

Hochschule für Technik und Wirtschaft Dresden

Fakultät Maschinenbau



UNIVERSITY OF APPLIED SCIENCES

Aufgabenblatt für die Diplomarbeit

im Studiengang / Studienrichtung

Fahrzeugtechnik / Kraftfahrzeugtechnik

Name des Diplomanden / Matr.-Nr.:

Marvin Eschke / 45912

Thema:

Entwicklung einer dynamischen Bilddatenbank für maschinelle Lernverfahren

Aufgabe:

Durch den Einsatz von maschinellen Lernverfahren kann die Erkennung von relevanten Infrastrukturinformationen deutlich verbessert werden. In der Diplomarbeit soll ein funktionsfähiges und erweiterbares Konzept für die Erkennung von Verkehrszeichen entwickelt und getestet werden.

Teilschritte:

- Einarbeitung und Literaturrecherche zur Thematik
- Konzeption und Implementierung einer Matlab-Bilddatenbank für verschiedene Verkehrszeichen
- Auswahl geeigneter Matlab-Erkennungsalgorithmen
- Test und Bewertung anhand typischer Testszenarien

Hochschulbetreuer: Prof. Dr. rer. nat. Toralf Trautmann
Dipl.-Ing. (FH) Dirk Engert

Ausgehändigt am: 01.04.2022

Einzureichen bis: 01.09.2022

A blue ink signature of Prof. Dr. rer. nat. Toralf Trautmann.

Prof. Dr. rer. nat. Toralf Trautmann
Verantwortlicher Hochschullehrer

A blue ink signature of Prof. Dr.-Ing. Gudrun Lange.

Prof. Dr.-Ing. Gudrun Lange
Prüfungsausschussvorsitzende

DIPLOMARBEIT

Marvin Jörg Eschke

Matrikelnummer: 45912

Studiengang: Fahrzeugtechnik

Gruppe: 18-052-01

Immatrikulationsjahr: 2018

Thema:

Entwicklung einer dynamischen Bilddatenbank für maschinelle
Lernverfahren

Betreuer:

Dipl.-Ing. (FH) Dirk Engert (HTW Dresden)

Betreuender Hochschullehrer:

Prof. Dr. rer. nat. Toralf Trautmann (HTW Dresden)

Ort:

Dresden

Abgabe:

01.09.2022

Inhaltsverzeichnis

Abkürzungsverzeichnis	III
Abbildungsverzeichnis	IV
Tabellenverzeichnis	VI
1 Einleitung.....	1
2 Grundlagen der Bildverarbeitung	2
2.1 Objekte und deren Klassifikation im Merkmalsraum	4
2.2 Lernfähigkeit einer Maschine.....	6
2.3 Einflussparameter	7
2.4 Zustände und Rechenleistung.....	8
2.5 Digitale Bildaufnahme	9
3 Stand der Technik	13
3.1 Überblick Begrifflichkeiten	13
3.2 Einsatzgebiete und Bewertungsmöglichkeit	14
3.3 Erkennungsalgorithmen.....	17
3.3.1 Funktionsweise eines CNN-Algorithmus	17
3.3.2 Funktionsweise eines ACF-Algorithmus.....	19
3.3.3 Funktionsweise eines Cascade-Algorithmus	23
4 Erstellung einer eigenen Bilddatenbank	25
4.1 Auswahl Kamera.....	25
4.2 Umsetzung Software.....	29
4.3 Programmablauf für Bildaufnahme	30
4.4 Inbetriebnahme Kamera	32
4.5 Zusammenführung der Skripte	32
4.6 Ermittlung der Standortspositionen der Stoppschilder	37
4.7 Aufnahmefahrten im Stadtgebiet Dresden	37
5 Bereitstellung geeigneter Bilddatenbanken	40
5.1 Bilddatenbank aus Aufnahmefahrten	40
5.2 Erweiterung durch existierende Bilddatenbank	43
5.2.1 Anpassung Bildformat und Positionsbeschreibung	44
5.2.2 Spiegelung eines Bilddatensatzes.....	46
5.3 Erstellung von Trainingsdatenbanken.....	48

6	Untersuchung der unterschiedlichen Algorithmen	49
6.1	Testszenarien mit dem ACF-Detektor	49
6.2	Testszenarien mit dem Fast-R-CNN-Detektor	55
6.3	Testszenarien mit dem Cascade-Detektor	58
6.4	Zwischenresultat der Untersuchung	62
7	Auswertung	65
7.1	Bewertung des ACF-Detektors	66
7.2	Bewertung des Fast-R-CNN-Detektors	68
7.3	Bewertung des Cascade-Detektors	70
7.4	Resultat der Auswertung	72
8	Zusammenfassung und Ausblick	73
	Literatur- und Quellenverzeichnis	74
	Eidesstattliche Erklärung	78
	Anlagenverzeichnis	79
	Anlagen	80

Abkürzungsverzeichnis

ACF	Aggregate Channel Features
CNN	Convolutional Neural Network
GigE	Gigabit Ethernet
fn	falsch negativ
fp	falsch positiv
ISA	Intelligent Speed Assistance
Lidar	Light detection and ranging
MMC	Matthews Correlation Coefficient
PoE	Power over Ethernet
rn	richtig negativ
ROI	Region Of Interest
rp	richtig positiv

Abbildungsverzeichnis

Abbildung 2.1: Grafik zur optischen Linse [8].....	10
Abbildung 2.2: Dreiecksbeziehung einer Linse [9].....	10
Abbildung 2.3: Veranschaulichung Histogramm [10].....	12
Abbildung 3.1: Aufbau künstliche Intelligenz [13].....	13
Abbildung 3.2: Schichten eines CNN [6].....	18
Abbildung 3.3: Größe des Bildes um Faktor 2 erhöht [15]	19
Abbildung 3.4: Bild um Faktor 2 verkleinert [15].....	20
Abbildung 3.5: Kantenabweichung bei Größenänderung [15].....	20
Abbildung 3.6: Rechenzeitverkürzung durch Änderung Sprünge in Maßstäben [15]	22
Abbildung 3.7: Annäherungsmöglichkeit für vorgeschlagene Berechnung [15].....	22
Abbildung 3.8: Ablauf ACF-Detektor bei Eingabebild [15].....	23
Abbildung 3.9: Funktionsprinzip Cascade-Detektor [17]	24
Abbildung 4.1: PoE-Switch mit angeschlossenen Komponenten	27
Abbildung 4.2: Position der Hardware im Aufnahmefahrzeug	28
Abbildung 4.3: Elemente für Kamerabefestigung	28
Abbildung 4.4 Komponenten und deren Anbindung im Überblick.....	29
Abbildung 4.5: Messaufbau für Zeitdifferenz.....	34
Abbildung 4.6: Testaufbau mit Markierungen auf Prüffeld	34
Abbildung 4.7: Differenz Kamera und Lidar-Sensor bei Auswertung	35
Abbildung 4.8: Aufbau Prüffeld für Differenzbestimmung [23].....	36
Abbildung 4.9: Übertragung der aktuellen Umgebung auf Laptop	38
Abbildung 4.10: Positionsdifferenz des Lidar-Sensors [23].....	38
Abbildung 5.1: Festlegung ROI Stoppschild in Matlab.....	42
Abbildung 5.2: Stoppschild gespiegelt.....	43
Abbildung 5.3: Vorfahrtsschild gespiegelt [24]	44
Abbildung 5.4: Positionskennzeichnung bei Institut für Neuroinformatik [24]	45
Abbildung 5.5: Vorgabe für Übergabe in Matlab	45
Abbildung 5.6: Inhalt der erstellten Tabelle	46
Abbildung 5.7: Berechnungsgrundlage für Spiegelung [24]	47
Abbildung 5.8: Positionsangabe für gespiegeltes Bild [24]	47
Abbildung 6.1: Ergebnis ACF-Detektor.....	50
Abbildung 6.2: Ergebnisse ACF-Detektor mit Testbildern aus Aufnahmefahrt	50
Abbildung 6.3: Möglichkeiten für Kennzeichnung des ROI [25].....	51
Abbildung 6.4: links genau und rechts grob gekennzeichnet	51
Abbildung 6.5: Ergebnisse nach Filterung	52
Abbildung 6.6: Reihenfolge Trainingsdatensatz links ungefiltert und rechts gefiltert.....	52
Abbildung 6.7: Position Verkehrsschild [30].....	53
Abbildung 6.8: Ergebnis des ACF-Detektors	54
Abbildung 6.9: Ergebnis Fast R-CNN-Detektor [28]	55
Abbildung 6.10: Ergebnis bei 41 Trainingsbildern und gespiegeltem Testbild [30]	56
Abbildung 6.11: gespiegelter Testdatensatz und dessen Ergebnis [30].....	57
Abbildung 6.12: Erhöhung des gespiegelten Trainingsdatensatzes [30].....	57
Abbildung 6.13: Ergebnis des Fast-R-CNN-Detektors.....	58
Abbildung 6.14: Erkennung Cascade-Detektor	59
Abbildung 6.15: links fein und rechts grob gekennzeichnete Trainingsdatensatz [29]	59
Abbildung 6.16: Filterungsauswirkung beim Cascade-Detektor	60
Abbildung 6.17: Einflussnahme Trainingsdaten mit erhöhten Schwellenwert [30]	61
Abbildung 6.18: Resultat von Cascade-Detektor.....	62
Abbildung 7.1: Berechnung Testbilder für Auswertung.....	65
Abbildung 7.2: Bewertungsergebnis ACF-Detektor	66

Abbildung 7.3: MCC-Werte des ACF-Detektors	67
Abbildung 7.4: Abstandserkennung ACF-Detektor	67
Abbildung 7.5: Bewertung Fast-R-CNN-Detektor	68
Abbildung 7.6: MCC-Werte des Fast-R-CNN-Detektors	69
Abbildung 7.7: Abstandserkennung R-CNN Detektor	69
Abbildung 7.8: Bewertung Cascade-Detektor	70
Abbildung 7.9: MCC-Werte des Cascade Detektors	71
Abbildung 7.10: Abstandserkennung Cascade-Detektor	71

Tabellenverzeichnis

Tabelle 4.1: Auszug Datenblatt Basler acA1920-40gc GigE-Kamera.....	25
Tabelle 6.1: Detektoren für Stoppschilderkennung in Matlab.....	49
Tabelle 6.2: Überblick des Zwischenresultates	63

1 Einleitung

Es gilt die Fahrzeugsicherheit stetig weiter zu verbessern, weswegen immer mehr Assistenzsysteme im Automobilbereich zum Einsatz kommen. Seit Juli 2022 ist beispielhaft eine neue Verordnung für eine Tempobremse in Kraft getreten, die für eine Homologation eines neuen Fahrzeugtyps zu beachten ist. Spätestens im Jahre 2024 ist ein solches System für alle Neufahrzeuge vorgeschrieben. Dem Fahrer wird hierbei die derzeitig vorhandene Geschwindigkeitsbegrenzung angezeigt und fordert von diesem die Anpassung des Fahrzeugtempos. Eine Umgebungserkennung wird somit notwendig, um die vorzufindenden Verkehrsschilder erfassen zu können. Dies wird derzeit mit integrierten Standortdaten im Navigationssystem und/oder Kamerasystem umgesetzt, die aber teilweise nicht die aktuelle Beschilderungslage wiedergeben oder zu falschen Interpretationen neigen. Die größten Schwächen der auf Fotografie basierenden Systemen treten bei Fahrten in der Dunkelheit oder bei LED-Zeichen auf, woraus sich wiederum Verbesserungspotenziale ergeben [1–3].

Dieses Verfahren der Objektbestimmung ist bisher kein Bestandteil des Lehrbereiches der Fahrzeugtechnik an der Hochschule für Technik und Wirtschaft Dresden, weswegen ein geeigneter Erkennungsalgorithmus für das dort genutzte Computerprogramm Matlab gesucht wird. Die verschiedenen vorzufindenden Ansätze und Beispiele für maschinelle Lernverfahren sind in dieser Arbeit näher analysiert und verglichen wurden, um deren einzelne Besonderheiten und Nutzbarkeit herauszufinden. Als Grundlage wird für dessen Überprüfung eine eigene Bilddatenbank benötigt, um die notwendigen Verkehrszeichen antrainieren und testen zu können. In Aufnahmefahrten sind dafür die Videos erstellt wurden, um die benötigten Fotodateien zu extrahieren. Eine Nutzung einer separaten Kamera ist daher notwendig. Die Schwerpunkte umfassen die Position des Objektes, Reihenfolge im Anlerndatensatz und die Anzahl der Bilder. Überdies wird der Zusammenschluss eines immer mehr im Einsatz kommenden Laserscanners erläutert, um noch weitere Informationen zur Umwelt und dem Abstand zum Objekt zu erhalten.

2 Grundlagen der Bildverarbeitung

Der Mensch hat eine enorme Leistungsfähigkeit, was das Erkennen von Objekten in seinem Umfeld betrifft. Er nimmt dabei nicht nur die gesammelten Eindrücke auf, sondern vergleicht diese mit den schon erworbenen Erfahrungen und Erlebnissen aus seinem bisherigen Leben. Nach dem aktuellen Stand der Technik gibt es keine Möglichkeiten, dass ein technisches System die visuellen Sinneseindrücke eines Menschen und die daraus abgeleiteten Erkenntnisse vollständig und fehlerfrei übernehmen kann. Ein Objekt ist daher so zu beschreiben und zu klassifizieren, sodass es sich von anderen unterscheidet. Dies ist dabei eine hauptsächliche Aufgabe in der maschinellen unterstützten Objekterkennung. Ein Mensch hat mit dieser Angelegenheit keine Probleme und verarbeitet die komplexen Wahrnehmungen mit seinem Gehirn, worauf sich ein Computerprogramm nicht stützen kann. Diesem stehen lediglich Zahlen und mathematische Operationen zur Verfügung, was erst eine Ein- und Zuordnung der einzelnen Gegenstände ermöglicht. Zur Bewältigung dieser Herausforderung besteht die Notwendigkeit der elektronischen Rechenanlage tunlichst viele Informationen und Daten zur Verfügung zu stellen.

Für die Umsetzung der Erkennung werden viele einzelne Bilder benötigt, die wiederum mathematisch umzuwandeln sind. Aus diesem Ergebnis lassen sich verschiedene Unterscheidungseigenschaften extrahieren, was eine Zuordnung zulässt und sogenannte Merkmalsvektoren bildet. Diese beschreiben dabei gleich aussehende Objekte, welche sich dann wiederum in eine Klasse einordnen lassen. Die Zusammenfügung wird hierbei als Clustering bezeichnet. Nimmt man beispielsweise eine Anzahl von Bildern verschiedener Hunde- und Hasenrassen, dann sollte jede Tierart für sich klassifiziert werden. Die Rassen jeder Klasse haben zwar eigene Abweichungen zueinander, aber weisen ebenfalls bestimmte Hauptmerkmale auf. Diese gilt es zu finden, um einen aufgenommenen Hasen nicht als Hund zu erkennen und auszugeben. Im Gegensatz dazu gibt es die Klassifizierung, wo die Unterscheidungsmerkmale nicht erst zu finden sind. Dem Computerprogramm sind die Merkmale in einer geeigneten Form zu übermitteln, worauf dieser sich antrainiert und eine entsprechende Einordnung des Gegenstandes vornehmen kann [4].

In der Realität kann ein zu erkennendes Objekt gedreht oder verschoben sein, was die Erkennung zusätzlich erschwert. Eine weitere Einflussnahme ist bei dem Blickwinkel, dem vorherrschenden Umwelteinfluss und andere Besonderheiten zu beachten. Diese rufen Abweichungen zum Ideal hervor und sind demgemäß mit einzubeziehen. Ein besonderes Augenmerk ist infolgedessen auf die Datenbank und Anwendung zu legen.

Durch zusätzliche Einflussparameter und deren Untersuchung steigt auch der Aufwand und die Zeit für die computergestützte Berechnung. Es ist daher ratsam, bei der Aufnahme der Objekte auf dessen Orientierung zu achten und diese erst bei idealeren Positionen aufzunehmen. Die günstigsten Bedingungen sind folglich in vier Varianzen festgehalten:

- Translationsinvarianz, wo die vorhandenen Objektparameter selbst beim Verschieben konstant bleiben
- Rotationsinvarianz, wo die vorhandenen Objektparameter selbst nach Rotation konstant bleiben
- Größeninvarianz, wo die vorhandenen Objektparameter bei unterschiedlicher Vergrößerung oder Verkleinerung konstant bleiben
- Spiegelinvarianz, wo die Objektparameter beim Spiegeln über die Symmetrieachse konstant bleiben.

Alle Bedingungen sind nicht zu erfüllen, was hieraus schnell ersichtlich wird. Es kann beispielhafterweise Abweichungen im Winkel und den Umgebungsbedingungen vorherrschen. Trotzdem sollte der resultierende Merkmalsvektor so beschrieben werden, dass er alle Objekte in einer bestimmten Klasse erkennen kann. Dieser lässt sich mit den folgenden geometrischen und topologischen Besonderheiten erweitern, die wie folgt festgelegt sind:

- Fläche umfasst die Anzahlen von Pixeln eines Objektes
- Masse beschreibt die Summe der Grauwerte eines Objektes
- Umfang beschreibt die Anzahl der Randpixel eines Objektes
- Schwerpunkt eines Objektes kann über die Form eines Binärbildes oder des Grauwertbildes des Objektes berechnet werden.

Eine andere Möglichkeit bietet die Zuordnung mittels Formparameter, die den Verlauf des Objektrandes beschreibt. Dafür können verschiedene Codierungen zur Hilfe genommen werden:

- Randcodierung nach Freeman: Der Rand eines Objektes wird mit einer bestimmten Zahlenfolge verschlüsselt. Als Startpunkt dient ein vorgegebener Objektpixel, über der man sich dann im Uhrzeigersinn immer weiter herantastet. Die entstehende Codierung beschreibt dann dieses vollständig.
- Differenzialcode: Es beschreibt die Ableitung der vorangegangenen Randcodierung, was wiederum eine Richtungsänderung aufzeigen kann. Berechnet wird dies durch das Differenzieren von zwei benachbarten Funktionswerten in einem Freeman Code.
- Fourierdescriptoren: Die Randcodierung beschreibt im Grunde eine immer wiederkehrende Funktion, da man das Objekt beliebig oft umkreisen kann. Es ist hierfür eine Fourierreihe aufstellbar, die mit ihren Koeffizienten die Objektparameter widerspiegeln.

Für die statischen Größen können die Kombinationen normierter zentraler Momente verwendet werden. Sie bilden Merkmalsvektoren mit sieben Parametern, aus denen ein Rückschluss auf das Objekt entsteht. Es ergeben sich daraus sieben Objektmerkmale die eine Größen-, Translations- und Rotationsvariante vorweisen. Bei Spiegelung ändert sich lediglich das Vorzeichen des siebten Parameters, was als Besonderheit festzuhalten ist.[5]

2.1 Objekte und deren Klassifikation im Merkmalsraum

Die davor erläuterten Merkmalsvektoren geben die bestimmten Unterscheidungen der einzelnen Objektklassen wieder. Weist ein zu untersuchender Gegenstand ähnliche Besonderheiten auf, dann wird es sich in einer entsprechenden Gruppe einteilen lassen. Diese sind in vielen Fällen nicht bekannt, weswegen man sich erste Lernobjekte zur Hilfe nimmt. Durch diese Hilfsmittel und einer Phase des Einlernens ist es dann möglich, die verschiedenen Objektklassen zu definieren und im Merkmalsraum festzulegen.

Die Phase des Lernens kann in zwei unterschiedlichen Arten stattfinden:

- überwachtes Lernen: Ist eine Vielzahl von Parametern als Verbund zu erkennen, dann wählt man diese Form des Lernens. Es werden spezifische Merkmale der Anlernobjekte gesucht und hervorgerufen, um die gemeinsamen Objektgruppen zu bilden. Das Training dient zur Festlegung der Grenzen und der späteren Klassifizierungsmöglichkeit. Durch diese entstehen die zu findenden Klassen im Merkmalsraum, was dann die neu zu untersuchenden Bilder einordnen lässt.
- unüberwachtes Lernen: Am Anfang sind hierbei noch keine Parameter zu erkennen. Die einzelnen Klassen sind vorerst zu definieren und zu finden. Mit Hilfe des Clustering und deren genutzte Trennverfahren bilden sich eigene Merkmalsräume. Es ist dabei festzuhalten, dass nicht alle eventuell vorzufindenden Merkmale eine eigene Gruppe bilden und somit verworfen werden können. Die entsprechenden Unterschiede bilden ein gewisses Gebiet, wodurch sich im Testfall ein zu untersuchendes Objekt einordnen lässt.

Der nächste Schritt umfasst eine Klassifikationsphase. Es beschreibt dabei das neue Klassifizieren eines unbekannten Objektes, wobei eine Berücksichtigung seines individuellen vorzufindenden Merkmalsvektors stattfindet. Das entsprechende notwendige Testobjekt ist in der richtigen trainierten Gruppe einzuordnen. Dies entspricht der Klassenzuordnung in der Realität und sollte demzufolge keine Abweichungen aufweisen. Umgesetzt wird dies in den meisten Programmen durch dafür vordefinierte Algorithmen, die üblicherweise als Klassifikator fungieren. Die Auslegung hierfür unterliegt verschiedenen Ansätzen, wodurch sich deren Umsetzung unterscheidet. Es ist für die eigene Anwendung und deren Problemstellung stets der passendste Ansatz zu finden, um eine möglichst präzise Unterscheidung und deren spätere Zuordnung zuzulassen. In manchen Anwendungsfällen kann es dazu kommen, dass keine direkte Klasse zu finden ist. Dies wird als Rückzuweisungsklasse bezeichnet und verlangt von dem Anwender eine manuelle Zuordnung.[5]

2.2 Lernfähigkeit einer Maschine

Die meisten momentan vorzufindenden elektronischen Hilfsmittel können nur vorgegebene und programmierte Befehle ausführen. Der Anwender wählt ein dafür passendes Hilfsmittel und deren Software aus. Die Kommunikation der ausgewählten Eingabeparameter und die damit verbundene Ausgabe sind in den meisten Fällen über einen Bildschirm realisiert. Die im Hintergrund ablaufenden vorprogrammierten Rechenschritte sind nicht immer ersichtlich und nachvollziehbar, was für die meisten Benutzer zugleich irrelevant ist. Der Programmierer hingegen hat die entsprechenden stattfindenden Abfolgen und Gesetzmäßigkeiten in eine entsprechende Computersprache zu übersetzen und einzupflegen. Die Software arbeitet durch die Eingabe des Benutzers die vorgegebenen Abläufe ab und gibt die entsprechende Reaktion als Grafik auf dem Monitor aus. Ein solches Programm lernt daher nicht weiter dazu und kann sich demzufolge nicht weiterentwickeln. Zur Umsetzung wird folglich die Nutzung eines Algorithmus genutzt, um diese Fähigkeit zu einem gewissen Teil zu ermöglichen. Die Maschine lernt insofern mit weiteren Trainingsobjekten und wendet diese dann auf neue vorzufindende Aufgaben an. Daraus wird die stets wahrscheinlichste Lösung im Merkmalsraum gesucht, gespeichert und ausgegeben. Das künstliche Generieren von Wissen aus Eingabedaten ist somit realisiert und verbessert zu gleich den Algorithmus stets weiter.

