abbildung 1 - Amortisierungsdauer.....	8
Abbildung 2 - Original zu Subtraktion	11
Abbildung 3 – Links: Graustufenbild Rechts: Nach Thresholding mit Bildrauschen	12
Abbildung 4 – Links: Thresholding mit Bildrauschen Rechts: Nach morphologischen Operationen.....	13
Abbildung 5 – Links: Zur Übergabe an das Clustering bereite Bild. Rechts: Originalbild, über das ein Cluster als ausgefüllte Kontur gelegt wurde.....	14
Abbildung 6 - Testumgebung Beispielbild.....	15

8.3 Tabellenverzeichnis

Tabelle 1 - Zeitlicher Rahmen.....	7
Tabelle 2 - Personalkosten.....	8
Tabelle 3 - Ressourcen.....	9
Tabelle 4 - Algorithmusvergleich	10
Tabelle 5 – Testszenarien	15
Tabelle 6 - Soll-Ist Vergleich	17
Tabelle 7 – Glossar	19



MB Automation GmbH & Co. KG
Josef-Mühlbauer-Platz 1 | 93426 Roding | Germany
Tel.: +49 9461 952 0 | Fax: +49 9461 952 1101
Mail: info@muehlbauer.de | Web: www.muehlbauer.de