Die Erfahrungen aus der Vergangenheit leiten somit Rückschlüsse für die Zukunft ab. Es ist beispielsweise bei einem Verkaufsstand von Bratwürsten und Getränken eine Abhängigkeit zum Wetter erkennbar, wodurch unterschiedliche Umsätze zu verzeichnen sind. Bei einem sonnigen und warmen Tag ist voraussichtlich ein hoher Konsum von Lebensmitteln erkennbar. Diese Umwelteinflüsse können sich aber ändern, was sich zwangsläufig auf den Verzehr und deren unmittelbar verbundenen Umsatz auswirkt. Diese Faktoren stehen deshalb in einem zu findenden Verhältnis. Ein statisches Verfahren mithilfe einer linearen Regression ist hierbei beispielsweise nutzbar, um eine bessere Vorhersage über die entsprechenden zu kaufenden Einkaufsmengen auf dem Großmarkt treffen zu können. Eine Gerade wird demzufolge als Beobachtungsgrundlage genutzt. Natürlich ist es ebenso denkbar, dass Störeinflüsse und nicht vorhersehbare Ereignisse keine hundertprozentige Vorhersage zulassen. Dies wird als Unsicherheit des vorzufindenden Modelles benannt und ist daher mit in die Geradengleichung

aufzunehmen. Als Ergebnis erhält man einen zweidimensionalen Parameterraum, der die Verkaufsmenge ins Verhältnis zu den Umwelteinflüssen setzt. Der angesprochene Algorithmus kommt zu tragen, indem dieser die bestmögliche Gerade im Raum findet und darstellt. Es ist folglich eine Bewertungsfunktion zu finden, die mit ihren Vor- und Nachteilen am besten zum Fallbeispiel passt. Betrachtet man also die allgemeine Geradengleichung für die folgendes gilt:

$$y = b \cdot x + a. \quad (2.1)$$

Die Parameter a und b sind so zu wählen, dass es eine möglichst kleine Abweichung in der entsprechenden Bewertungsfunktion gibt. Die Überlegung der Nutzung einer Gerade mit höherer Ordnung ist ratsam, jedoch obliegt dem Programmierer die Entscheidung sowie die Auslegung des Algorithmus. Hiermit ist es ihm dennoch möglich, die Lernfähigkeit und die Vorhersage des Gerätes zu beeinflussen. Es leitet sich die Frage ab, inwiefern man dieser Vorhersage vertrauen kann und ob diese Annahmen auch mit den zukünftigen Ereignissen und den daraus resultierenden Ergebnissen übereinstimmen.

Im Umkehrschluss wird in dem beschriebenen Beispiel immer wieder die Ereignisse mit den Ergebnissen validiert und verglichen, um die besten Parameter für a und b zu erhalten. Es wird daher eine Datenbank benötigt und angelegt, die eine bessere Vorhersage treffen lässt. Der Algorithmus überprüft ständig die Realität mit dem berechneten Wert. Ein Mehrwert in der Zuverlässigkeit ist demzufolge zu erhalten, was das eigene Vertrauen zum Algorithmus stärkt.[6]

2.3 Einflussparameter

Die besten Parameter für eine gute Vorhersage zu finden ist nicht trivial. Die Erfahrung des Programmierers nimmt einen großen Stellenwert ein. Mit seinem erlangten Wissen in diesem Themengebiet ist es ihm dadurch möglich reale Probleme zu lösen. Der erste Schritt der Lösungsfindung stellt die Präzisierung des Sachverhalts dar. Es sind hierfür verschiedene Einzelschritte zu gehen, um die Komplexität aufzuschlüsseln und aufzulösen. Ein System ist daher notwendig, um an die entsprechenden Daten für den Algorithmus zu gelangen und die Einflussparameter herauszufinden. Dafür ist ein Aufbau eines Datensatzes und deren Auswertung unumgänglich, worauf sich der entsprechende Algorithmus stützen kann. Eine Ausnahme bildet hierbei nur das bestärkende Lernen, was aus

Belohnung und Bestrafung seinen Algorithmus verbessert. Es wird hierfür meist eine Simulation verwendet, um die entsprechenden Gegebenheiten zu trainieren.

Der Aufbau eines Datensatzes ist die wesentlichste Herausforderung. Es kann je nach Anwendungsfall aus einer Vielzahl von Bildern bestehen, um einen ausreichenden Trainingssatz zu erhalten. Je nach Lernmethode sind diese noch zu beschriften und zu markieren, um die einzelnen Klassen widerzuspiegeln. Der Algorithmus kann die falsch markierten Erkennungsmerkmale nicht erkennen, was dabei immer zu beachten ist. Es werden ansonsten Besonderheiten angelernt, auf die man ihn gar nicht auslegen will. Des Weiteren ist auf eine gewisse Artenvielfalt zu achten. Im Beispiel einer Hundeklassifizierung sollte also nicht nur eine Hunderasse die Gesamtheit aller widerspiegeln, sondern dem Lerndatensatz sind tunlichst viele verschiedene Arten einzupflegen. Die Qualität und Quantität der Daten sind infolgedessen ein entscheidendes Kriterium.

Für die Umsetzung und Lösung der vorzufindenden Probleme nutzt man die verschiedenen mathematischen Ansätze und wählt die passendsten dafür aus. Nach der Auswahl sind diese immer auf ihrer Glaubwürdigkeit zu überprüfen. Die sogenannten Hypothesentests aus dem Bereich der Statistik bieten sich hierfür an. Daran erkennt man, ob der Algorithmus das richtige Ergebnis liefert und somit für die Anwendung zu verwenden ist. Es sind folgende drei Kriterien zu beachten:

- Wahl des passendsten Algorithmus: Ist ein einfacher Algorithmus für die Problematik ausreichend oder sollte es eher komplexer umgesetzt sein.
 - Wahl der Variablen: Welche Variablen bringen eine Unterscheidung und Zuordnung für das Problem, um die bestmögliche Vorhersage treffen zu können.
 - Güte des Modells: Dies gibt Aufschluss darüber, wie gut das erstellte Modell funktioniert und wie vertrauenswürdig dessen Vorhersagen sind.
- [6]

2.4 Zustände und Rechenleistung

Ein vordefinierter Algorithmus kann zwei Zustände besitzen. Zum einen ist das der Zustand der Lernphase, wobei dieser mit vordefinierten Datensätzen trainiert wird. Der zweite Zustand ist als Betriebsmodus gekennzeichnet, wo der Input klassifiziert und als entsprechende Variable oder Wahrscheinlichkeit ausgegeben wird. Ein

weiterer wichtiger Aspekt ist die Speicher- und Rechenkapazität des anzuwendenden Algorithmus, was einen Einfluss auf die Zeit der Lernphase besitzt. Der Anwender sollte sich daher bewusst sein, dass der Aufbau eines neuronalen Netzwerkes sehr rechenintensiv sein kann. Eine Hilfe hierfür können Anbieter darstellen, die Cloud Rechen-Ressourcen anbieten und demzufolge für zusätzliche Rechenleistung sorgen.

Im nächsten Schritt ist die Häufigkeit des Anlernprozesses festzulegen. Im Grunde gibt es hierfür zwei treffende Optionen. Im ersten Fall ist der Algorithmus angelernt und es kommen keine neuen veränderten Datensätze hinzu. Demzufolge sind schon alle Besonderheiten erlernt, weswegen dieser nicht mit neuen Trainingsdaten trainiert und erweitert wird.

Für Problemstellung in denen sich der Datensatz und die daraus resultierenden Entscheidungen verändern, ist die Hinzunahme von neuen Trainingssammlungen ratsam. Der Algorithmus hat somit die Möglichkeit, sich ständig weiterzuentwickeln und sich den neuen Gegebenheiten anzupassen. Dieser Prozess bezeichnet sich dabei als Parametertuning. Im Zusammenhang mit dem Thema Ausreißer ist abzuwägen, ob diese einen Einfluss auf den bestehenden Algorithmus nehmen und die Option beizubehalten ist.

Im Betriebsmodus sieht es anders aus, welcher im Verhältnis dazu gesehen eher wenig Rechenleistung benötigt. Trotzdem sollte auch dies bei der Auslegung der Komponenten berücksichtigt werden, um keine zeitkritischen Situationen hervorzurufen.[6]

2.5 Digitale Bildaufnahme

Ein Mensch nimmt seine Umgebung mittels seiner Augen auf, was erst durch das Gehirn zu einem Bild zusammengefasst wird. Eine Orientierung in der Umwelt und das Umsetzen von auftretenden Aufgaben wird diesem daher erlaubt. Das Auge nutzt hierfür das physikalische Prinzip der Sammellinse, wodurch ein davor befindliches Objekt auf die Netzhaut projiziert wird. Mit der Projektion durch Spiegelung um die optische Achse ist die Wiedergabe in Originalgröße unmöglich. Grund dafür sind die entsprechenden einfallenden Lichtstrahlen, welche von der Linse gebrochen werden. Nachfolgend wird hierbei in drei charakteristische Strahlengänge unterschieden, welche in einem gemeinsamen Gegenstandspunkt beginnen. Der erste Strahl verläuft dabei parallel zur optischen Achse und wird

durch die Linse in den Brennpunkt gebrochen. Der Brennpunkt beschreibt den Punkt auf der Achse, wo alle vor der Linse befindlichen parallellaufenden Strahlen gebrochen werden und in diesem zusammentreffen. Der Abstand zwischen diesem und der Linse wird als Brennweite bezeichnet, was in Abbildung 2.1 ersichtlich wird [7].

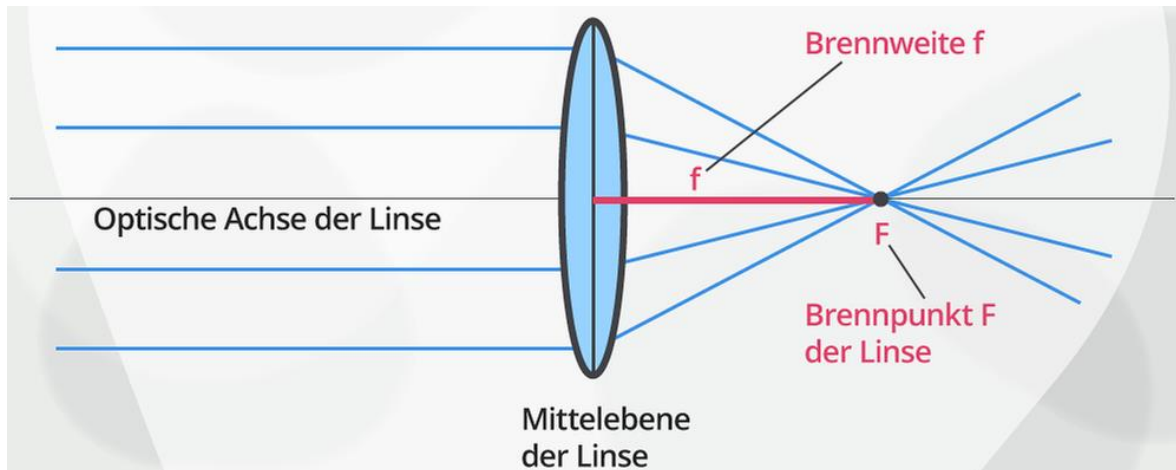


Abbildung 2.1: Grafik zur optischen Linse [8]

Spiegelt man diesen Punkt um die Mittelebene der Linse, dann erhält man einen weiteren Brennpunkt. Durch diesen verläuft ein weiterer Strahl und wird erst nach der Linse parallel zur optischen Achse gebrochen. Der letzte Strahl verläuft direkt durch die Mitte der Linse und behält dabei seinen Einstrahlungswinkel. Infolgedessen treffen diese drei Strahlen nach der Linse wieder zusammen, was mit dem schon beschriebenen Effekt der Größenänderung behaftet ist und in Abbildung 2.2 verdeutlicht wird.

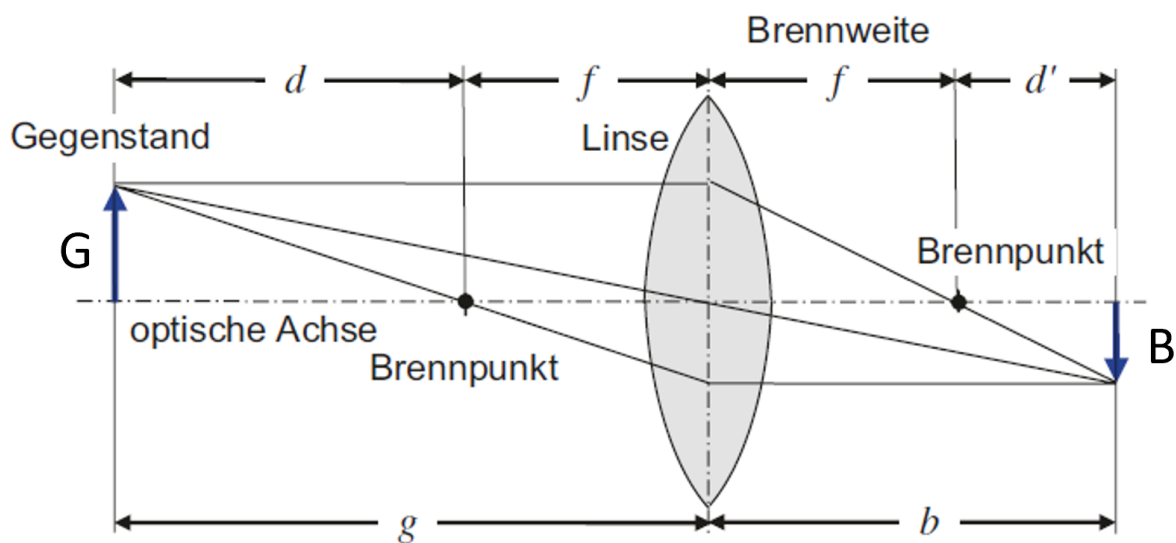


Abbildung 2.2: Dreiecksbeziehung einer Linse [9]

Die unbekannte Bildgröße (B) ist über eine entsprechende Dreiecksbeziehung zu bestimmen. Dafür ist der Gegenstandsabstand (d) und die Brennweite (f) zu ermitteln, um den Abstand d' zu erhalten.

$$d' \cdot d = f^2 \quad (2.2)$$

Die vorzufindenden Abstände sind ebenfalls ermittelbar, wodurch man die schon weitverbreitete und bekannte Linsengleichung erhält.

$$\frac{1}{b} + \frac{1}{g} = \frac{1}{f} \quad (2.3)$$

Die gesuchte und projizierte Bildgröße (B) lässt sich aus den Verhältnissen bestimmen, wobei die Abstände und die Brennweite als Berechnungsgrundlage dienen.

$$\frac{B}{G} = \frac{b}{g} = \frac{f}{d} = \frac{d'}{f} \quad (2.4)$$

Eine weitere wichtige Größe ist dabei die Blendenzahl, welche das einfallende Licht ins Objektiv begrenzt und daher als Blendenöffnung bezeichnet wird. Es handelt sich hierbei um einen meist ringförmigen Durchgang, welcher verstellbar ist und sich in der Nähe der entsprechenden Linse befindet. Im komplett geöffneten Zustand findet keine Beschränkung statt und eine maximale Beleuchtungsstärke wird erreicht. Diese Einstellungsmöglichkeit wird vor allem bei verschiedenen Witterungsverhältnissen notwendig, wodurch ein neutrales Foto zu erzeugen ist. Das Bild kann bei hellen Umgebungsbedingungen und einer breiten Öffnung sehr überbelichtet sein, was an den vielen weißen Pixelanteilen zu erkennen ist. Eine Reduzierung der Blendenöffnung wird infolgedessen nötig, um die Lichtanteile zu begrenzen. In dunkleren Umgebungen wird wiederum viel Einfallslight benötigt, weswegen eine große Öffnung einzustellen ist. Im Gegensatz dazu sind bei einem sehr unterbelichteten Bild viele dunkle Stellen erkennbar. In der digitalen Bildverarbeitung wird für die Darstellung ein sogenanntes Histogramm erstellt, was in der Abbildung 2.3 im oberen Bereich zu finden ist. [9]

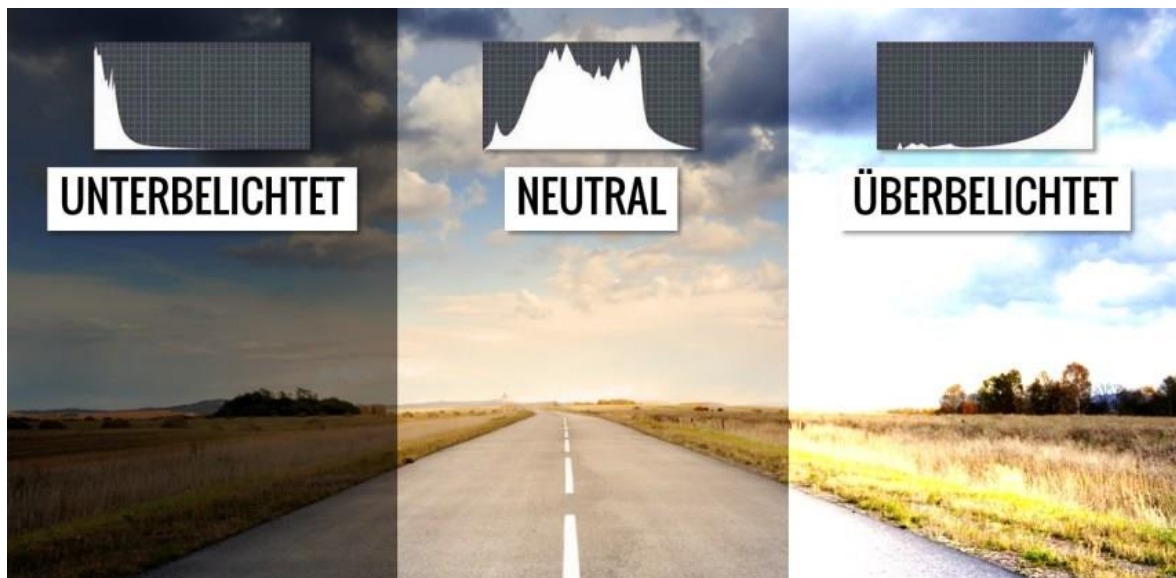


Abbildung 2.3: Veranschaulichung Histogramm [10]

Das Histogramm beschreibt die Tonwerte eines Bildes. Ein Tonwert ist die stufenweise Unterscheidung eines hellen oder dunklen Anteiles eines Bildpunktes, was wiederum auf der horizontalen Achse angetragen ist. Begonnen wird links mit den dunkleren Anteilen. Die vertikale Achse gibt die Anzahl der vorzufindenden Tonwerte an. Diese Darstellung verdeutlicht ebenfalls eine Unter- oder Überbelichtung, obwohl dabei das Ausgangsbild nicht rekonstruiert werden kann [11].

3 Stand der Technik

Die Objekterkennung beschreibt die unterschiedlichen Methoden, um beispielhafterweise ein Gegenstand im vordefinierten Raum zu erkennen. Die Detektion kann mit optischen, akustischen oder auch anderen Auffindungsverfahren erfolgen. Die vorzunehmende Beurteilung ist folglich für die verschiedenen Ansätze von Algorithmen vorzunehmen. [6]

3.1 Überblick Begrifflichkeiten

In diesem Tätigkeitsbereich treten immer wieder Nomenklaturen auf, die nachfolgend näher erläutert und eingeordnet werden sollen. Als großer Überbegriff dieses Themengebietes ist die künstliche Intelligenz zu nennen. Sie umfasst die komplette Informationsverarbeitung, wobei der Mensch als Vorbild genommen wird. Hierbei untergeordnet ist das maschinelle Lernen zu nennen, was die entsprechenden Algorithmen für die Datengewinnung beschreibt. Das menschliche Gehirn wird dabei versucht nachzubilden, um die entsprechenden Daten aufzunehmen und daraus Besonderheiten zu erlernen. Diesen Aufbau der neuronalen Netze und die damit verbundene tiefere Auswertung wird schlussendlich als Deep Learning bezeichnet. Die Abbildung 3.1 soll diesen Zusammenhang verdeutlichen [12].

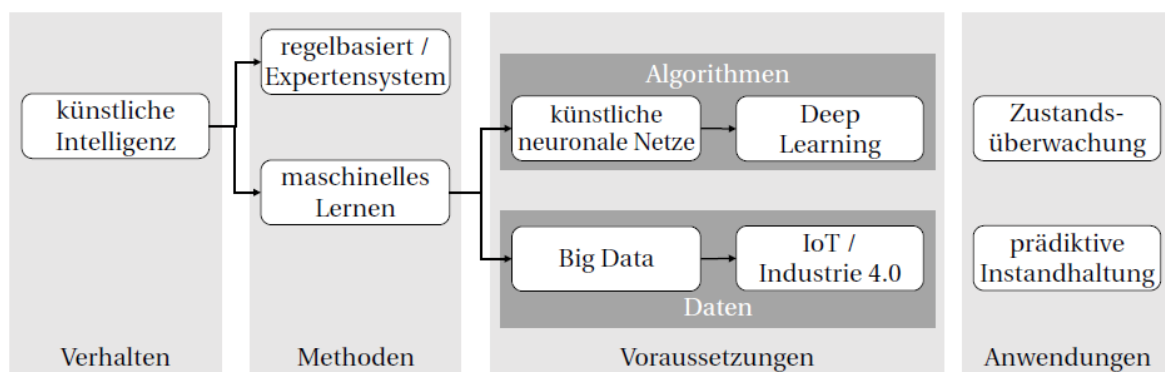


Abbildung 3.1: Aufbau künstliche Intelligenz [13]

Das Ziel ist die Übernahme einer Aufgabenstellung in die computergestützte Umgebung. Für die Bewältigung dieser Anwendung wird ein Trainingsdatensatz benötigt, der die entsprechenden Anlernparameter besitzt und einen Algorithmus antrainiert. Er soll beispielhaft verschiedene Verkehrszeichen auf einem Eingabebild erkennen und folglich unterschiedlich kennzeichnen. Diese Fähigkeit wird dabei als Generalisieren bezeichnet und gibt dem Anwender die Möglichkeit

einer Klassifizierung. Es kann aber ebenfalls passieren, dass dieser die genauen Eigenschaften erlernt hat und demzufolge keine anderen Positionen des Verkehrsschildes mehr zulässt. Dieser Zustand wird als Übertrainieren bezeichnet und sollte tunlichst vermieden werden [13].

3.2 Einsatzgebiete und Bewertungsmöglichkeit

Grundsätzlich gibt es drei verschiedene Ansätze, um ein Algorithmus zu trainieren. Beim überwachten Lernen lernt der Algorithmus aus Ein- und Ausgabedaten, wobei der korrekte Funktionswert zur Eingabe vorgegeben wird. Das Ziel sind dabei Assoziationen herzustellen, was durch mehrere Lerndurchläufe erreicht wird. Die zu erwerbenden Besonderheiten sind in den Anlerndatensätzen enthalten, sodass der ausgewählte Algorithmus diese antrainieren kann. Das Verfahren wird deshalb in Problemstellungen angewendet, wo die Klasseneinordnung schon vorgegeben ist. Die Bilder sind dabei in irgendeiner Form gekennzeichnet, wo beispielhaft ein Hund vorzufinden ist. Die Fotosammlung und eine üblicherweise dazugehörige Informationstabelle geben die korrekte Zuweisung und Position der Tiere wieder. Mit Zuhilfenahme eines beliebigen Testbildes kann dieser Algorithmus überprüft werden. Folglich können dabei vier Ergebnisse auftreten:

- richtig positiv (rp): Es ist ein Hund auf dem Testbild vorhanden und dieser wird so erkannt.
- richtig negativ (rn): Es ist kein Hund auf dem Testbild vorhanden und es wird kein Hund ausgegeben.
- falsch negativ (fn): Ein Hund ist auf dem Bild zu sehen, aber dieser wird nicht erkannt.
- falsch positiv (fp): Es ist kein Hund zu entdecken, aber es wird einer auf dem Bild erkannt.

Mit dieser Grundlage ist es möglich die Treffergenauigkeit zu bestimmen, die wiederum von der Gesamtanzahl der getesteten Bilder abhängt.

$$\text{Treffergenauigkeit} = \frac{rp + rn}{\text{Anzahl der getestenden Bilder}} \quad (3.1)$$

In gewissen Anwendungsfällen kann die Prämisse anders verteilt sein, sodass die Sensitivität (Richtig-positiv-Rate) oder Spezifität (Richtig-negativ-Rate) herangezogen und berechnet wird.

$$\text{Sensitivität} = \frac{rp}{rp + fn} \quad (3.2)$$

$$\text{Spezifität} = \frac{rn}{rn + fp} \quad (3.3)$$

Die meisten Anwender interessieren sich aber eher die Wirksamkeit und der damit verbundene positive Anteil der klassifizierten Objekte.

$$\text{Wirksamkeit} = \frac{rp}{rp + fp} \quad (3.4)$$

Die Trennfähigkeit gibt die negativ klassifizierten Objekte wieder.

$$\text{Trennfähigkeit} = \frac{rn}{rn + fn} \quad (3.5)$$

Auf das Beispiel übertragen wird bei einer Sensitivität von 100 Prozent immer ein Hund erkannt, obwohl vielleicht Testbilder ohne gesuchtes Objekt vorhanden sind. Die Spezifität erreicht im Umkehrschluss einen Wert von 0 Prozent, wodurch keine richtig negative Zuordnung stattfindet. Bei einem Wechsel des Resultates in die anderen Extrema ist wiederum nie ein Hund auffindbar. Es ist demzufolge festzuhalten, dass keiner der beiden Zustände erstrebenswert ist. Eine Ausgewogenheit ist daher anzustreben. Dies lässt sich mit der F1-Kennzahl berechnen, was wiederum ein weiteres Kriterium ist und die vorzufindende Harmonie beschreibt.

$$F1 = 2 * \frac{\text{Sensitivität} * \text{Wirksamkeit}}{\text{Sensitivität} + \text{Wirksamkeit}} \quad (3.6)$$

Die Kennzahl der Harmonie beinhaltet nicht alle vier aufzufindenden Ergebnisse eines Testprozesses und neigt bei nichts symmetrischen Trainingsdaten zur Qualitätsüberschätzung, weshalb sie in der Kritik steht und sich ein anderes Kriterium durchgesetzt hat. Es handelt sich hierbei um den Matthewsche Korrelationskoeffizient (MCC), welcher alle vier Ergebnisse einbezieht und bei einem Wert von 1 eine perfekte Klassifikation ausgibt [14].

$$MCC = \frac{rp * rn - fp * fn}{\sqrt{((rp + fp) * (rp + fn) * (rn + fp) * (rn + fn))}} \quad (3.7)$$

Eine Alternative zum gekennzeichneten Trainingsdatensatz ist das unüberwachte Lernen, bei denen die Daten keine Klassifizierungsvorgabe erhalten. Die direkte Einflussnahme für die angestrebte Zielbewältigung des Algorithmus ist demzufolge nicht gegeben. Das Auffinden von Strukturen und Besonderheiten liegt im Vordergrund. Dementsprechend kommt das Clustering hierfür zum Einsatz, um bestimmte naheliegende Besonderheiten von Daten einzugruppieren. Die notwendigen Klassen werden dabei selbstständig gefunden und erstellt, wodurch keine direkte Kontrolle erfolgen kann. Erst nach Beendigung des Anlernprozesses wird durch Testfälle die Klassifizierung und Einordnung überprüft. Das unüberwachte Lernen fungiert daher indirekt und erreicht nicht immer im ersten Prüfzyklus das erwünschte Resultat. Dies liegt vor allem an der fehlenden Kennzeichnung und Klassifizierungsmöglichkeit im Anlernprozess, wodurch dem Algorithmus keine Stützstellen gegeben sind. Die selbstständig erkannten Strukturen bilden somit ihre eigenen Klassen, was bei bestimmten Problemen zu Lösungen führen kann. Diese können Unterscheidungsprobleme in einem neuen Themengebiet sein, wodurch sich hiermit Lösungsvarianten für den neuen Datensatz ergeben. Die Unterschiede sind damit auffindbar. Des Weiteren wird eine Reduzierung des Zeitaufwandes durch die nicht stattfindende Kennzeichnung der Trainingsdaten erreicht, was mit der Gefahr eines nicht repräsentativen Datensatzes verbunden ist. Als Beispiel ist das Kaufverhalten der Kunden im Supermarkt zu nennen. Eine Abschätzung der zusammengekauften Produkte wäre ohne diesen Ansatz nicht denkbar, um ein entsprechendes Marketing ausrichten zu können.

Zum Schluss ist das bestärkende Lernen zu nennen. In manchen Problemstellungen gibt es keine ersichtliche beste Strategie, weswegen eher auf die Erfahrung gesetzt wird. Mit Zuhilfenahme dieses Lernverfahrens ist eine Erkennung einer erfolgreichen Vorgehensweise des Algorithmus und dessen Bestärkung möglich. Die positiven Entscheidungen entwickeln diesen immer weiter, wodurch sich eine Lösung der Aufgabe entwickelt. Es entspricht einem belohnenden Modell, wobei der Input eine Rückmeldung zur Folge hat. Diese wird wiederum gewertet und erzeugt einen Erfahrungsgewinn des Algorithmus. Eine mathematische Belohnungsfunktion ist hierfür integriert und ermöglicht diese Verstärkung. Dies lässt sich nur durch viele Inputinformationen und Datensätze erreichen. Die benötigten Regeln und Zuordnungen kristallisieren sich dabei

heraus, was mit einer enormen Rechenleistung des genutzten Computers verbunden ist. Zum Einsatz kommt dies in der Robotik, wo die davor festgelegte Aufgabenstellung in vielen Trainingssituationen angelernt wird.

Üblicherweise wird die Einsatzbedingung in einer virtuellen Umgebung nachgebildet, um diesen Prozess umzusetzen. Bei Fehlentscheidungen führt dies nicht zu Beschädigungen des Roboters, was eine Kostenreduzierung in der Entwicklung zur Folge hat. Ein weiterer Vorteil ist die ständige Möglichkeit der Anpassung von Szenarien, wodurch eine Flexibilität erreicht werden kann. Erst wenn dies erfolgreich absolviert und angelernt ist, kann ein Test in der Realität erfolgen. [6]

3.3 Erkennungsalgorithmen

In vielen Fällen wird ein Algorithmus eines maschinellen Lernens als KI - Technologie beschrieben. Dieser Begriff ist nicht definiert, aber viele Unternehmen vermarkten mit dieser Aussage ihre neuen entwickelten Produkte. Der Grund dafür ist die Assoziation der Kunden mit einer zukunftssträchtigen und neuartigen Technik. Im Hintergrund verrichten aber hierbei verschiedene Ansätze von Algorithmen eine Aufgabe, die in ihren Funktionsweisen nachfolgend beschrieben werden.

3.3.1 Funktionsweise eines CNN-Algorithmus

In diesem Abschnitt wird ein neuronales Netz für die Informationsverarbeitung genutzt, was dem Aufbau eines menschlichen Gehirns ähnelt. Die stattfindenden Berechnungen basieren auf mathematisch beschriebene Matrizen. Im Überblick gesehen besteht dieses Netz aus drei Hauptuntergliederungen bzw. Schichten. Als Erstes ist die Eingangsschicht zu nennen. Diese empfängt die Eingabedaten des Anwenders, was zum einen Trainings- oder Testdatensätze sein können. Danach sind mehrere Zwischenschichten mit Knoten und Verzweigungen zu finden. In ihr wird der Datensatz mit Matrizenrechnungen gefiltert und Besonderheiten extrahiert, wodurch ein Vorhersagewert entsteht. Die Verzweigungen sind folglich meistens mit Gewichtungen verbunden, die sich im Anlernprozess ausbilden oder für die Berechnung im Testfall als Grundlage dienen. Für die Ausgabe des Resultates ist die Ausgangsschicht zuständig. Sie dient als Output und gibt die im Netzwerk gewonnenen Erkenntnisse als Wahrscheinlichkeitswert wieder. Der Nutzer kann dies interpretieren und für seine Auswertung nutzen. Der Aufbau soll in der Abbildung 3.2 ersichtlich sein.

Dieses Netzwerk benötigt für den Anlernprozess Trainingsbilder, um sein künstliches neuronales Netz darauf auszubilden. Eine große Datenbank mit klassifizierten Bildern ist somit notwendig, um die Besonderheiten der entsprechenden Objekte anzueignen. Als typische Vertreter sind dabei folgende Algorithmen zu nennen:

- Ein besonderes Augenmerk gilt den Trainingsdaten und die Auswahl des Netzwerkes, was nicht für jeden Anwendungsfall gleich gut geeignet ist. Des Weiteren kann das Modell neu erstellt oder auf einem schon vorhanden aufgebaut werden. Ebenso ist es möglich ein schon davor trainiertes Netz zu wählen, was den eigenen benötigten Trainingsdatensatz reduziert und zu schnelleren Ergebnissen führt. Der Zeitaufwand wird dadurch herabgesetzt, was einen zusätzlichen Kostenvorteil zur Folge hat. Als negativer Punkt ist die nicht ersichtliche Auswirkung des unbekannten Anlernprozesses im davor trainierten Netzwerk zu sehen, was einen Einfluss auf den eigenen Anwendungsfall haben kann. Eine Testung ist infolgedessen unumgänglich, was bei negativem Resultat einen Neuaufbau der Zwischenschichten fordert. Das Training und die entsprechende Wahl der vorhandenen Optionen sind mit sehr hohen Auslastungen der Hardware verbunden, was wiederum einen leistungsfähigen Computer voraussetzt. [6]

3.3.2 Funktionsweise eines ACF-Algorithmus

Bei einem ACF (Aggregate Channel Features)-Algorithmus wird ein zu erkennendes Bild in verschiedenen Auflösungen heruntergebrochen. Die Merkmalsinformationen bleiben zu einem gewissen Teil bestehen, wodurch die Rechenzeit verkürzt wird. Es bleiben lediglich die wichtigen Informationen für die Beurteilung bestehen.

Der Algorithmus nutzt hierfür ein anderes Lernprinzip, wie es beim CNN-Algorithmus der Fall ist. Zum Beginn wird ein Gradientenhistogramm genutzt, was die entsprechenden Kanten eines Bildes erkennt und folglich die Verteilung der Winkel wiedergibt. Durch eine daraus resultierende Ableitung der vorzufindenden Parameter lässt sich die Größe und Orientierung der Kante festlegen. Dies erfolgt bei jedem vorzufindenden Pixel, was mit einer entsprechenden Gewichtung versehen wird. Im nächsten Schritt wird die Größe des eingelesenen Bildes verändert.

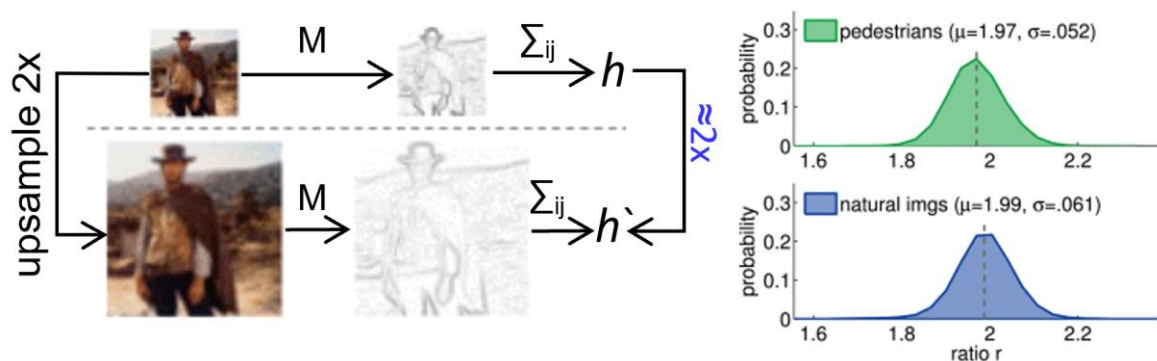


Abbildung 3.3: Größe des Bildes um Faktor 2 erhöht [15]

In der Abbildung 3.3 ist das Bild um den Faktor zwei vergrößert (upsample) und der Gradient (M) bestimmt. Folglich sind die Kanten stärker hervorgehoben, was eine nähere Betrachtung zulässt. Im letzten Durchlauf summiert man diese Bildinformationen auf, was zu den rechts befindlichen Diagrammen in Abbildung 3.3 führt. Bei der Erwartungswertbetrachtung (μ) der Bildbereiche ist der gleiche Faktor der Vergrößerung wiederzufinden. Die Varianz (σ) bleibt entsprechend klein, was einem unverzerrten Bild entspricht. Diese Beobachtung ist für den Menschen (pedestrians) und dem Hintergrund (natural imgs) im Bild ersichtlich. Eine Vergrößerung behält dementsprechend den ursprünglichen Informationsgehalt. Ebenso erhöht es die Anzahl an Pixel, was zu einer längeren Rechenzeit führt. Ein Verbesserungspotenzial mit dieser Maßnahme ist somit nicht ersichtlich.

Eine Betrachtung mit Bildverkleinerung wird somit notwendig, was im Umkehrschluss zu einer Verkürzung der Rechenzeit und der damit verbundenen Kosten führt. Die Fotopixel werden hierbei durch ein Zusammenfassen der Bildpunkte reduziert. In der Abbildung 3.4 ist beispielhaft das Resultat einer Verkleinerung (downsample) mit dem Faktor 2 zu sehen.

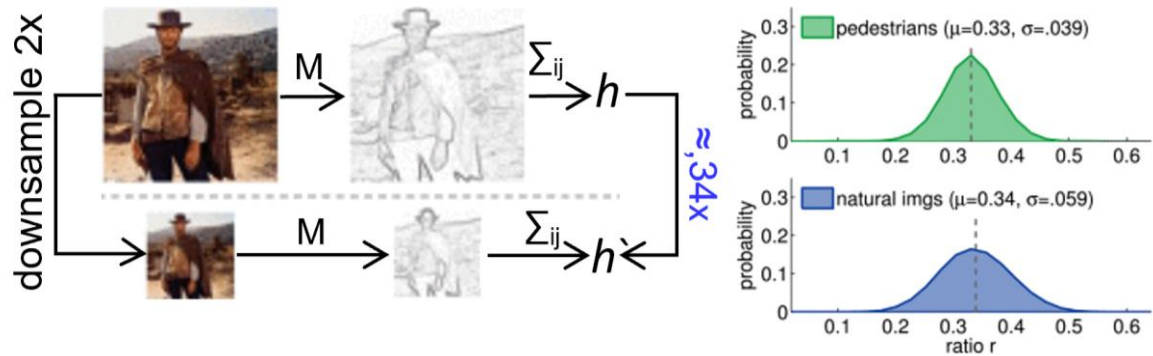


Abbildung 3.4: Bild um Faktor 2 verkleinert [15]

Durch die Bestimmung des Gradientenhistogrammes und der Aufsummierung der Bildinformation mit Erwartungswertbetrachtung wird ersichtlich, dass der Informationsgehalt sinkt. Die Varianz nimmt beim Menschen ab, was auf eine Streuung schließen lässt. Die Besonderheiten bleiben dabei bestehen, weswegen sich dieses für ein Verbesserungspotenzial eignet. Deshalb sind die Kantenabweichungen in den entsprechenden Richtungen zu betrachten.

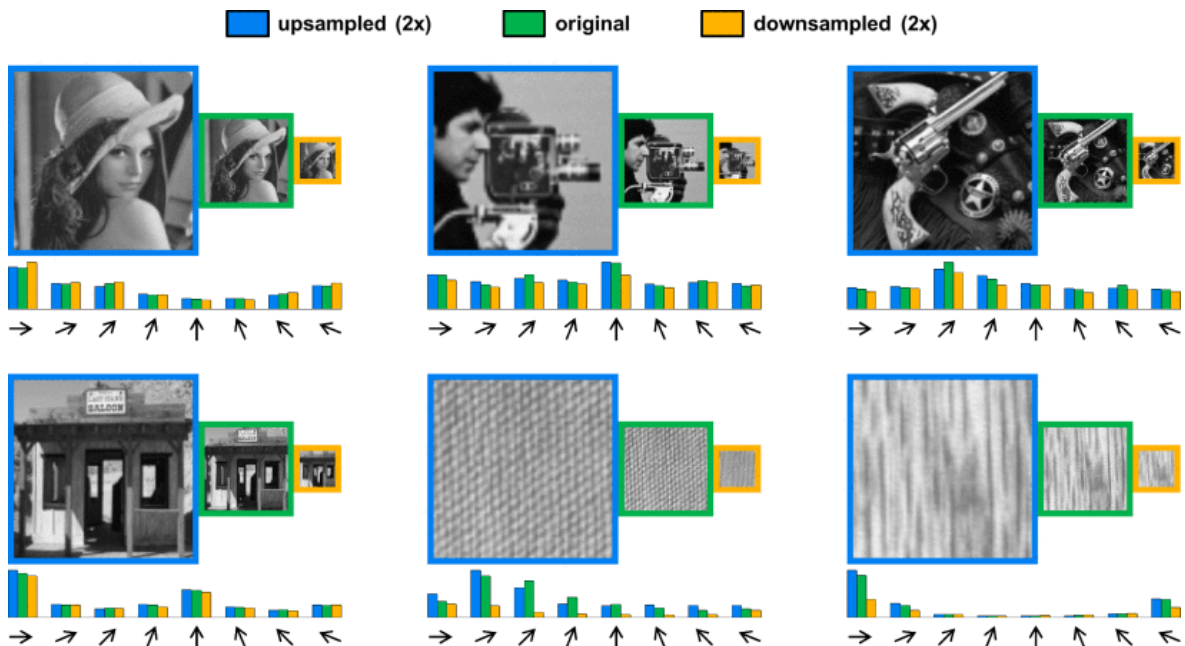


Abbildung 3.5: Kantenabweichung bei Größenänderung [15]

In der Abbildung 3.5 ist das jeweilige Originalbild um den Faktor 2 vergrößert und verkleinert. Nachfolgend sind acht unterschiedliche Richtungen definiert, um deren

Kantenausprägung zu vergleichen. Es sind bei den Bildern von Personen und Objekten keine hohen Informationsverluste erkennbar, weshalb sich diese für den Algorithmus eignen. Lediglich bei Fotos mit atypisch statischen Verhalten ist ein enormer Unterschied bei der Reduzierung der Größenänderung ersichtlich, wodurch sich diese nicht für den Ansatz eignen.

Ein Multiscale-Feature kann darauf aufbauend die Leistungsstärke erhöhen. Es wird dabei als erstes die breite Anzahl von verschiedenen Merkmalen zusammengefügt, um eine verschiebungsinvariante Funktion (Ω) zu erhalten. Diese beinhaltet die tiefen Ebene eines Bildes, was als Kanalbild (C) beschrieben wird. Daraus lassen sich wiederum bestimmte Merkmale extrahieren, um die Kantenverläufe und Farbstatistiken zu erhalten. Mit der Festlegung des Faktors und dem daraus resultierenden Maßstab (s) lässt sich über verschiedene mathematische Ansätze eine Funktion erstellen. Mit dessen Nutzung ist eine schnelle Berechnung realisierbar.

Mit dieser ist eine schnelle Erstellung einer Merkmalspyramide möglich, welche die verschiedenen skalierten Bilder nutzt. Das Ausgangsbild wird mit dem definierten Maßstab reduziert, wodurch es eine Rückführung zulässt. Die Kanten und Merkmale lassen sich somit extrahieren, was wiederum mit Rechenaufwand verbunden ist und keinen Vorteil bietet.

Ein anderer Weg ist eine Kantenbestimmung sofort vorzunehmen. Hierfür wird ein entsprechender Ausgleichswert (λ) notwendig, welchen man in experimentellen Versuchen erlangt. Diese Korrektur erhöht dabei die Genauigkeit. Eine Zusammenführung benachbarter Pixel kann vorgenommen werden, was die Kanten glättet und die Merkmale stärker extrahiert. Dadurch gehen Informationen verloren und die Rechenzeit verkürzt sich, was aber mit einer schnelleren Objekterkennung verbunden ist. Das Originalbild wird zusätzlich nicht in jedem Maßstab herunterskaliert (Proposed Pipeline), was in der Abbildung 3.6 zu sehen ist.

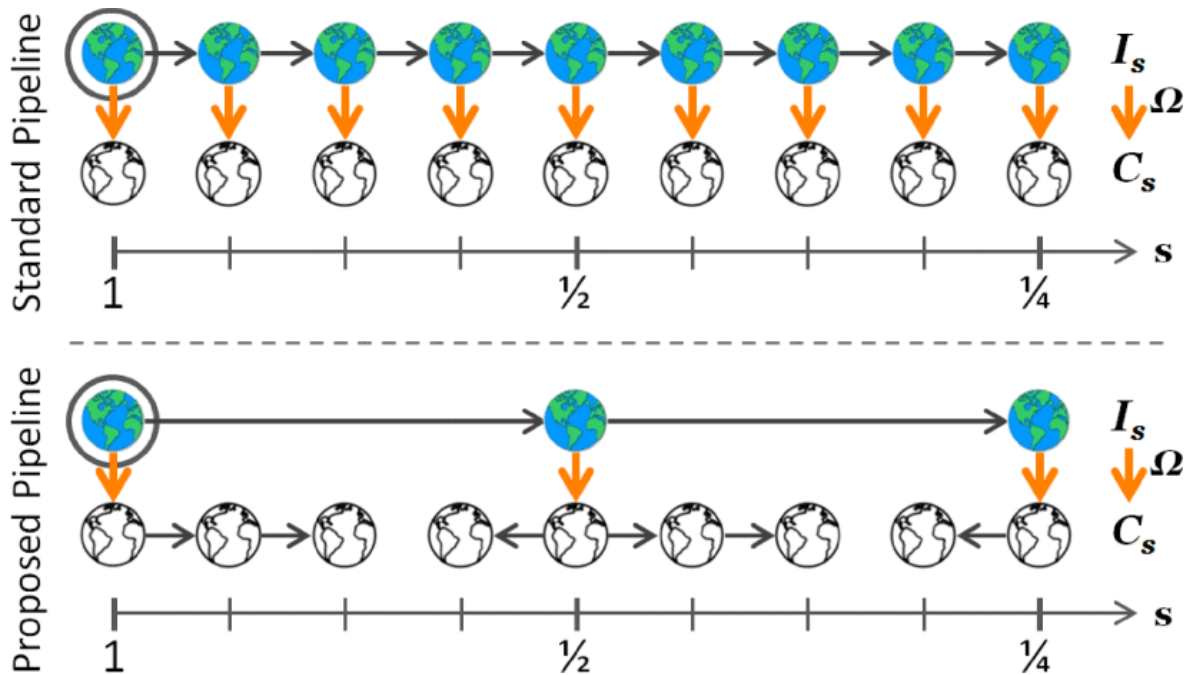


Abbildung 3.6: Rechenzeitverkürzung durch Änderung Sprünge in Maßstäben [15]

Die weitere Sprungerhöhung des vorgegebenen Maßstabs kann somit die Bearbeitungszeit weiter verringern. Die benachbarten Fotos der reduzierten Bilder lassen sich mit einer Annäherungsformel berechnen, was zur Abbildung 3.7 führt.

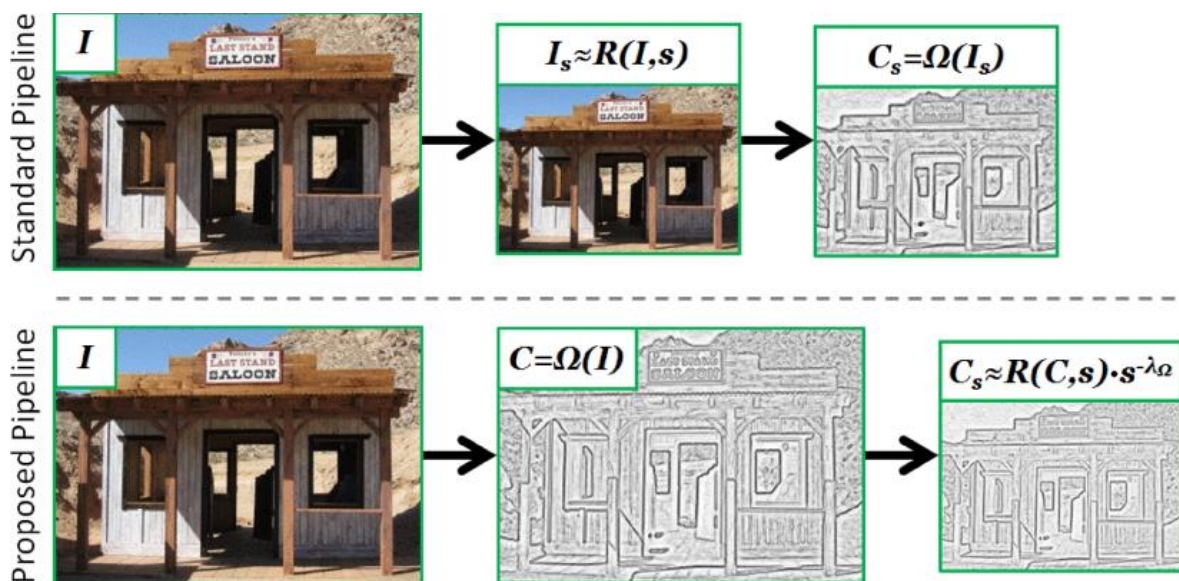


Abbildung 3.7: Annäherungsmöglichkeit für vorgeschlagene Berechnung [15]

Die Genauigkeit nimmt dadurch ab, aber eine weitere Leistungssteigerung des Algorithmus wird erzielt.

Genutzt wird diese Art der Bildverarbeitung bei ACF-Detektoren. Das Eingangsbild wird in mehreren Kanälen extrahiert und an den entsprechenden Stellen geglättet. Der Algorithmus nutzt deren geringste Auflösung und trainiert sich die

entsprechenden Merkmale in Entscheidungsbäume an. Dies ist in Abbildung 3.8 nochmal ersichtlich.

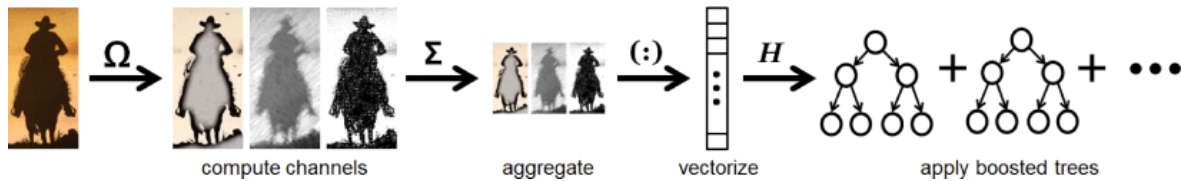


Abbildung 3.8: Ablauf ACF-Detektor bei Eingabebild [15]

Die entsprechenden Objekte sind vom Hintergrund gelöst, was dann wiederum eine Zuordnung und Auffindung zulässt. [15]

3.3.3 Funktionsweise eines Cascade-Algorithmus

Der Cascade-Detektor kann Objekte mit festen Seitenverhältnissen erfassen, weswegen sich dieser für die Erkennung von Verkehrszeichen, Gesichtern und Fahrzeugen eignet. Das Verhältnis kann sich durch eine Veränderung des Blickwinkels zum Objekt verändern, was eine Unbrauchbarkeit zur Folge hätte. Die verschiedenen Ausrichtungen sind demzufolge anzutrainieren, damit eine Detektion auch in diesen Fällen ermöglicht wird. Der Kaskadenklassifizierer kommt für den Anlern- und Testprozess zum Einsatz, welcher aus mehreren Stufen besteht und das entsprechende Eingabemedium durch definierte Bereiche untersucht. Das Ergebnis fällt hierbei positiv aus, wenn etwas erkannt wird. Ist dies nicht der Fall, dann erhält es eine negative Kennzeichnung. In dem entsprechenden Durchlauf sind die Abstufungen so programmiert, dass sie negative Einträge schneller verwerfen und somit nicht weiter betrachten. Das Hauptaugenmerk wird deshalb auf die positiven Bereiche gelenkt [16].

Für das Training eines solchen Erkennungsalgorithmus sind zwei unterschiedliche Bilddatenbanken nötig. Die erste wird hierbei als positive Probe bezeichnet und beinhaltet die entsprechenden Objekte. Diese sind in der Bildsammlung gekennzeichnet, um sich die entsprechenden Merkmale antrainieren zu können. Der zweite Datensatz besteht aus negativen Proben, wodurch bestimmte Muster generiert werden können. In diesen sollten ähnliche Objekte vorhanden sein, um eine Einflussnahme gegen eine Verwechslungsgefahr vorzunehmen. Eine Reduzierung von fehlerhaften Detektionen ist somit erreichbar [16].

Betrachtet man als Beispiel die Gesichtserkennung, dann müssen möglichst viele Fotos mit und ohne vorhandenes Objekt in Ordnern abgespeichert sein. Der

entsprechende Klassifikator kann die vorzufindenden Merkmale antrainieren, was mit Zuhilfenahme der einzelnen Pixelwerte erfolgt. Es werden dabei markante dunkle Bereiche als schwarze Rechtecke gekennzeichnet, wodurch sich beispielsweise die Augen wie in Abbildung 3.9 aus dem Gesicht extrahieren lässt.

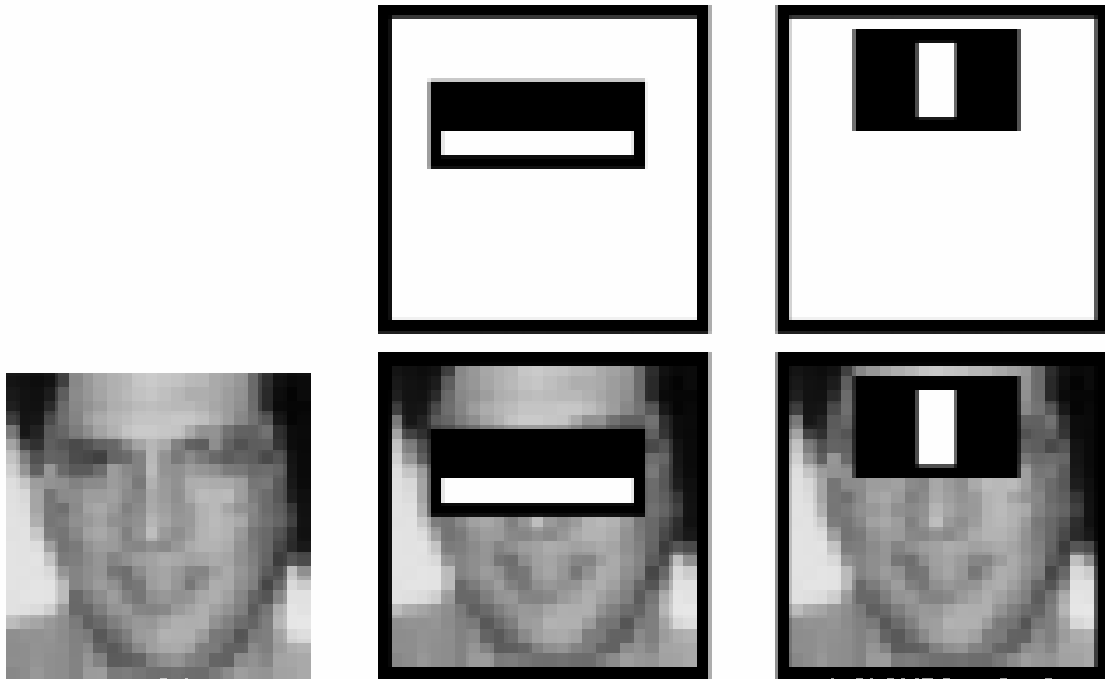


Abbildung 3.9: Funktionsprinzip Cascade-Detektor [17]

Mit dieser Methode lassen sich auch irrelevante Gesichtsbereiche erkennen, wodurch sich diese unterscheiden lassen. Als Einstellmöglichkeit gibt es einen sogenannten Schwellenwert, welcher die entsprechende Empfindlichkeit reguliert und eine Filterungsmöglichkeit gibt. Unwichtige Bereiche wie der Hintergrund lassen sich somit vernachlässigen. Diese Tatsache steigert somit die Effizienz und gestattet eine effektivere Merkmalsextraktion des anzulernenden Objektes [17].

4 Erstellung einer eigenen Bilddatenbank

Für die Erkennung eines Verkehrszeichens ist eine große Datenbank von Bildern nötig. Diese dient als Grundlage des Trainingsvorgangs der verschiedenen Algorithmen und ermöglicht eine Untersuchung derer. Das für die Diplomarbeit bereitgestellte Fahrzeug besitzt eine herstellerspezifisch angebrachte Frontkamera. Ein Zugriff auf diese ist nicht umsetzbar, woraus sich die Nutzung einer Alternative für die Erstellung einer eigenen Bilddatenbank ableitet.

4.1 Auswahl Kamera

Es ist demzufolge notwendig eine geeignete Kamera für den Einsatz zu finden, die ebenfalls flexibel in jedes Automobil angebracht werden kann. Damit die Kosten gering ausfallen und das vorhandene Equipment ausgelastet ist, wurde der Laborbestand der Hochschule für Technik und Wirtschaft Dresden begutachtet. In Anbetracht des Einsatzes fiel die erste Wahl auf eine GoPro. Dieser ist es möglich Bilder und Filme während der Fahrt aufzunehmen und auf einem Speichermedium abzuspeichern. Eine Liveübertragung in das Computerprogramm Matlab ist als Voraussetzung nicht umsetzbar. Eine zusätzliche Alternative ist demzufolge notwendig. Es handelt sich hierbei um eine Industriekamera der Firma Basler. In der Tabelle 4.1 ist ein Auszug der Kennungsdaten zu finden.

Tabelle 4.1: Auszug Datenblatt Basler acA1920-40gc GigE-Kamera

Komponenten	Beschreibung
Shutter-Technologie	Global Shutter
Auflösung	1920 px x 1200 px
Bildrate	42 fps
Farbe	Color
Schnittstelle	GigE

Durch eine zusätzliche Software ist die Einbindung in Matlab ausführbar und eine Datenübertragung garantiert. Wie man der Tabelle 4.1 entnehmen kann, besitzt diese einen sogenannten „Global Shutter“. Es handelt sich dabei um eine besondere Verschlusstechnik, die gewisse Vorteile mit sich bringt. Das aufzunehmende Bild wird bei dieser Technologie gleichzeitig auf allen Zeilen des

Sensors belichtet, was einer Momentaufnahme sehr nahekommt. Es sind infolgedessen alle Pixel gleichzeitig beleuchtet, um das vorzufindende Objekt aufzunehmen. Die Folge daraus ist ein unverzerrtes Foto, was für die Aufnahme von Verkehrszeichen ideal ist. Ein verschwommenes oder unscharfes Bild wird maximal reduziert, um den später zu trainierenden Algorithmus nicht zusätzlich zu beeinflussen. Wie in den anderen Kapiteln schon beschrieben, spielt die Qualität der Fotos eine entscheidende Rolle. Die Schrift des Stoppschildes sollte daher lesbar bleiben, was durch diese Technik zu garantieren ist. Laut Hersteller kann überdies ein ausgezeichnetes Rauschverhalten bei dunklen Situationen erreicht werden, was ebenfalls nicht jede vorhandene Kamera vorweisen kann. Die Aufnahme bei schlechteren Lichtverhältnissen wird demzufolge gleichermaßen kompensiert. Ein Verkehrszeichen bei Dämmerung bleibt somit erkennbar [18].

Als Schnittstelle der Kamera dient die Verbindungsart „GigE“ (Gigabit Ethernet). Es handelt sich dabei um eine einfache Anschlussmöglichkeit von Industriekameras an vorhandene Netzwerke eines Unternehmens. Als Verbindungselement kommt ein Standard-Ethernet-Kabel zum Einsatz, welcher eine hohe Datenrate und Kabellängen von bis zu 100 Metern gestattet. Laut Spezifikationen ist dabei eine Bilddatenschnittstelle von ca. 100 Mbit/s (Fast Ethernet) bzw. 1000 Mbit/s möglich, was eine schnelle Datenübertragung zum Auswertegerät gewährleistet. Ein weiterer Vorteil dieser Technologie ist die gleichzeitige Bereitstellung von Stromversorgung und Datenschnittstelle. Dies wird durch die PoE (Power over Ethernet) Technologie und dem achtadrigen Ethernet-Kabel erreicht, was wiederum der Kamera die Spannungsversorgung bereitstellt [19].

Ein PoE-Switch wird somit erforderlich, der die benötigte Datenrate und Energiezufuhr fürs Netzwerk bereitstellt. Es gibt hierfür zwei angebotene Systeme, die sich in einem aktiven oder passiven Verhalten unterscheiden. Der Vorteil eines aktiven Systems ist, dass dieser einen Computer erkennt und nicht mit einer Spannung versorgt. Es werden dementsprechend nur Daten zwischen dem Switch und dem Computer versendet. Bei einem passiven System erhält der Port des Computers einen Schaden, weil dieser für die Spannungsaufnahme nicht konzipiert ist. Ein aktiver Switch ist demzufolge für die Inbetriebnahme der Kamera notwendig und bei der Firma Netgear zu finden [20].

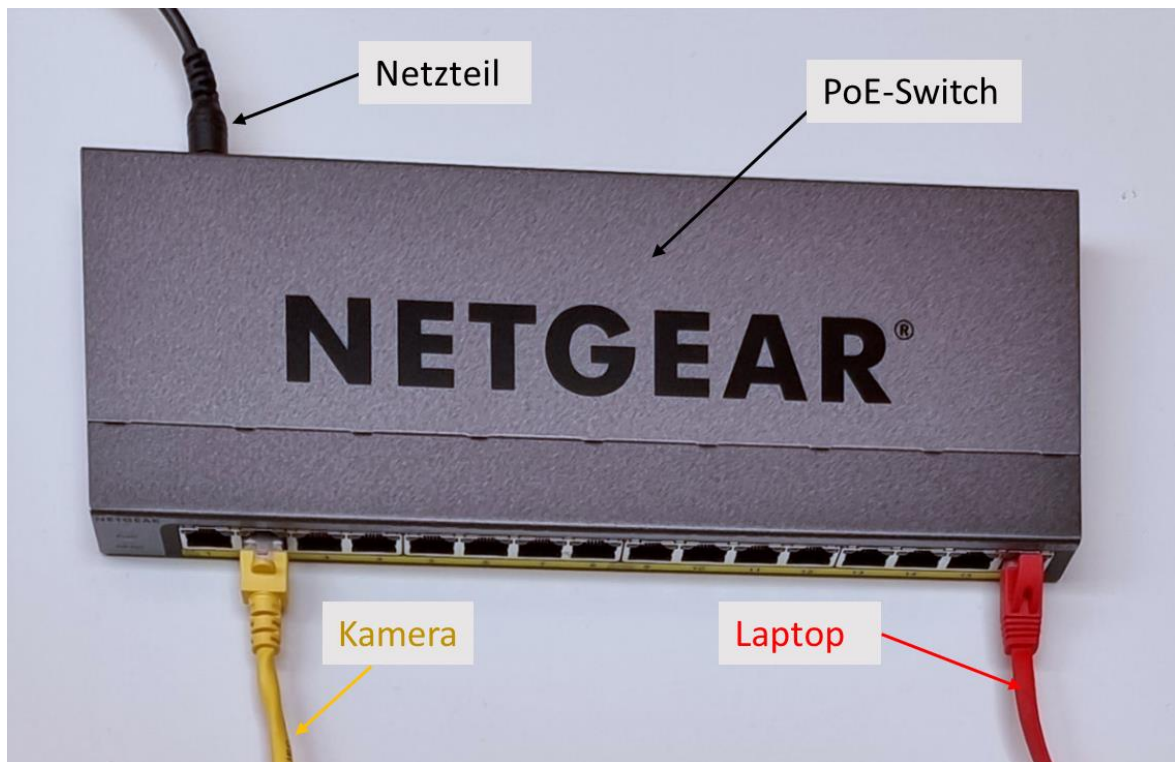


Abbildung 4.1: PoE-Switch mit angeschlossenen Komponenten

Dieser besitzt 16 aktive PoE-Ports und stellt eine Gesamtleistung von circa 76 Watt bereit, welcher in Abbildung 4.1 zu sehen ist. Das Netzteil ist dabei mit 54 Volt zu versorgen, um die entsprechende Leistung abrufen zu können. In dem bereitgestellten Testfahrzeug ist eine Anbindung für die Inbetriebnahme eines PoE-Switchs vorhanden. Dieser konnte folglich die benötigte Spannungsversorgung für den Verteiler bereitstellen [21].

Die Positionierung des Systems sollte im Kofferraum erfolgen. Die vorzufindenden individuellen Halterungsmöglichkeiten können somit genutzt werden, um ein Verrutschen des Switches zu verhindern. Die Kamera ist dabei stets an der Frontscheibe angebracht, um einen breiten Sichtbereich abzudecken und der Witterung nicht ausgesetzt zu sein. Auf dem Beifahrersitz ist der Laptop für die Ansteuerung platziert, der eine Fixierung durch diesen erhält. Die Verbindungen der beiden Geräte mit dem PoE-Switch sind durch zwei längerer Ethernet Kabel hergestellt. Eine Verlegung dieser in die Mittelkonsole und Dachhimmel ist vorteilhaft, um die Insassen nicht unnötig einzuschränken. Das Aufscheuern wird durch zusätzliche Fixierungen verhindert. Der Datenaustausch ist somit realisiert, was in Abbildung 4.2 ersichtlich wird.

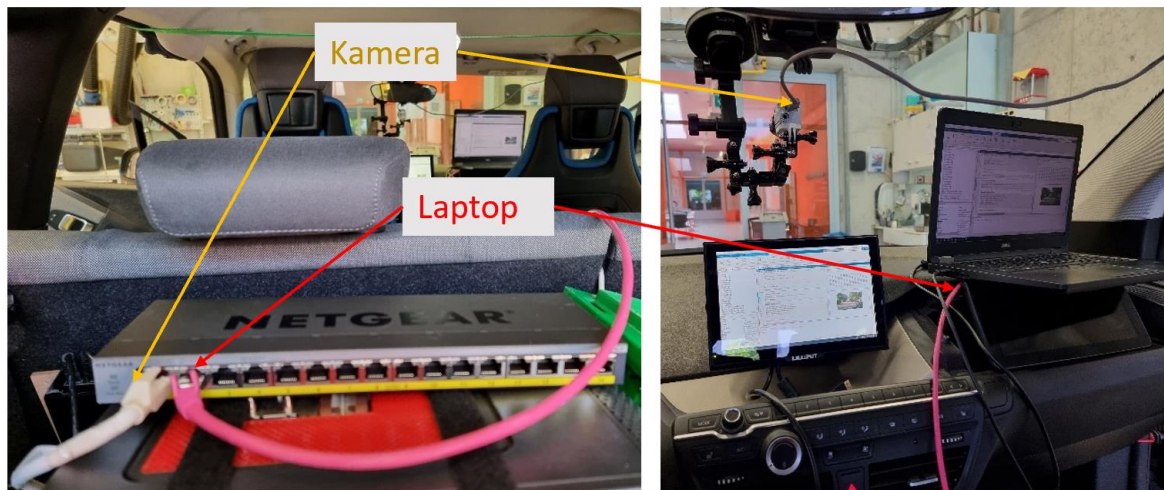


Abbildung 4.2: Position der Hardware im Aufnahmefahrzeug

Eine Halterung sorgt für die sichere Anbringung der Kamera im Fahrzeug. Diese besteht aus einem Saugnapf, was einen rückstandsfreien und festen Sitz garantiert. Die benötigten Verbindungselemente stammen von einer vorhandenen Actionkamera, was in Abbildung 4.3 ersichtlich wird.

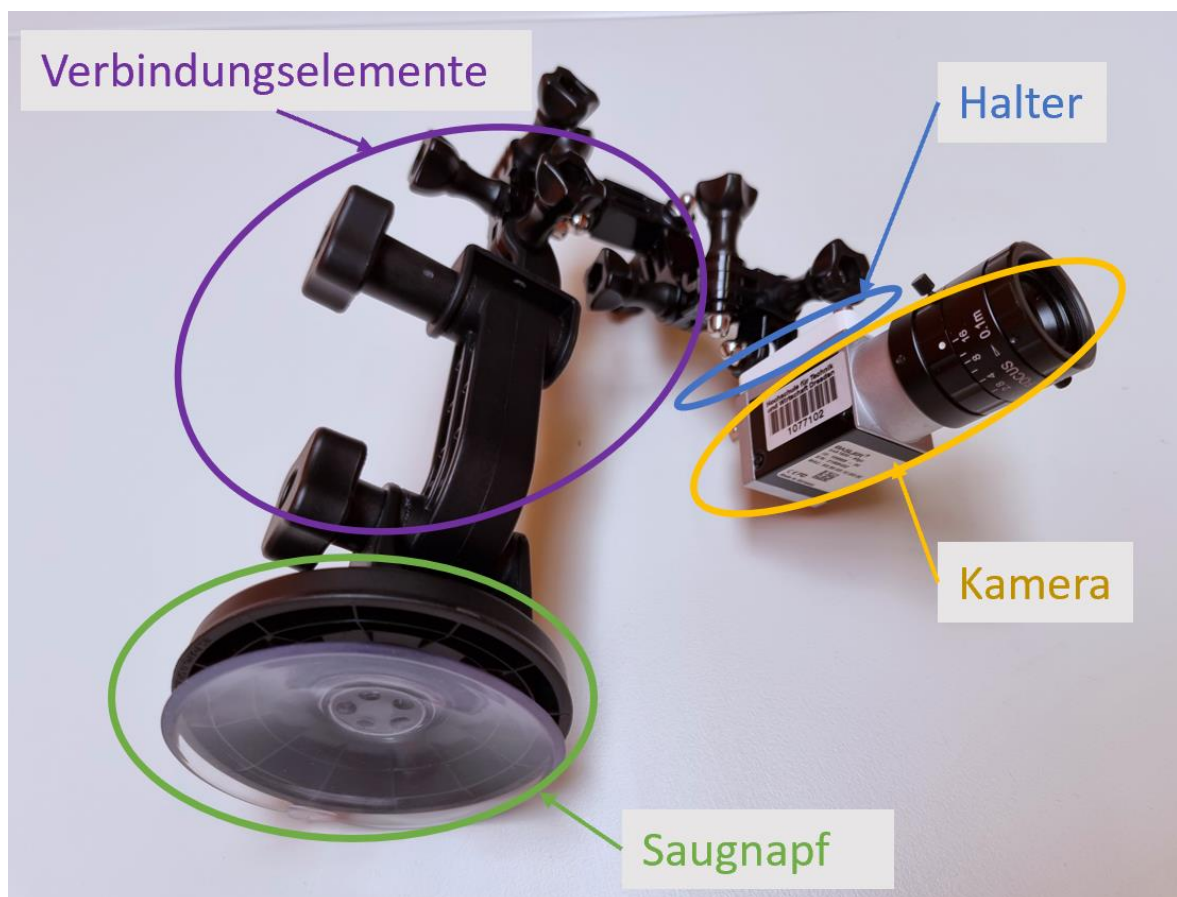


Abbildung 4.3: Elemente für Kamerabefestigung

Die einzelnen Verbindungselemente garantieren hierbei eine exakte Einstellung im dreidimensionalen Raum, um eine Nachjustierung zu ermöglichen. Als Verknüpfung

zwischen Kamera und dem letzten Verbindungselement dient eine in der Vergangenheit hergestellte individuelle Halterung aus dem 3D-Drucker. Eine feste Fixierung und die Möglichkeit der Ausrichtung des Aufnahmegerätes sind gegeben. Aufgrund der enormen Datenmengen ist ein zusätzliches Speichermedium unumgänglich, der sich ebenfalls an dem Computer befindet. Eine komplette Inbetriebnahme und ein Datenaustausch der entsprechenden Komponenten sind somit garantiert, was als Überblick in Abbildung 4.4 zu sehen ist.

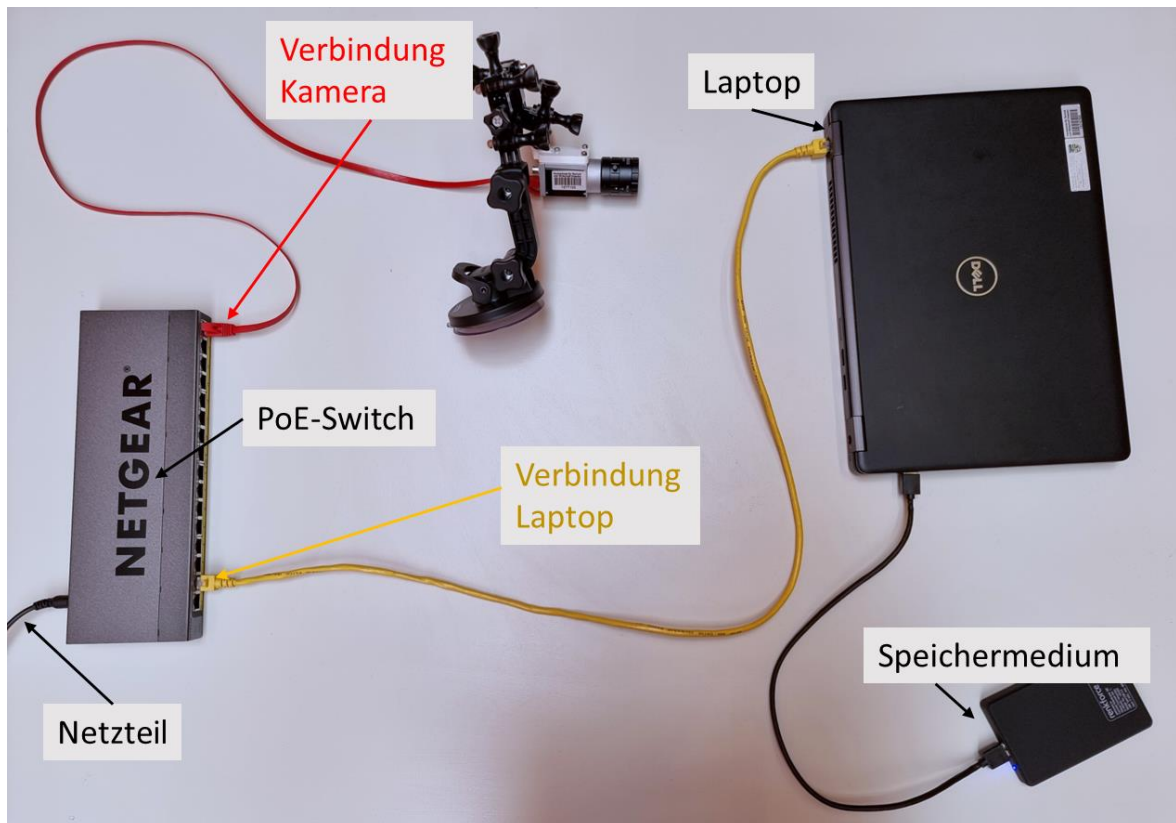


Abbildung 4.4 Komponenten und deren Anbindung im Überblick

4.2 Umsetzung Software

In dem Computerprogramm Matlab ist eine Ansteuerung der Kamera gefordert, um diese zu integrieren und zur Datenaufzeichnung nutzen zu können. Für die Umsetzung ist eine Zusatzsoftware erforderlich, um die Schnittstelle GigE zu nutzen. Eine Verbindung zwischen Hardware und Software ist damit realisiert. Mit der integrierten App „Image Acquisition Toolbox“ kann man die entsprechenden Daten abrufen, um sich beispielhaft Livebilder anzeigen und abspeichern zu lassen. Die Videofunktion erstellt hierbei längere Bildsequenzen, was bei den Aufnahmefahrten notwendig ist und eine Erstellung einer Bilddatenbank in Matlab ermöglicht.

Zusätzlich zu den Bilddaten ist der Abstand zum Verkehrszeichen zu ermitteln und abzuspeichern. Eine zweite parallellaufende Diplomarbeit beschäftigte sich in der gleichen Bearbeitungszeit mit einem Lidar-Sensor. Die Besonderheit des Sensors sind die austretenden unsichtbaren Laserstrahlen, die von Objekten zurückreflektiert werden. Ein daneben befindlicher integrierter lichtempfindlicher Sensor wertet die zurückgestrahlten Reflexionen aus und bestimmt daraus die Entfernung. Als Ergebnis erhält man Bildpunkte im dreidimensionalen Raum, mit denen sich die entsprechenden Abstände ausgeben lassen. Des Weiteren ist dieser am vorhandenen Versuchsfahrzeug verbaut, was eine parallele Messung gestattet.

Ein Parameter für die Zusammenführung der aufgenommenen Bildpunkte und des Bildes ist erforderlich, um eine Zuordnung zu ermöglichen. Als einzige überschneidende Größe hat sich dabei die Zeitsynchronisation herausgestellt. Die entsprechenden Zeitpunkte der einzelnen Bilder sind demzufolge mit Matlab separat abgespeichert, was mit einer individuellen Anpassung der Aufnahmefunktion verbunden ist. Ein eigenes Skript ist somit unausweichlich.

4.3 Programmablauf für Bildaufnahme

Eine Verbindung zwischen Kamera und Computer ist erreicht, wodurch zwingend eine Anpassung der Software nötig ist. Mit der integrierten App „Image Acquisition Toolbox“ können Fotos und Videos aufgenommen werden, die wiederum nur eine einzige Zeitangabe enthalten und somit als Grundlage des eigenen Skriptes dient. Die einzelnen Bildsequenzen im Video und deren Zeitpunkt ist daraus nicht direkt ersichtlich, weshalb eine Anpassung nötig ist. Erst dies ermöglicht eine Fusion zwischen Kamera und Lidar-Sensor.

Im ersten Schritt des neu erstellten Skriptes `Marvin_4_Aufnahme_Video.mlx` sind alle verfügbaren Kameras aufgelistet. Dies gestattet eine flexible Anpassung und Auswahlmöglichkeit, falls in der Zukunft ein anderes Aufnahmegerät zum Einsatz kommt. Die Basler Kamera wird hierbei aufgelistet und lässt eine Nutzung zu. Als Aufnahmeoption ist das Video zu wählen, um mehrere Bilder für die Auswertung und das Training der Algorithmen zu erhalten. Die internen Einstellmöglichkeiten der Kamera können ebenfalls verändert werden, was bei den meisten Herstellern ausführbar ist. Zum einen handelt es sich um die Bildfrequenz, die einen entscheidenden Einfluss auf die Bilddateigröße nimmt. Es sind laut Datenblatt dabei maximal 40 Bilder pro Sekunde ausführbar. In Testvorgängen

stellten sich zwei schwerwiegende Probleme heraus, die wiederum nicht zu vernachlässigen sind. Es ist zum einen die entstandene enorme Dateigröße zu nennen und zum anderen die erheblichen Zeiten zum Abspeichern. Mehrere sehr nahe liegende Verkehrsschilder können somit nicht aufgenommen werden, weil sich die Messung nicht starten lässt und im Vorgang der Speicherung befindet. Die Abstellung der Problematik kann lediglich mit der Herabsetzung der Bilder pro Sekunde erfolgen, wobei ein flüssig abspielendes Video für den Betrachter ersichtlich bleiben soll. Das menschliche Gehirn nimmt mit seinen Augen ungefähr 20 Bilder pro Sekunde als störungsfrei wahr, wodurch dieser als neuer Standardwert gesetzt ist. Die Datenmenge und die Zeit für die Speicherung kann man mit dieser Maßnahme um circa die Hälfte verkürzen [22].

Die Blendeneinstellung ist ebenfalls elektronisch minimal nachjustierbar, was einen weiteren Vorteil der genutzten Kamera darstellt. Üblicherweise wird diese vom Benutzer manuell über ein Drehrad am Objektiv eingestellt, was einen gewissen Aufwand bei den Testfahrten und der Bildnachbereitung bedeutet. Die interne elektronische Blendenunterstützung ist aktiviert, um dies zu umgehen. Diese Einstellung ermöglicht ein internes Aufhellen bzw. Abdunkeln der Bilder während der Aufnahme. Die Blende sollte dennoch manuell veränderbar bleiben, um eine grobe Einstellung vornehmen zu können. Infolgedessen ist ein ständiges Livebild auf dem Computer notwendig, um die Veränderungen zu visualisieren und dem Benutzer eine Rückmeldungsebene zu übermitteln. Eine Einflussnahme bei wechselnden Umwelteinflüssen ist demzufolge gegeben.

Das Skript soll nach Beendigung des Durchlaufs nicht immer wieder neu gestartet werden, wodurch eine weitere Eingabeabfrage im separaten Fenster erscheint. In diesem sind zwei Auswahlmöglichkeiten enthalten. Ist der Button „Neue Aufnahme“ gewählt, dann erfolgt erneute die Abfrage nach dem Start und der Beendigung der Messung. Es ergeben sich daraus wiederum verschieden lange Aufnahmevideos und eine freie Wahl des Startzeitpunkts. Durch die Bestätigung des Stoppvorgangs werden diese in einem separaten Ordner abgelegt und es erfolgt eine erneute Abfrage mit den schon erwähnten Auswahlmöglichkeiten. Wählt der Benutzer die Option „Abbruch“, dann wird das Skript beendet und alle geöffneten Fenster schließen sich.

4.4 Inbetriebnahme Kamera

Wie schon erwähnt ist in dem davor beschriebenen Skript ein ständiges Livebild auf dem Ausgabegerät zu beobachten. Diese Funktion wird ebenso für die Ausrichtung der Kamera genutzt, wodurch eine Nachjustierung im dreidimensionalen Raum ausführbar ist. Die entsprechenden Verbindungselemente lassen sich hierfür mittels Schrauben lösen, was eine Einstellmöglichkeit zulässt. Die Kamera wird auf den Blickbereich des Fahrers eingestellt. Die Blende wird des Weiteren manuell verstellt, was mit dem am Objektiv befindlichen Drehrad gestattet wird. Es kann somit ein unter- oder überbelichtetes Foto entgegengewirkt werden. Je nach Witterungsverhältnis und Tageszeit kann sich die äußerlichen Gegebenheiten ändern, wodurch hiermit entgegengesteuert werden kann. Infolgedessen ist während der Fahrt eine bestmögliche Bildqualität erreicht.

Der programmierte Algorithmus ist in einem ersten Testprozess zu überprüfen. Das vorhandene Prüffeld der Hochschule besitzt kein verankertes Stoppschild, weswegen ein transportables zum Einsatz kommt. Dies gestattet eine freie und individuelle Platzierung, was für die ersten Untersuchungen sehr vorteilhaft ist. Der vorhandene vollelektrische BMW I3 dient dafür als Testfahrzeug. In diesem ist der PoE-Switch vorhanden, um die entsprechenden Computer und Hardwarekomponenten zu verknüpfen und anzusteuern. Hierbei ist der Abstand zum Verkehrsschild ebenfalls zu überprüfen, wodurch der Lidar-Sensor anzusteuern ist. Dieser ist schon fest auf dem Fahrzeugdach verbaut, weswegen hierfür keine weiteren Anpassungsarbeiten notwendig sind. Die beiden Skripte konnten daraufhin ausgeführt und getestet werden. Die Tests verliefen hierbei erfolgreich, weswegen der Gedanke einer Zusammenführung beider Skripte aufkam. Zu diesem Zeitpunkt war das Skript des Lidar-Sensors noch von einem zusätzlichen Computer angesteuert, was sich aber ändern sollte.

4.5 Zusammenführung der Skripte

Die Skripte sollten zusammengefügt werden, um einen Laptop für die Testfahrt einzusparen und den Aufwand zu reduzieren. Es erwies sich als sehr umständlich, zeitgleich beide Skripte auf den Geräten zu starten und wieder anzuhalten. Infolgedessen sollte dies mit einem Start- und Stoppbefehl geschehen, um die Bedienung zusätzlich zu vereinfachen. Die Befehle der Ansteuerung des Lidar-Sensors sind demzufolge in das vorhandene Skript zu integrieren, um eine

Zusammenführung dieser zu ermöglichen. Im Ergebnis können über einen Befehl beide Aufnahmegeräte gesteuert werden. Daraus leitet sich die Notwendigkeit einer weiteren Probefahrt auf dem Testfeld ab, um dieses in der Realität zu überprüfen.

Im Resultat ergeben sich unterschiedlich gemessene Wegstrecken der genutzten Hardware. Die Aufzeichnung der Kamera beginnt ab dem Startvorgang und die des Lidar-Sensors ist hierbei verzögert. Dies ist damit zu begründen, dass der Laserscanner eine gewisse Zeit für die Initialisierung und Aufwärmphase benötigt. Insofern wird die äußere Umgebung erst später aufgenommen, was sich als Problem darstellt. Dementsprechend steht der Gedanke im Raum, zuerst den Lidar-Sensor anzusteuern und danach die Kamera. Dies lässt sich aber ebenfalls im Programmablauf nicht ändern, weil erst nach Beendigung von dessen Schleife die Kamera starten würde. Eine Synchronisation ist somit nicht mehr gewährleistet.

In einem weiteren Versuch ist die Aufnahmedauer des Lidar-Sensors zu erhöhen und die Bildaufzeichnung zu starten. Eine Verkehrszeichenerkennung wird somit von beiden Geräten garantiert. Es stellte sich aber ein neues Problem bezüglich der erstellten Bilddatengröße und Verarbeitungszeit ein. Diese erhöhen sich in einem Bereich, was nicht mehr zu tolerieren ist und keine Aufnahme von mehreren Verkehrsschildern zulässt. Ein weiterer Nachteil sind die damit verbundenen längeren Zeiten des Einlesens beim Auswerten.

Des Weiteren entstand zu einem fortlaufenden Zeitpunkt die Anforderung nach längeren Aufnahmelängen des Lidar-Sensors, was diesen Ansatz schlussendlich nicht als Lösung ermöglicht. Die Umsetzung in ein Skript ruft mehr neue negative Punkte hervor, weswegen die Ansteuerung weiterhin in zwei separaten Skripten verbleibt. Jetzt gilt es noch die Zeitsynchronisation der aufgenommenen Daten zu überprüfen, weswegen der Zeitverzug zwischen Lidar-Sensor und Kamera nötig ist.

Eine Zusammenführung der beiden Datenbanken kann durch die Ermittlung der vorzufindenden zeitlichen Differenz erlangt werden. Diese resultieren aus den unterschiedlichen Skripten, der genutzten Laptops, die Positionen der Sensoren zueinander und dem Ansprechverhalten der gewählten Aufnahmegeräte. Infolgedessen ist eine weitere Testfahrt auf dem Prüffeld notwendig, um diese Abweichung zu ermitteln. Die für die Messfahrt ausgewählte Kamera ist hierbei an der Seitenscheibe des Fahrzeuges angebracht. Es ist bei der Befestigung der Sensoren darauf zu achten, dass diese zueinander eine möglichst geringe

Abweichung in Fahrtrichtung aufweisen. Dieser Aufbau wird in Abbildung 4.5 ersichtlich.

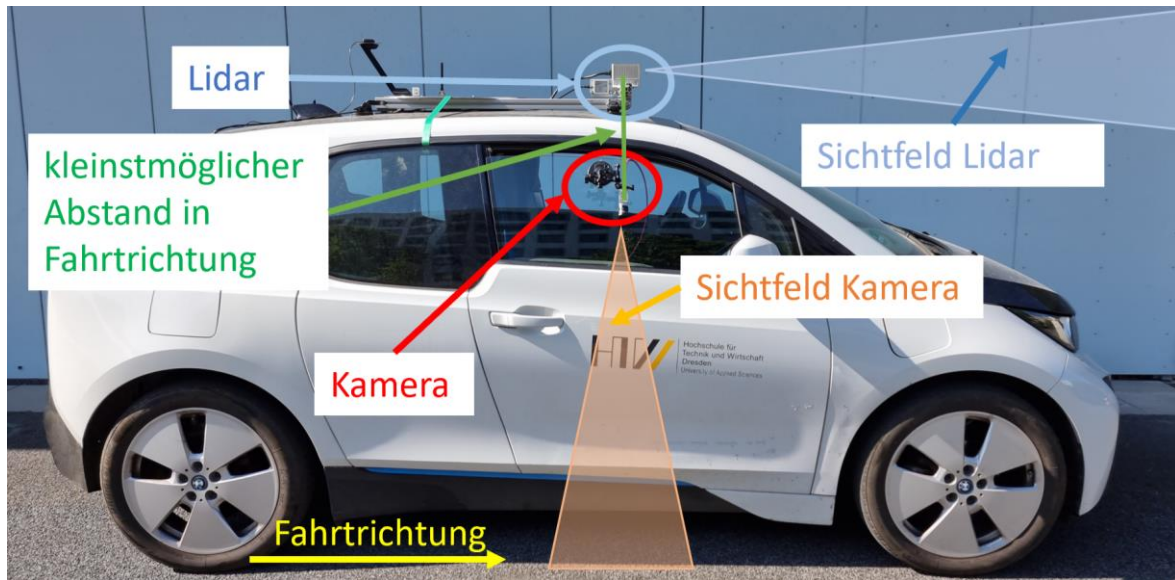


Abbildung 4.5: Messaufbau für Zeitdifferenz

Im nächsten Schritt ist ein mobiles Stoppschild auf dem Prüffeld zu platzieren, damit der Lidar-Sensor einen auswertbaren Gegenstand vorfinden kann. Insofern ist dieser Abstand bestimmbar, wodurch das Skript den genauen Zahlenwert ausgibt. Mit Zuhilfenahme eines Rollbandmaßes und gelber Markierungskreide sind im Abstand von 5 Meter zum Verkehrsschild die entsprechenden Entfernungen auf dem Prüffeld aufgezeichnet. Der maximale Wert beträgt in diesem Testaufbau 25 Meter, was mehrere Messpunkte für die Auswertung ergibt und in Abbildung 4.6 ersichtlich wird.

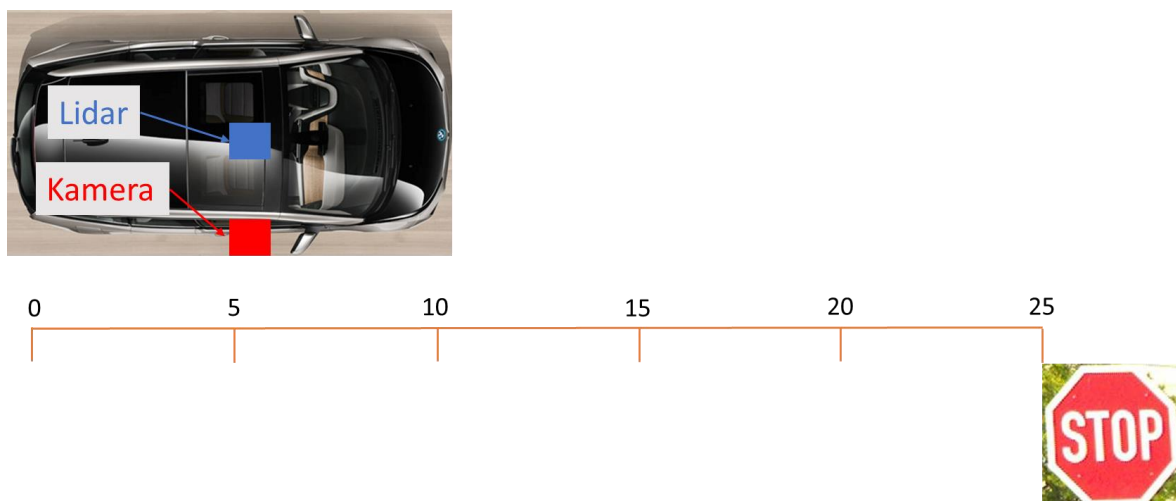


Abbildung 4.6: Testaufbau mit Markierungen auf Prüffeld

Die Kamera nimmt dabei in der entsprechenden Aufzeichnungszeit die Fahrbahn auf, worauf die verschiedenen Entfernungen angetragen sind. In den darauffolgenden Aufnahmefahrten sind beide Sensoren parallel anzusteuern, sodass diese ihr Umfeld zeitgleich aufnehmen. Des Weiteren speichern diese beiden ihren gerade vorzufindenden Zeitstempel ab, was eine Zusammenführung gestattet.

Nach mehreren Aufnahmefahrten konnte die Auswertung der Daten erfolgen. Hierfür ist das Video der Kamera bis zum Messpunkt abzuspielen. Die Bildsequenz ist folglich so auszurichten, dass die Entfernungsangabe der Markierung sich mittig befindet. Demzufolge kann man davon ausgehen, dass in diesem Moment die Kamera exakt über dem Messpunkt steht. Als Nächstes ist der dazugehörige Zeitstempel auszulesen, um die entsprechenden Daten vom Lidar-Sensor zu erhalten. Mit dieser Übermittlung konnten die passenden Punktwolken herausgefiltert werden, um den Abstand zu ermitteln. Folgende Auswertungsdaten sind in Abbildung 4.7 beispielhaft entstanden:

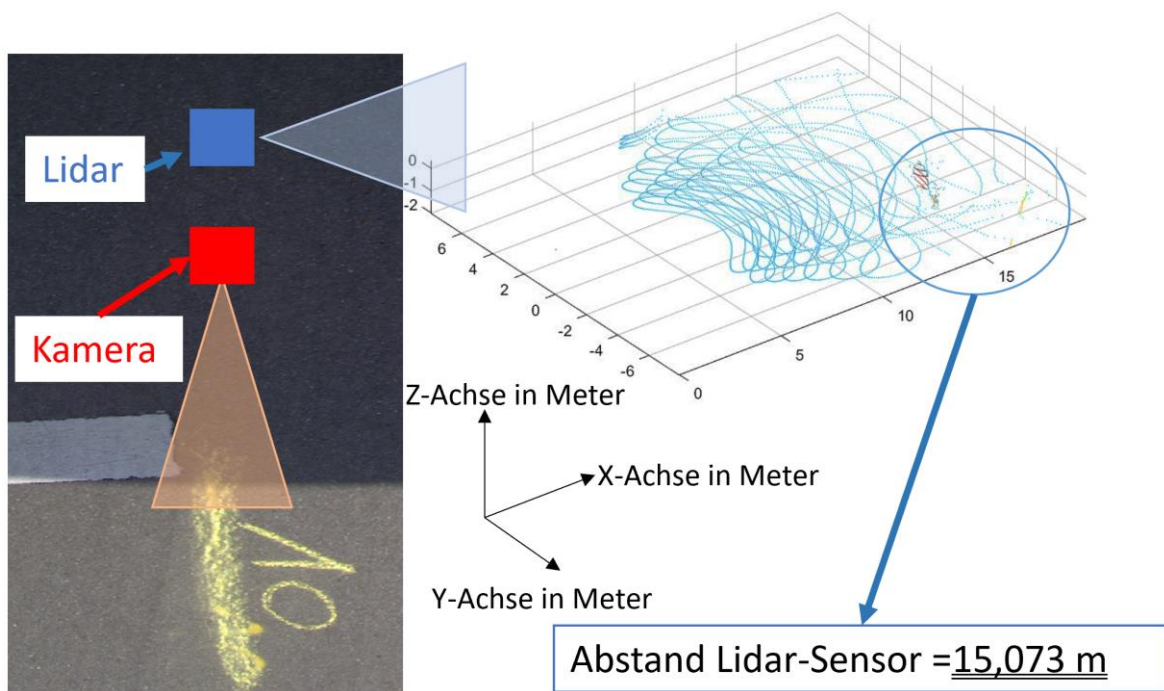


Abbildung 4.7: Differenz Kamera und Lidar-Sensor bei Auswertung

Das Stoppschild befindet sich dauerhaft auf dem 25-Meter-Messpunkt, weswegen die Differenz zum von der Kamera aufgenommenen Bild zu berechnen ist. Dies wird in Abbildung 4.8 ersichtlich.

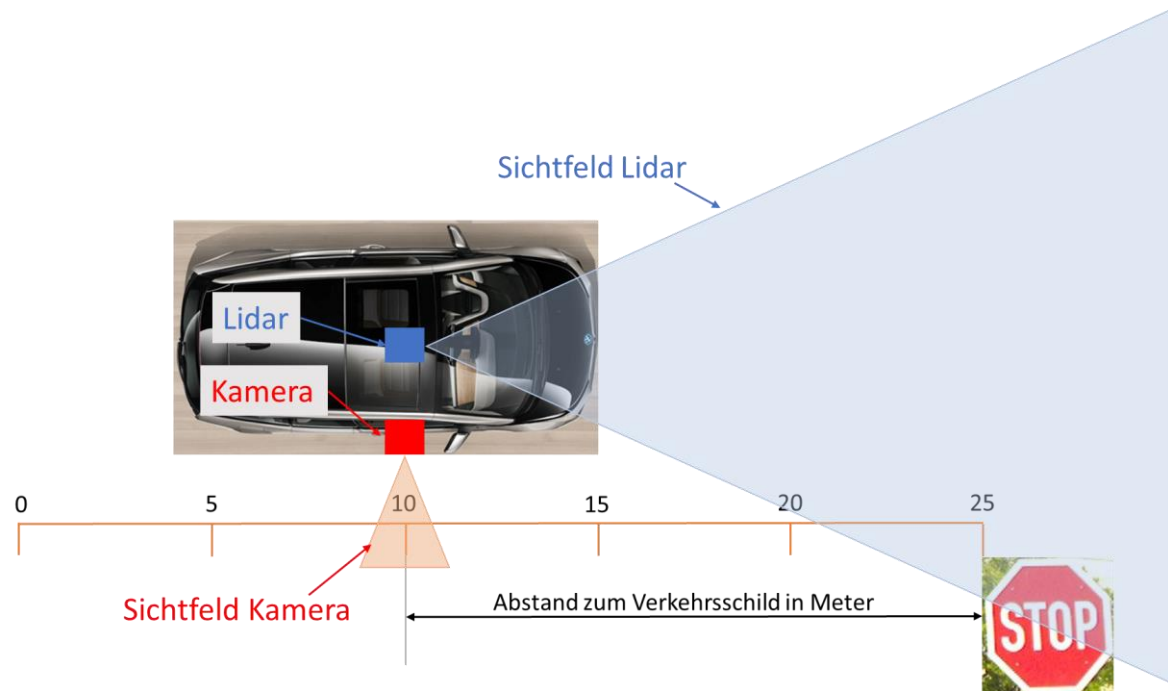


Abbildung 4.8: Aufbau Prüffeld für Differenzbestimmung [23]

Berechnung Abstand Kamera:

$$25 \text{ Meter} - 10 \text{ Meter} = \underline{15 \text{ Meter}}$$

Berechnung Abstand Lidar-Sensor:

$$\text{Ermittlung aus Skript} = \underline{15,073 \text{ Meter}}$$

Differenz Kamera zum Lidar-Sensor:

$$15 \text{ Meter} - 15,073 \text{ Meter} = - \underline{0,073 \text{ Meter}}$$

Wie man dem Ergebnis entnehmen kann, liegt eine Abweichung von - 0,073 Meter vor. Die Differenz ist demzufolge ermittelbar, obwohl ein zusätzlicher Einfluss des Abstandes der Sensoren nicht auszuschließen ist. Diesen Unterschied zueinander zu reduzieren ist nicht notwendig, da dieser Wert nur als grober Anhaltspunkt dient und hierbei schon ein tolerierbares Resultat erzielt. Es lassen sich somit verschiedene Testbilder mit unterschiedlichen Entfernungen zum Verkehrsschild erzeugen und mit den ausgewählten Algorithmen überprüfen. Eine genaue Abschätzung für die sichere Erkennung lässt sich somit treffen. Durch diesen Versuchsaufbau und deren Auswertung konnte eine Zusammenführung der beiden Datenbanken über den abgespeicherten Zeitwert bewiesen werden.

4.6 Ermittlung der Standortpositionen der Stoppschilder

Für die Erstellung einer eigenen Bilddatenbank von Stoppschildern ist deren Lage in Dresden erforderlich. Ein Abfahren aller Straßen ist zwar möglich, aber mit einer enormen Datenmenge und Auswertungszeit verbunden. Die Suche nach einer Alternative ist daher unerlässlich, um die Verkehrsschilder effektiver anzufahren. Diese wurde mit der Kontaktierung der Stadt Dresden gefunden, die dem Autor für den Zeitraum der Erstellung der Diplomarbeit die entsprechenden Standorte bereitstellte. Infolgedessen war eine Vereinbarung für die Datenüberlassung notwendig, um eine Weitergabe an Dritte zu unterbinden.

Die übermittelten Standortdaten zeigen hierbei einen besonderen Datentyp auf. Entsprechend ist eine Umwandlung geboten, um diese auf einem frei wählbaren Routenplaner hochladen zu können. Eine effektive Fahrstrecke mit maximal zehn Stoppschildern kann damit erstellt werden, was an der Begrenzung der Wegpunkte des genutzten Programms zugrunde liegt. Eine Navigation mittels eines internetfähigen und mobilen Endgerätes ist insofern ausführbar. Das effektive Anfahren der Verkehrsschilder ist demzufolge umgesetzt.

4.7 Aufnahmefahrten im Stadtgebiet Dresden

Mit der Erstellung der verschiedenen Routen ist ein Aufnehmen der entsprechenden Objekte möglich. Die vorab festgelegten Routenpunkte können dabei die Fahrzeit minimieren und die Aufnahmegröße der dazugehörigen Videodateien reduzieren. Durch das gezielte Anfahren der Verkehrsschilder kann die Aufnahme im Fahrzeug rechtzeitig gestartet und wieder gestoppt werden. Ein Auffinden und erneutes Anfahren sind somit ausgeschlossen.

Der Abstand der aufgenommenen Bilddaten zum Verkehrszeichen ist ebenfalls aufzunehmen. Der entsprechende Lidar-Sensor auf dem Dach des Versuchsfahrzeuges ist hierfür anzusteuern und liefert die entsprechenden Messdaten, um daraus in der Auswertung die Entfernung zu bestimmen. Der dafür notwendige zusätzliche Laptop ist auf der Rücksitzbank positioniert und gesichert, sodass er bei einem eventuellen auftretenden Hindernis sicher im Fahrzeug untergebracht ist. Die Schnittstelle der Datenübertragung erfolgt über den schon beschriebenen PoE-Switch im Kofferraum. An diesen sind die beiden Laptops und die genutzten Sensoren angeschlossen. Die Ausrichtung der Frontkamera unter

dem Rückspiegel ist somit durchführbar. Die Justierung erfolgt hierfür auf dem Testgelände, was in der Abbildung 4.9 ersichtlich wird.

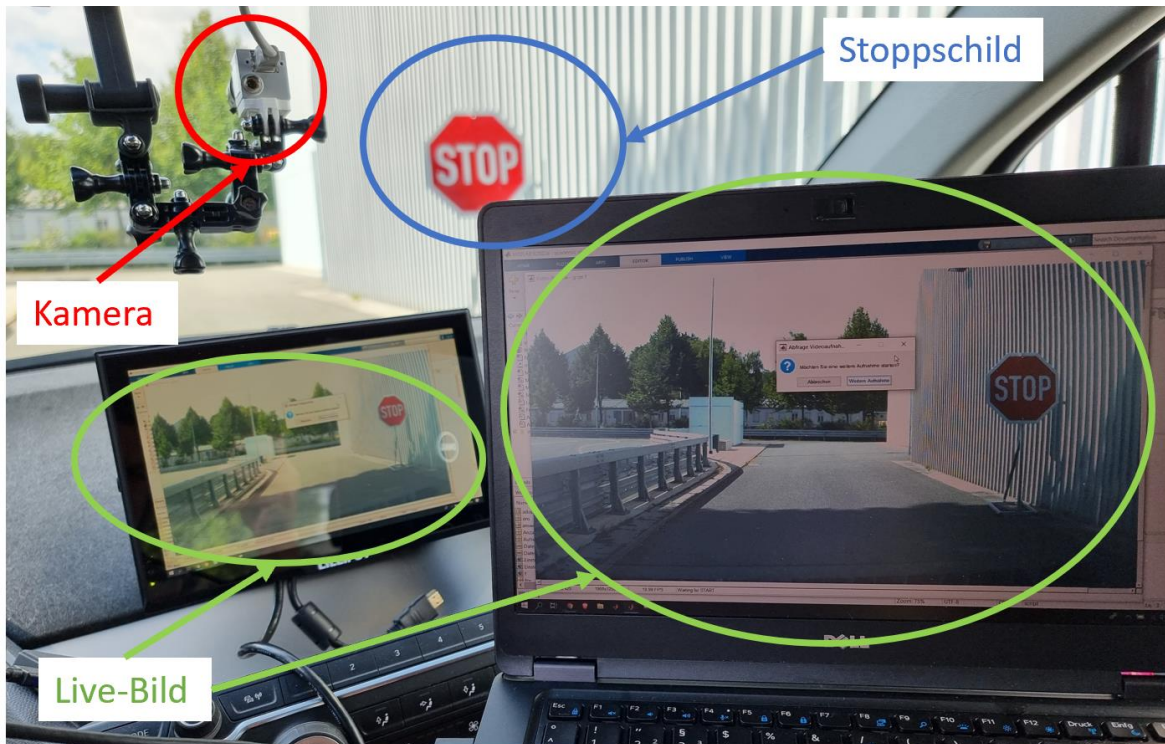


Abbildung 4.9: Übertragung der aktuellen Umgebung auf Laptop

Hierbei ist zu beachten, dass die Kamera möglichst keine Abweichung zum Lidar-Sensor aufweist. In Y-Richtung kann dies mit minimaler Abweichung eingehalten werden, aber in X- und Z-Richtung ist die Aufnahme der Nachkorrekturwerte notwendig. Diesen Abstand soll die Abbildung 4.10 verdeutlichen.

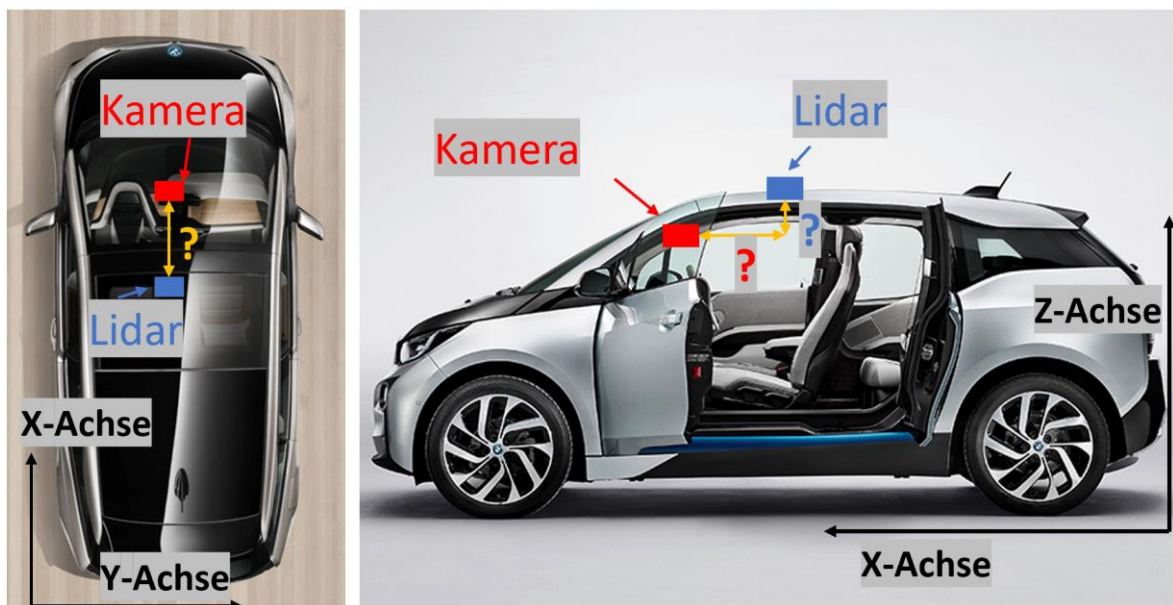


Abbildung 4.10: Positionsdivergenz des Lidar-Sensors [23]

Der tatsächliche Abstand von Kamera zum Verkehrszeichen kann somit in der späteren Auswertung berechnet werden.

Zu beachten sind die während der Fahrt vorzufindenden Vibrationen und Nachstellungsarbeiten, die eine weitere minimale Abweichung zur Folge haben. Eine Nachjustierung durch das vorzufindende Livebild ist daher ausführbar, aber beeinflusst das Ergebnis. Die dabei ermittelte Genauigkeit ist für diesen Anwendungsfall aber tolerierbar, weil es sich um eine erstmalige approximative Abschätzung handelt. Eine Reinigung der Frontscheibe ist ebenfalls erforderlich, um eine bestmögliche Bildaufnahme für den Trainingsdatensatz der Erkennungsalgorithmen zu garantieren. Die Inbetriebnahme des Lidar-Sensors kann schlussendlich ebenfalls erfolgen, um mit der Messfahrt beginnen zu können. Die schon bereits erstellten Routen können nach diesen Handlungen abgefahren werden. Die Erstellung der Videos ist hierbei sehr zeitaufwendig und mit einer großen Datenmenge verbunden, wodurch ein zusätzliches Speichermedium unumgänglich ist.

5 Bereitstellung geeigneter Bilddatenbanken

Die Aufnahmefahrten und Recherchetätigkeiten sind beendet, weswegen die Erstellung geeigneter Datensätze nötig ist. Als Grundlagen hierfür dienen die selbst erstellten Videos und die Bildsammlung des Instituts für Neuroinformatik [24].

5.1 Bilddatenbank aus Aufnahmefahrten

Um einen Überblick über die absolvierten Fahrten zu erhalten, ist eine separate Excel-Tabelle zu erstellen und mit der Bezeichnung `Überblick_Aufnahmen.xlsx` abzuspeichern. Diese beinhaltet zum einen das Datum der Aufnahmen und deren entsprechenden Aufnahmeordner. Des Weiteren sind die einzelnen abgespeicherten Videobezeichnungen zu finden, um diese dann einlesen und auswerten zu können. Als ergänzende Information ist der Standort herauslesbar, welche für eine genaue Zuordnung dient und ein unnötiges erneutes Anfahren ausschließen lässt. Da sich das Verkehrsschild auf unterschiedlichen Positionen vor dem Fahrer befinden kann, ist zusätzlich die Lage des entsprechenden Objektes festgehalten. Üblicherweise sind diese zwar auf der rechten Seite zu finden, aber auf Abbiegestreifen und großen Kreuzungen sind diese auch links zu entdecken. In den letzten beiden Spalten sind Besonderheiten und Bemerkungen dokumentiert, die während der einzelnen Route aufgetreten sind. So kann ein Stoppschild beispielsweise beschädigt oder verdeckt sein, was somit leichter ersichtlich wird und eine Filterung zulässt.

Im nächsten Schritt ist das Skript `Auswertung_1_TabelleFuerVideoErstellen.mlx` zu verwenden. Mit diesem ist die Erstellung einer 11 spaltigen Tabelle in Matlab durchführbar, um die entsprechenden Daten für die Auswertung aufnehmen zu können. Für die Erkennungsalgorithmen wird es notwendig, dass die entsprechenden Bilddateinamen vorhanden sind. Infolgedessen ist in der ersten Spalte die Variable `imageFilename` zu finden, wo diese Bezeichnungen ihren Platz für die Abspeicherung erhalten. Des Weiteren soll die Position auf dem Foto und die Art des Verkehrszeichens zu erkennen sein, weswegen diese ihre Eintragungsmöglichkeiten erhielten. Die Art wird hierbei in Nummern beschrieben, wobei sich an der Bilddatenbank des Instituts für Neuroinformatik zu orientieren ist. Dementsprechend bekommen diese die gleichen Kennzeichnungen, was eine bessere Übersicht mit Zuordnung garantiert. Die Zahl 14 steht demzufolge

stellvertretend für ein Stoppschild, was bei dem Anlernen mehrerer oder unterschiedlicher Verkehrszeichen eine Rolle spielt. Ebenfalls bietet diese Tabelle die Eintragung des jeweiligen Abstandes der genutzten Sensoren zum aufgenommenen Objekt. In den darauffolgenden Spalten sind die entsprechenden Routen, Videos und Frames festgehalten, was für die Zuordnung und dem späteren Nachvollziehen eines nicht involvierten Begutachters wichtig ist. Ein Frame bezeichnet dabei den Bildabschnitt des Videos, wo das entsprechende Foto zu extrahieren ist. Infolgedessen ist es mit diesen Angaben möglich, verschiedene Abstände zum Objekt abzuspeichern und zu dokumentieren. Gleichermäßen gibt es eine Eintragungsmöglichkeit für die Lage der vorzufindenden Fahrzeuge im Aufnahmefoto, falls diese für spätere Projekte benötigt werden. In der letzten Spalte ist das Datum mit der Zeit festgehalten, um die passenden Entfernungen aus den Lidardaten zu bestimmen. Die erstellte Tabelle wird hierbei als Variable *StoppschilderFrames.mat* abgespeichert, welche als Grundlage für die Erstellung der eigenen Bilddatenbank dient.

Im Skript *Auswertung_2_Videodateien_Aufnahmefahrt.mlx* wird diese aufgerufen und eingeladen, um die entsprechenden Informationen eintragen zu können. Als erstes wird hierfür separat ein Ordner erstellt, um einen Speicherort für die einzelnen extrahierten Fotos zu erlangen. Diese dienen später als Trainingsdatensatz für die ausgewählten Erkennungsalgorithmen. Dementsprechend wird zum Beginn der Auswertung die Route und Bildposition festgelegt, um die benötigte Fotodatei aufzurufen. Des Weiteren ist eine Videofunktion enthalten, um einen geeigneten Zeitpunkt für die Extraktion und Entfernung zum Objekt manuell einstellen zu können. Ebenso lässt sich die Bildnummer festlegen, welche die Zeile und Bezeichnung in der Tabelle vorgibt. Am Ende des Auswertungsskriptes ist die Abstandsbestimmung über einen Laserscanner ausführbar, welche von einer parallellaufenden Diplomarbeit mit eingebunden ist. Folglich kann der Zeitstempel des Bildes genutzt werden, um die genauen Punktwolken des Lidar-Sensors zu erhalten. Es stellte sich in der späteren Begutachtung heraus, dass in manchen Sequenzen kein Verkehrsschild zu erkennen ist. Grund dafür sind unterschiedlich zugewiesene Aufnahmezeiten, welche aufgrund einer sporadisch aussetzenden Synchronisation der Laptops entstand. Diese aufgetretene Differenz ist nachträglich nicht mehr genau bestimmbar, weswegen die Entfernungsermittlung nicht für jede Route möglich ist.

Ein Aufbau einer großen Bilddatenbank mit dazugehöriger Informationstabelle ist somit realisiert.

Das Einlesen der Trainingsfotos ist insofern gewährleistet, aber die genaue Position des Verkehrsschildes ist dennoch nicht übermittelbar. Für diese Aufgabe kommt das selbst erstellte Skript `Labeler1_BildLabeler.mlx` zum Einsatz, welches als erstes die ausgefüllte Variable `StoppschilderFrames.mat` mit der entsprechenden Tabelle einliest und damit ein Einlesen aller Bilddateien ermöglicht. Nachfolgend wird der in Matlab integrierte „Image Labeler“ aufgerufen, welcher speziell für diese Aufgabe entwickelt und in Abbildung 5.1 zu sehen ist.

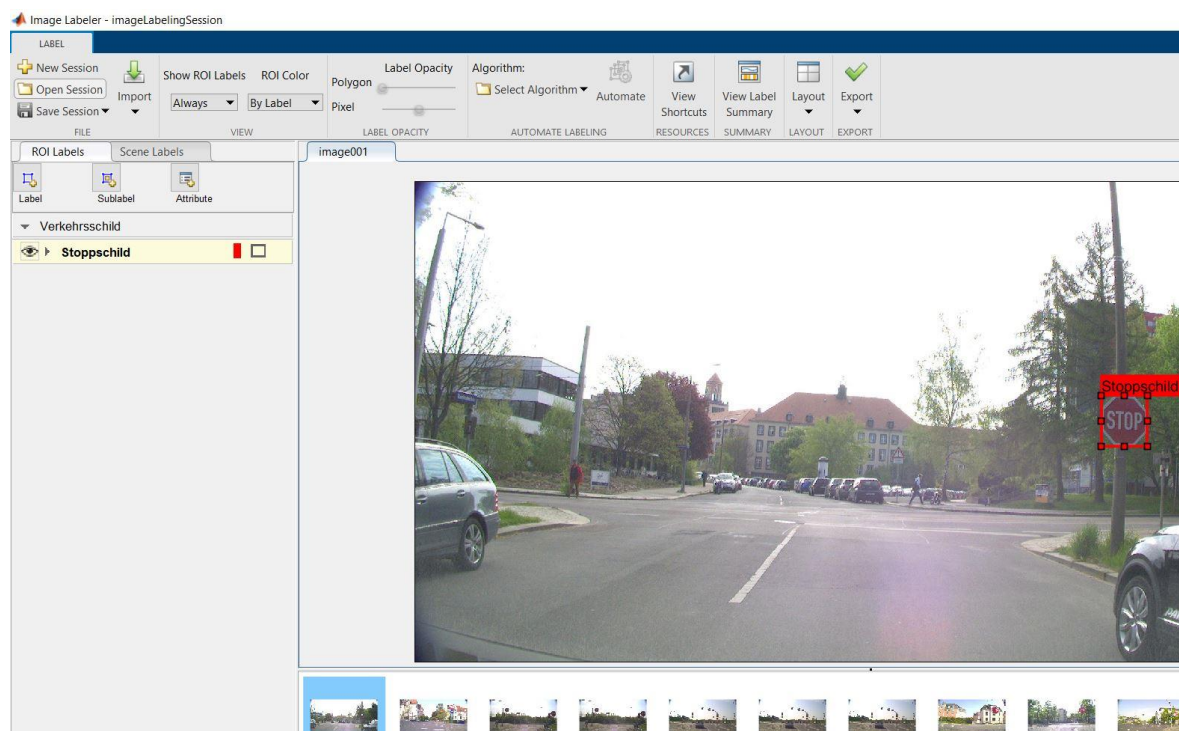


Abbildung 5.1: Festlegung ROI Stoppschild in Matlab

Die Begrenzungsrahmen sind mit diesem Programm manuell an den Verkehrszeichen anbringbar, was üblicherweise als ROI-Festlegung bezeichnet wird. Die Fotografien erhalten somit die geforderten Positionsdaten der anzulernenden Objekte, was wiederum für die entsprechenden Algorithmen gefordert ist. Abspeichern lässt sich dies mit der Variablen `gTruth.mat`, was zum derzeitigen Stand noch nicht automatisiert abläuft. Die fehlenden Informationen sind somit erzeugt, aber noch nicht in der 11 spaltigen Tabelle enthalten. Infolgedessen wird eine dritte Variable mit der Bezeichnung `stopSignsAndCarsAll.mat` erzeugt, die eine Zusammenfassung gestattet und schlussendlich eine genaue Lageposition übermitteln kann.

Es können hiermit verschiedene Variationen von Fotosammlungen aus den Aufnahmefahrten erstellt und erweitert werden, was folglich für die eigene Bilddatenbank zu nutzen ist. Eine davon beinhalten sehr nahe aufgenommene Stoppschilder, wobei die andere wiederum weitere entferntere enthält. Im dritten Ordner sind alle entstandenen nutzbaren Fotografien des Objektes vorzufinden, um die Mindestanzahl an Bildern fürs Training der zu untersuchenden Erkennungsalgorithmen herauszufinden. Ebenso ist es damit möglich, diese in einem davor definierten Intervall zu steigern und die Auswirkungen auf Testbildern erkenntlich zu machen. Infolgedessen sind unterschiedliche Auswertungen erfolgt, um die betrachteten Detektoren einem Vergleich zu unterziehen. Des Weiteren bietet diese Aufteilung eine Erweiterungsmöglichkeit der Fotosammlung für eventuelle zukünftige Projekte.

5.2 Erweiterung durch existierende Bilddatenbank

Die eigene erstellte Bilddatenbank kann für bestimmte Testzwecke genutzt werden, was aber nicht alle Überprüfungsfälle abdeckt und somit nicht ausreichend ist. Zum einen ist es notwendig, die Positionen der Verkehrsschilder im Foto zu verändern. Dies soll einen Aufschluss darauf geben, ob dessen Änderung einen Einfluss auf den Trainingsdaten- und Testdatensatz nimmt. Der Hintergrund und die Größe des Verkehrszeichens sollen hierbei unverändert bleiben, um keinen zusätzlichen Einflussfaktor mit einzubringen. Als Schlussfolgerung bleibt nur eine Spiegelung des gesamten Bildes um die vertikale Achse. Ein Stoppschild eignet sich wie in Abbildung 5.2 zu sehen nicht, weil die Lesbarkeit des Schriftzugs nicht mehr gewährleistet ist und somit falsche Besonderheiten aufweist.



Abbildung 5.2: Stoppschild gespiegelt

Der ausgewählte Algorithmus würde demzufolge die Schrift falsch antrainieren, was unweigerlich zu einer Abweichung und Beeinflussung im Überprüfungsprozess führt. Die Erkennung der Testbilder ist somit ungenügend und gibt die Realität nicht wieder. Eine Aussage über die Positionsveränderung ist somit nicht möglich. Eine weitere Fotosammlung ist folglich nötig, um diesen Testfall zuzulassen und abzudecken.

Das Institut für Neuroinformatik hat in Vergangenheit einen Bilddatensatz erstellt, was für diese Überprüfung genutzt werden kann. Diese besteht aus 42 unterschiedlichen Verkehrsschilderarten [24].

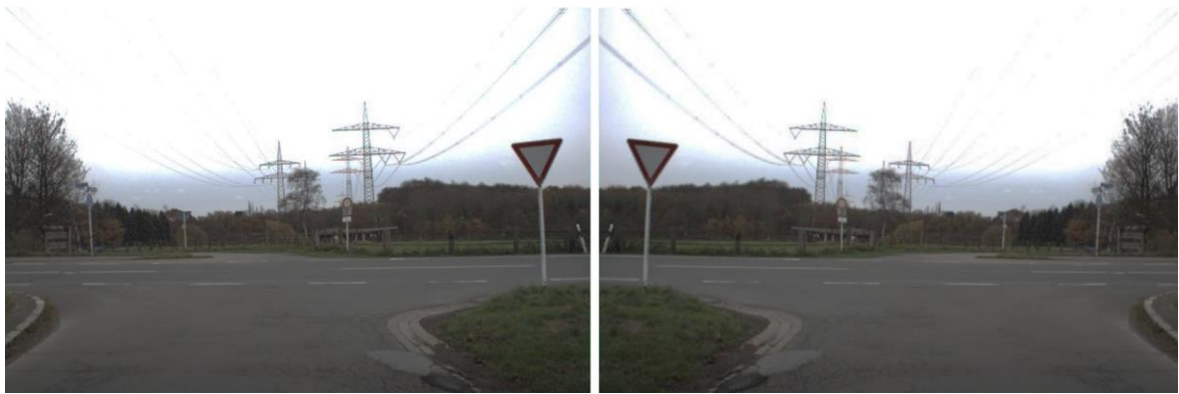


Abbildung 5.3: Vorfahrtsschild gespiegelt [24]

In der Abbildung 5.3 ist ein entsprechendes Vorfahrtsschild gespiegelt. Der Hintergrund und die Form bleiben diesmal bestehen, weswegen es sich für diesen Testfall eignet. Die heruntergeladenen Dateien benötigen hierfür eine Aufbereitung, um sie in Matlab nutzen zu können.

5.2.1 Anpassung Bildformat und Positionsbeschreibung

Zum einen weisen die Bilddaten der Aufnahmefahrt ein anderes Dateiformat auf, was eine weitere Anpassung fordert. Dementsprechend sind alle Fotos des Institutes für Neuroinformatik einzulesen und mit angepasstem Format separat in neuen Ordnern abzuspeichern. Diese können folglich ohne Anpassungsarbeiten in die vorhandenen Skripte eingelesen werden, was ein Training der Algorithmen erlaubt. Dem Datensatz ist eine zusätzliche Textdatei bereitgestellt. Diese gibt die Lage und Position des Verkehrsschildes wieder, wodurch diese ebenfalls in die entsprechende Variable zu überführen ist. Jede Bilddatei enthält somit die zwingenden vier Positionsdaten, welche mit eckigen Klammern in der entsprechenden Tabellenspalte abgespeichert sind. Diese ist für den Trainingsdatensatz notwendig, um sich die Merkmale anzutrainieren. Bei einer

genaueren Begutachtung der heruntergeladenen Daten ist festzustellen, dass die Kennzeichnung der Position nicht der geforderten Eingabe von Matlab entspricht. Als Startwert dient immer der erste linke obere Ausgangspunkt, über dessen der erste Ausrichtungspunkt an der linken oberen Ecke des Objektes gesetzt wird. Die Koordinaten des zweiten Ausrichtungspunktes an der rechten unteren Ecke des Objektes schließt wiederum die genaue Lage des Begrenzungsrahmens ab, wodurch diese in eckigen Klammern ($[x_1, y_1, x_2, y_2]$) festgehalten sind. Die Abbildung 5.4 soll dies verdeutlichen.

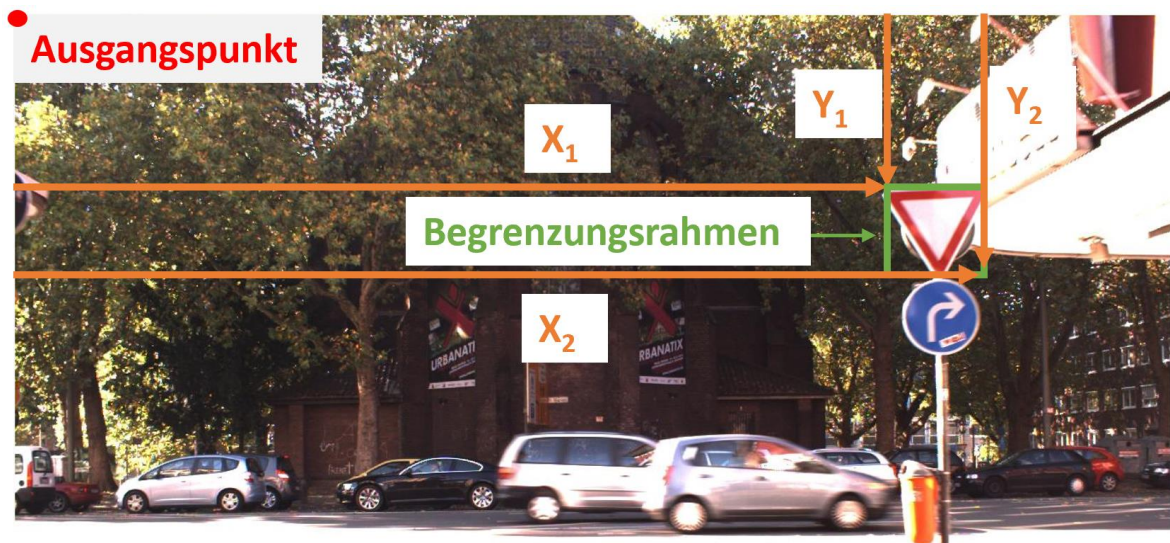


Abbildung 5.4: Positionskennzeichnung bei Institut für Neuroinformatik [24]

Der entsprechende Algorithmus in Matlab benötigt aber die Eingabe des linken oberen Punktes und die Abmessungen des Begrenzungsrahmens ($[x_1, y_1, \text{Breite}, \text{Höhe}]$), was in Abbildung 5.5 ersichtlich wird.



Abbildung 5.5: Vorgabe für Übergabe in Matlab

Eine Überarbeitung der Daten ist folglich unumgänglich, um die entsprechenden Breiten und Höhen zu erhalten. Als Berechnungsgrundlage dient hierfür folgender Zusammenhang:

$$\text{Breite} = X_2 - X_1 \quad (5.1)$$

$$\text{Höhe} = Y_2 - Y_1 \quad (5.2)$$

Die Daten sind infolgedessen so aufbereitet, dass diese für die Betrachtung der gewählten Algorithmen genutzt werden können. In der dritten Spalte der Variable ist die Information der Schildart zu finden, um ein Training mit mehreren verschiedenen Verkehrszeichen zu ermöglichen. Die Zahl 13 beschreibt in diesem Falle ein Vorfahrtsschild.

	1	2	3
	imageFilename	Position	Art
1	'image001.jpg'	[973,335,58,55]	13
2	'image002.jpg'	[112,463,31,29]	13
3	'image003.jpg'	[731,422,41,37]	13
4	'image004.jpg'	[678,443,29,26]	13

Abbildung 5.6: Inhalt der erstellten Tabelle

In Abbildung 5.6 wird der Dateinhalt der Tabelle ersichtlich. Eine Speicherung erfolgt mit dem Dateinamen `stopSignsAndCarsAll.mat`, wodurch diese ohne jegliche Anpassung der Skripte verwendbar ist.

5.2.2 Spiegung eines Bilddatensatzes

Für die Bewertung der verschiedenen Algorithmen ist die Lage der Verkehrszeichen im Trainingsdatensatz zu überprüfen und mit einzubeziehen. Die vorhandenen Bilddateien sind hierfür zu spiegeln und die Position des ROI anzupassen. Eine mathematische Regel ist dafür zwingend erforderlich, um eine gleiche Kennzeichnung der Verkehrszeichen nach der Spiegelung zu garantieren. Die manuelle Festlegung mit dem integrierten „Image Labeler“ erzeugt hierbei eine Abweichung zur Vorgabe, wodurch mehr oder weniger Besonderheiten aus dem Hintergrund hinzukommen können. Diese sind nicht erwünscht und verfälschen das Resultat, weswegen eine Berechnung unumgänglich ist. Die entsprechenden

Höhen, Breiten und Y-Werte bleiben bei den einzelnen Verkehrszeichen gleich, weswegen nur der X-Wert anzupassen ist. Infolgedessen ist der Abstand zum Bildrand zu bestimmen, was in der Abbildung 5.7 verdeutlicht wird.

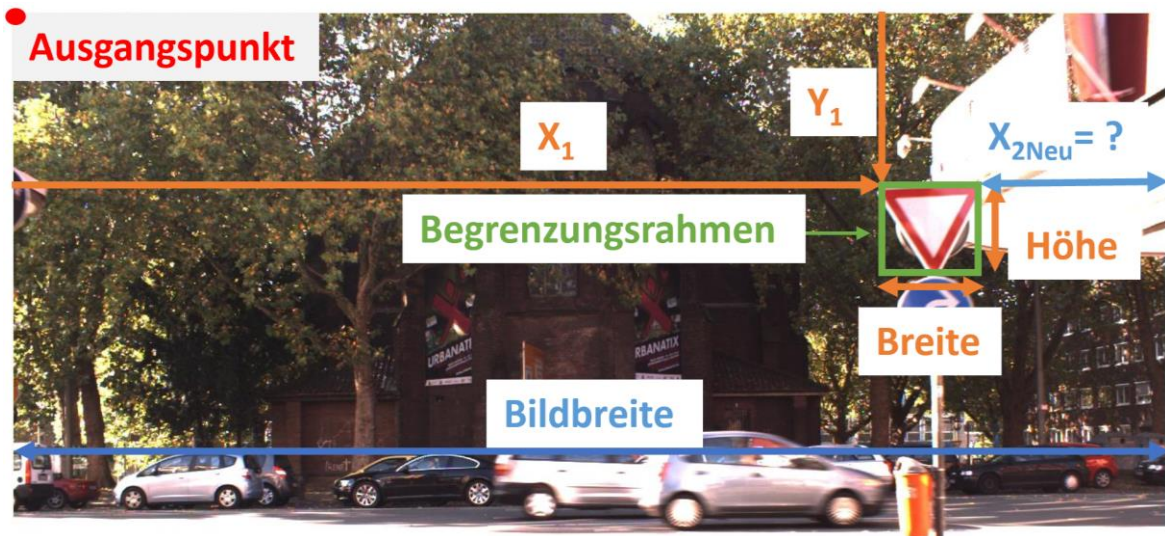


Abbildung 5.7: Berechnungsgrundlage für Spiegelung [24]

Die Bildbreite ist demzufolge zu bestimmen, um den unbekannten Abstand $X_{2\text{Neu}}$ mit nachfolgender Formel zu berechnen.

$$X_{2\text{Neu}} = \text{Bildbreite} - X_1 - \text{Breite} \quad (5.3)$$

Der veränderte Wert ist folglich in der Tabelle zu korrigieren, wodurch er zu dem über die vertikale Achse gespiegelten Bild passt. Die neue Positionsbeschreibung des vorhandenen Objektes soll die Abbildung 5.8 verdeutlichen.



Abbildung 5.8: Positionsangabe für gespiegeltes Bild [24]

5.3 Erstellung von Trainingsdatenbanken

Die Erstellung weiterer Trainingsdatensätze ist durchführbar, um nicht nur die Beispielfotos von Matlab zu nutzen. Eine eigene Sammlung mit Stoppschildern ist folglich aus den Aufnahmefahrten entstanden. Es handelt sich hierbei um zwei Bilddatenbanken, welche das zu detektierende Objekt von nah und fern beinhaltet. Eine Untersuchung der Einflussnahme der Größe des vorzufindenden Schildes ist somit gegeben. Ebenso sind diese beiden vereinigt und in einem getrennten Ordner abgespeichert. Es kann hiermit der Einfluss der Anzahl von Trainingsbildern erarbeitet und bewertet werden. Des Weiteren sind zwei Ordner mit Vorfahrtsschildern zur Beurteilung der Einflüsse aus der Spiegelung erstellt. Der eine enthält somit das Original und der andere die um die vertikale Achse gespiegelte Fotosammlung. Diese sind wie folgt aufgelistet und abgespeichert:

- 00_Matlab_Beispieldatensatz (41 Bilder) [25]
- 01_Aufnahmefahrt_nah (151 Bilder)
- 02_Aufnahmefahrt_weit (137 Bilder)
- 03_Aufnahmefahrt_nah_und_weit (287 Bilder)
- 04_Vorfahrtsschild (52 Bilder) [24]
- 05_Vorfahrtsschild_gespiegelt (52 Bilder) [24].

6 Untersuchung der unterschiedlichen Algorithmen

Die erstellten Bilddatenbanken gewährleisten eine Untersuchung geeigneter Erkennungsalgorithmen. Als Vorgabe ist hierfür das Computerprogramm Matlab zu nutzen, was die entsprechende Auswahl einschränkt. Ein Vorteil sind die im Internet bereitgestellten Beispiele, welche für die entsprechenden Tests und Bewertungen zu nutzen sind. Eine Grundlage ist demzufolge gegeben, dennoch sind verschiedene Ansätze und Varianten erforderlich.

Für einen ersten Überblick und deren Auswahl ist die Hilfetabelle auf der Internetseite des Programmanbieters nützlich. In drei unterschiedlichen Beispielen ist dabei explizit das Stoppschild detektiert, wodurch diese sich für eine nähere Betrachtung eignen. In der Tabelle 6.1 sind die genutzten Detektoren und Beispiele aufgeführt [26].

Tabelle 6.1: Detektoren für Stoppschilderkennung in Matlab

Detektor	Matlab Beispiele
acfObjectDetector	Train ACF-Based Stop Sign Detector [27]
fastRCNNObjectDetector	Train Fast R-CNN Stop Sign Detector [28]
trainCascadeObjectDetector	Train Cascade Stop Sign Detector [29]

Diese nutzen für ihre Prüfung das gleiche verwendete Testbild, wodurch sich ein Vergleich anbietet. Demzufolge ist es bei einer richtigen Auffindung auf dem Bild möglich, den entsprechenden Wert des Ergebnisses in einer Tabelle festzuhalten. In manchen Fällen ist dies aber nicht ersichtlich, wodurch im Resultat eine 0 zu finden ist. Die Gründe dafür sind falsche oder fehlende Markierungen des Stoppschildes.

6.1 Testszenarien mit dem ACF-Detektor

Es handelt sich bei diesem Beispiel um ein Erkennungsalgorithmus, der das schon beschriebene ACF-Verfahren nutzt. In dem dafür bereitgestellten Skript werden zunächst alle 41 Bilddateien des mitgelieferten Trainingsdatensatzes und dessen Merkmalsbereiche der vorhandenen Verkehrszeichen eingelesen. Der daraus

angelernte Detektor wird dabei mit dem gleichbleibenden Testbild untersucht. Als Ergebnis erhält man eine Erkennungsbox, welche zusätzlich das ermittelte Vertrauen (Confidence) auf dem Foto ausgibt. Das Resultat kann einen Wert von minus unendlich bis plus unendlich annehmen, was keine große Aussagekraft besitzt und keinen idealen Mindestwert für eine sichere Beurteilung zulässt [27].



Abbildung 6.1: Ergebnis ACF-Detektor

Wie in Abbildung 6.1 zu erkennen, wird beim eingesetzten Trainingsdatensatz ein Resultat von circa 60 erzielt. Das gesuchte Objekt ist folglich richtig erkannt und gekennzeichnet. Eine Interpretation ist zum derzeitigen Stand nicht möglich, weil seitens des Entwicklers kein erreichter Mindestwert angegeben wird. Dies ist bei weiteren Testfotografien ebenfalls ersichtlich, wo positive und negative Zahlenergebnisse vorzufinden sind. In einigen Fällen ist das Stoppschild erst gar nicht zu erkennen, was die Festlegung eines eigenen Zielwertes nicht gestattet. Ein Ausschnitt der Untersuchung ist in Abbildung 6.2 zu finden.



Abbildung 6.2: Ergebnisse ACF-Detektor mit Testbildern aus Aufnahmefahrt

Dies ist als Erkenntnis nicht zufriedenstellend, weswegen der Einfluss der Markierung des ROI am Verkehrszeichen zu untersuchen ist. Der „Image Labeler“ von Matlab kommt demzufolge zur Verwendung. Die zu erkennenden Objekte im Trainingsdatensatz sind dafür zum einen grob und fein gekennzeichnet, um deren Einflussnahme beurteilen zu können. Zur Verdeutlichung ist dies in der nachfolgenden Abbildung 6.3 zu sehen.

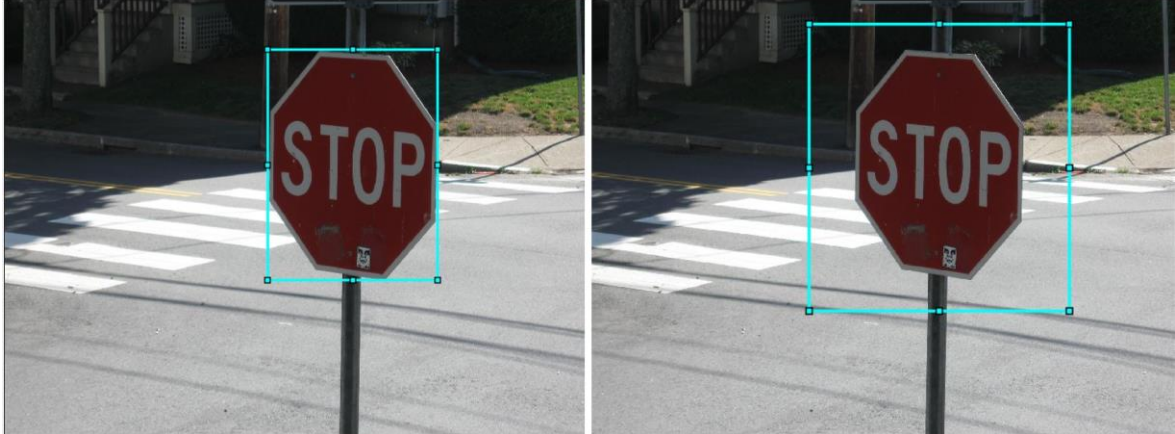


Abbildung 6.3: Möglichkeiten für Kennzeichnung des ROI [25]

Die beiden Bildordner können insofern getrennt zum Training des Erkennungsalgorithmus genutzt werden. In der Auswertung erhält man unterschiedliche Ergebnisse, was in der Abbildung 6.4 zu sehen ist.



Abbildung 6.4: links genau und rechts grob gekennzeichnet

Eine genaue Kennzeichnung bringt zum einen mehr Detektionen zum Vorschein, wofür es keine Erklärung gibt. Folglich sind im ersten Fall zu viele mögliche Verkehrszeichen erkannt, was als Auswertung zu mehreren Objekten führt. Es ist aber auch ersichtlich, dass das Stoppschild im linken Fall mit einem höheren Wert gekennzeichnet ist. Aus dieser Beobachtung entstand die Idee der Filterung des Datensatzes nach dem maximalen Wert, was zu dem Ergebnis in Abbildung 6.5 führt.



Abbildung 6.5: Ergebnisse nach Filterung

Ein geeignetes und nutzbares Resultat ist mit dieser Maßnahme erreicht. Aus diesem Grund ist eine dauerhafte Filterung nach dem Höchstwert notwendig, um vorzufindende falsche Detektionen nicht angezeigt zu bekommen. Aufgrund der sehr großen Grenzen des Vertrauenswertes kann hierfür aber kein zusätzlicher exakter Schwellenwert angegeben werden. Es ist demzufolge je nach Anwendung und Einsatzgebiet eine entsprechende Iteration notwendig, um eine Fehleinschätzung in der Bewertung auszuschließen. Dies ist in der Bearbeitungszeit dieser Diplomarbeit nicht umsetzbar, weswegen lediglich der maximale Wert als Filterung dient.

In der nachfolgenden Untersuchung ist der Einfluss der Reihenfolge des veränderten Trainingsdatensatzes zu erkennen. Die komplette Bildsammlung ist deshalb gespiegelt, sodass mit dem letzten Foto begonnen wird und ein erneutes Anlernen erfolgt.



Abbildung 6.6: Reihenfolge Trainingsdatensatz links ungefiltert und rechts gefiltert

Das Resultat mit vielen fehlerhaften Detektionen ist in Abbildung 6.6 zu sehen. Das Testbild wird dabei dauerhaft mit einem hohen Vertrauenswert gekennzeichnet. Die Filterung bringt keine Verbesserung, weswegen eine Änderung der Reihenfolge nicht vorgenommen werden kann. In weiteren Untersuchungen mit Anordnungsänderung ist ein ähnlicher Effekt ersichtlich. Demzufolge sollte der

Ordner fürs Anlernen unverändert bleiben, wonach dieser folglich auf das Beispiel angepasst ist.

Die Position des Verkehrszeichens ist ebenfalls zu überprüfen. Der Trainingsdatensatz mit den Vorfahrtsschildern kommt deshalb zum Einsatz. Die Anzahl der Anlernbilder ist zu Beginn auf 41 beschränkt, um einen gewissen Vergleich zur Erkennung eines Stoppschildes zu erhalten. Das entsprechende Testbild ist hierbei aus dem Internet entnommen. Das Ergebnis zeigt viele falsche Kennzeichnungen an, weswegen eine Erhöhung auf 52 Fotos nötig ist. Dies führt nach der Filterung zu einem nutzbaren Ergebnis, was in Abbildung 6.7 nachzuvollziehen ist [24; 30].

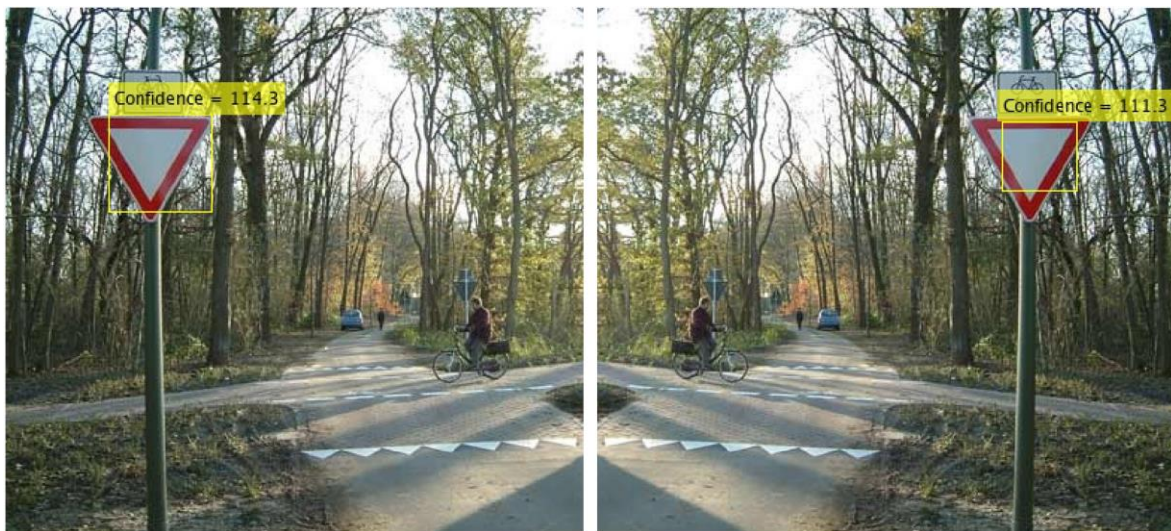


Abbildung 6.7: Position Verkehrschild [30]

Das hierbei genutzte Kontrollbild ist des Weiteren um die vertikale Achse gespiegelt mit einem leicht daraus resultierend höheren Wert. Im Anschluss daran konnte der Trainingsdatensatz gespiegelt werden, was zu niedrigeren Erkennungswerten führt. Das rechts befindliche Objekt erreicht dabei lediglich einen Wert von 86,7. Die linke Seite ist mit 101,6 hierbei wieder etwas höher, aber kann ebenfalls nicht den schon erreichten Höchstwert übertreffen.

Im letzten Prüfablauf sind die verschiedenen selbst erstellten Datensätze zu nutzen. Die Testbilder sind die beiden Stopp- und Vorfahrtsschilder. Es ist in Anlehnung des Beispiels mit 41 Bildern zu beginnen und um dessen Wert ständig zu erhöhen, bis die maximale verfügbare Anzahl erreicht ist. Als Filterung dient hierbei der maximale Wert, wobei sich das zu erkennende Objekt nur auf der rechten Seite befand. Folglich entsteht als Resultat der ganzen Untersuchungen die Abbildung 6.8, die den Einfluss der genutzten Menge der Trainingsbilder widerspiegelt.

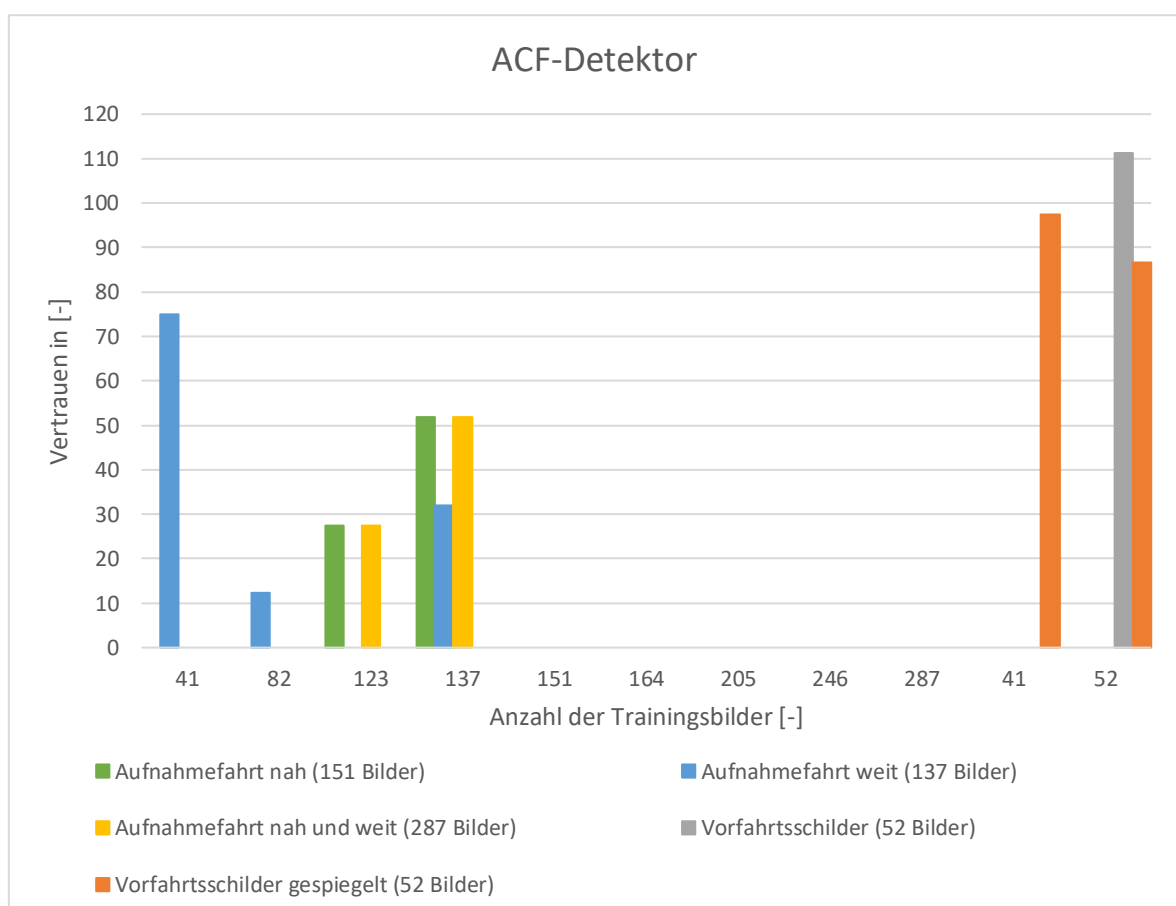


Abbildung 6.8: Ergebnis des ACF-Detektors

Der entsprechende Algorithmus erkennt das Stoppschild mit 41 weit entfernten Trainingsbildern am besten. Dieser Wert fällt stückweise immer weiter ab und erhöht sich erst wieder bei 137 Bildern. Die nahen Aufnahmen zeigen am Anfang ein schlechteres Erkennungsergebnis an, was sich erst ab 123 Fotos verbessert. Bei der Betrachtung von Vorfahrtsschildern ist erst ab 52 Bildern eine Kennzeichnung erkennbar. Des Weiteren ist wieder der Einfluss des gespiegelten Datensatzes ersichtlich, was hierbei sogar bei kleinerer Anzahl an Anlernfotos zu einer richtigen Detektion führt. Die Position und Reihenfolge nimmt demzufolge einen erheblichen Einfluss auf das entsprechende Resultat. Dieses Ergebnis ist keinesfalls zufriedenstellend, weswegen der Support von Matlab kontaktiert wurde. Hierbei stellte sich heraus, dass der entstandene Trainingsdatensatz zu gering ist und daher der Einfluss der Reihenfolge ersichtlich wird. Als Vorgabe gibt es keine benötigte Gesamtanzahl, weswegen man diesen lediglich durch Versuche bestimmen kann. Im Bezug des berechneten Vertrauenswertes gibt es ebenfalls keinen exakten Wert, wodurch ein iterativer Prozess für spätere Anwendungen unumgänglich ist. Der dabei ermittelte Schwellenwert kann für eine bessere Filterung der schon ausgewerteten Ergebnisse genutzt werden.

6.2 Testszenarien mit dem Fast-R-CNN-Detektor

Die Grundlage für den nächsten Algorithmus bildet der Trainingsdatensatz von CIFAR-10. Dieser besteht aus 50 000 Bilddatensätzen und ist auf 10 unterschiedliche Klassen aufgeteilt [31].

In dem Beispiel von Matlab wird dieser für das Training eines R-CNN-Detektors genutzt, um damit die Gewichtungen und Merkmale für das Netzwerk anzutrainieren. Es entsteht daraus ein Fast-R-CNN Detektor, welcher als Grundlage für die weitere Begutachtung dient. Im nächsten Schritt wird diesem Erkennungsalgorithmus ein weiterer kleinerer Trainingsdatensatz übergeben, um eine Feinabstimmung hervorzurufen und auf ein bestimmtes Objekt stärker fokussiert zu sein. Die zweite Anlernfotosammlung enthält demzufolge lediglich das Verkehrsschild, um ein verbessertes und gestärktes Netzwerk für die Untersuchung zu erhalten. Eine geringe Anpassung der Gewichtung ist die Folge. Durch dieses davor trainierte Netzwerk kann eine Reduzierung der Trainingszeit erreicht werden. Würde man diese Gegebenheit nicht nutzen, dann wäre ein noch höherer eigener Datensatz des zu erkennenden Objektes nötig. Die Zeit des Anlernens steigt in diesem Falle, was für gewisse Anwendungsgebiete ungünstig sein kann.

Der gleiche Trainingsdatensatz mit 41 Bildern dient wiederum als erste Untersuchungsgrundlage. Die einstellbaren Optionen bleiben unverändert. Die Bewertungsskala ist mit einem Wertebereich von 0 bis 1 angegeben, wobei ein möglichst hoher Wert anzustreben ist. Wie hoch dieses Resultat für ein ausreichendes Vertrauen ausfallen sollte, überliegt dabei den entsprechenden Anwendungsfall und deren Testauswertung. Das durch Matlab vorgegebene Beispiel erreicht einen Wert von 0,64415 und ist in Abbildung 6.9 ersichtlich [28].



Abbildung 6.9: Ergebnis Fast R-CNN-Detektor [28]

Es ist dabei zu erkennen, dass der entsprechende erstellte Auswertungsrahmen das komplette Verkehrszeichen einschließt und keine Fehldetektionen erkennbar sind. Diese beiden zusätzlichen Gesichtspunkte unterstützen einen guten Detektionsbereich, weswegen dieser ebenfalls weiteren Tests zu unterziehen ist. Als Filterung dient erneut der maximale Erkennungswert, um die Ergebnisse vergleichen zu können.

Der Einfluss des ROI ist mit den unterschiedlichen Kennzeichnungen zu überprüfen. Als Resultat sind lediglich minimale Abweichungen zu erkennen, wodurch kein Einfluss festzuhalten ist.

Daraufhin ist die Einflussnahme der Reihenfolge des entsprechenden Trainingsdatensatzes zu untersuchen, was beim ACF-Erkennungsalgorithmus zu erheblichen Abweichungen der Resultate und falschen Detektionen führt. Der neusortierte Beispieltrainingsordner kann insofern wieder genutzt werden, der sich durch den Beginn des letzten Fotos auszeichnet. Die erneute Überprüfung durch das schon bekannte Testbild ruft dabei wiederkehrende Anzeigeergebnisse hervor. Des Weiteren bietet das von Matlab bereitgestellte Beispielskript eine Möglichkeit der zufälligen Anordnung von Trainingsbildern. Dieser Test erzielt ein ähnliches Resultat, weswegen eine Auswirkung der Reihenfolge auszuschließen ist.

Die Einflussnahme der Positionsänderung im Trainingsdatensatz ist zu überprüfen, weswegen die 41 Vorfahrtsschilder wieder zu nutzen sind. Das entsprechende Testbild bleibt ebenfalls gleich, um einen Vergleich mit den anderen Algorithmen zu ermöglichen. Im Resultat enthalten die Markierungsboxen das zu findende Verkehrszeichen und deren Erkennungswert. Bei der Spiegelung um die Vertikalachse sinkt dieser Wert gering, was ebenfalls auf eine gewisse Ausrichtung des Algorithmus hindeutet. Dies soll die Abbildung 6.10 nochmals verdeutlichen.



Abbildung 6.10: Ergebnis bei 41 Trainingsbildern und gespiegeltem Testbild [30]

Die senkrecht um die Mittelachse gespiegelte Trainingsdatenbank ist demzufolge zu nutzen, um den Einfluss bei 41 Fotos zu verdeutlichen. In der Abbildung 6.11 ist dies ersichtlich.



Abbildung 6.11: gespiegelter Testdatensatz und dessen Ergebnis [30]

Im Ergebnis ist eine schwächere Schilderkennung auffällig. Die entsprechenden Werte sinken in beiden Fällen, obwohl das links stehende Objekt eine größere Verschlechterung zu verzeichnen hat. Diese Feststellung ist weiter zu untersuchen, wodurch die Anzahl der Bilder auf 52 Bilder zu erhöhen ist.



Abbildung 6.12: Erhöhung des gespiegelten Trainingsdatensatzes [30]

In der Abbildung 6.12 wird das weitere Absinken der Werte ersichtlich, wodurch eine höhere Anzahl an Bildern keine Verbesserung bei diesem Testbild hervorruft.

Die Auswirkung der Anzahl der Testbilder ist zu untersuchen, um eine gewisse Tendenz zu erkennen. Für diese Art der Bewertung wird die eigene erstellte Bilddatenbank genutzt, weil diese eine höhere Anzahl von Stoppschildern zur Verfügung stellt. Im Ergebnis kann man feststellen, dass bei steigender Anzahl von Trainingsbildern der entsprechende Erkennungswert steigt und sich auf einen Wert einpegelt. Dies ist in Abbildung 6.13 ebenfalls nachvollziehbar.

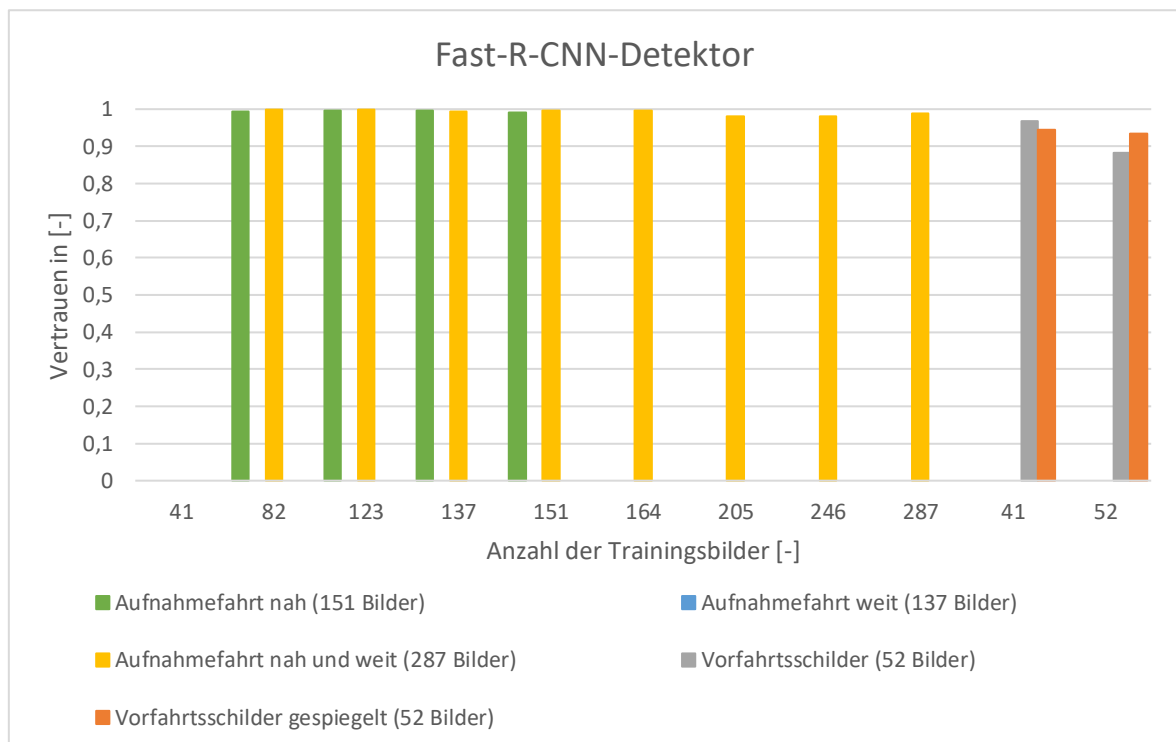


Abbildung 6.13: Ergebnis des Fast-R-CNN-Detektors

Des Weiteren ist zu erkennen, dass für die Auffindung eines Objektes im Testbild eher nahe Trainingsdaten benötigt werden. Im Anlernprozess mit weiten Aufnahmen ist das gesuchte Schild in keinem Durchlauf erkennbar, weswegen dies nicht zur Nutzung dient. Ebenso ist der schon angesprochene leichte Einfluss des gespiegelten Trainingsdatensatzes ersichtlich, was aber nicht zu erheblichen Abweichungen wie beim ACF-Detektor führt.

6.3 Testszenarien mit dem Cascade-Detektor

Der Detektor benötigt einen zusätzlichen Ordner für das entsprechende Training des Algorithmus. In diesem sind wiederum sogenannte Negativproben enthalten. Es handelt sich dabei um eine Anzahl von Fotos mit typischen Fahrsituationen und Verkehrsschildern, auf den der Erkennungsalgorithmus nicht zu trainieren ist. Es sind folglich gleichartig wirkende Objekte mit einzubeziehen, wo eine Verwechslungsgefahr besteht. Dies kann beispielhafterweise eine ähnliche Form und Farbauswahl eines anderen Schildes sein, was zu einer falschen Detektion führen kann. Es ist infolgedessen möglich ähnlich wirkende Objekte einzubeziehen, um Fehleinschätzungen entgegenzuwirken. Ein geeigneter Bilddatenordner ist folglich nötig, welcher aus dem Institut für Neuroinformatik zu erstellen ist. Dieser ermöglicht demzufolge erst eine weitere Untersuchung und kam bei jedem Testlauf zum Einsatz [24].

Das Beispiel in Matlab gibt zu erkennen, dass das Verkehrszeichen lediglich mit einem gelben Begrenzungsrahmen und dem englischen Wort „sign“ (Schild) gekennzeichnet ist. Ein entsprechender Erkennungswert ist daher nicht gegeben und nutzbar. Der Trainingsdatensatz und das Testbild sind dabei identisch wie bei den anderen schon untersuchten Algorithmen, was in Abbildung 6.14 deutlich wird.



Abbildung 6.14: Erkennung Cascade-Detektor

Der Einfluss des ROI ist folglich zu testen, was zu unterschiedlichen Ergebnissen führt. Es ist dabei festzustellen, dass ein grob gekennzeichnete Trainingsdatensatz zu einer richtigen und falschen Detektion führt. Eine dauerhafte sehr feine Markierung hingegen bringt keine Erkennung hervor, wodurch diese näher zu untersuchen ist. Die Abbildung 6.15 soll dieses Resultat verdeutlichen.



Abbildung 6.15: links fein und rechts grob gekennzeichnete Trainingsdatensatz [29]

Ein Ansatz ist demzufolge gefordert, um einen Vergleich der Algorithmen zu ermöglichen. Dies gestaltet sich dabei komplizierter, weil kein Zahlenwert für die Markierung angegeben wird. Die Lage der Ergebnisbox ist hierbei in einer Tabelle

abgespeichert, wobei die mit kleinstem Umfang am Anfang festgehalten ist. Dennoch bietet dieser Detektor weitere Einstellungsmöglichkeiten an, um den Erkennungsschwellenwert zu ändern und falsche Objekte zu unterdrücken. Eine Erhöhung des Wertes erzeugt bei grob gekennzeichneten Anlernbildern die gewünschte richtige Detektion, was aber mit einer ständigen Anpassung verbunden ist und keine endgültige Lösung sein kann. Gleiches ist bei Senkung des Schwellenwertes ersichtlich, wodurch beim nah markierten Trainingsdatensatz ebenfalls eine richtige Auffindungsbox am Stoppschild zu erkennen ist. Eine dauerhafte Änderung dieses Einstellwertes ist zum derzeitigen Stand nicht realisierbar, wodurch ein anderer Ansatz nötig wird. Als Filterung dient daher der erste Rahmen in der Ergebnistabelle, welcher den kleinsten Umfang aufweist. Dies ist damit zu begründen, dass lediglich ein einziges Verkehrszeichen erkannt und angezeigt werden soll. Eine Untersuchung ist wie bei den anderen Algorithmen ausführbar, wodurch ein Vergleich realisiert ist. Die Entfernung des zu detektierenden Objektes spielt deshalb im Testbild eine sehr entscheidende Rolle. Ein sehr weit entferntes Stoppschild erzeugt hierbei eine eher kleinere Markierungsbox, was vorteilhaft für die Filterung ist. Das eher sehr nahe und flächenmäßig größere Erkennungsobjekt erzeugt eine größere Kennzeichnung, wodurch es eher herausgefiltert wird. Dieses wird demzufolge eher als Fehldetektion interpretiert, was in Abbildung 6.16 ersichtlich wird.

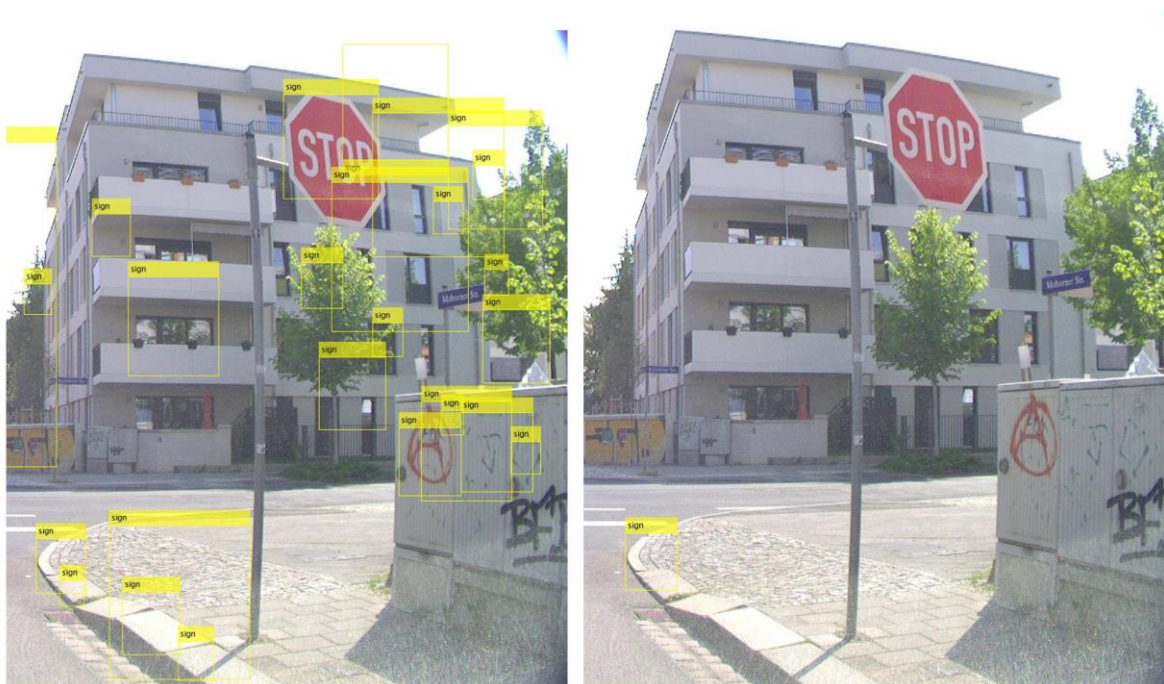


Abbildung 6.16: Filterungsauswirkung beim Cascade-Detektor

Dieser Fehlentscheidung kann mit der Betrachtung des größeren Umfangs oder einer Erhöhung des Erkennungsschwellenwertes entgegengewirkt werden, was folglich die Anzahl der Detektionen reduziert und den vorteilhafteren Kennzeichnungsrahmen ausgibt. Dies konnte nicht für jedes neue Testbild geschehen, weswegen es auf den kleinsten Rahmen festgelegt ist.

Der Einfluss der Reihenfolge ist dementsprechend testbar, wobei wieder mit dem letzten Bild des Trainingsdatensatzes begonnen wird. Es ist keine Veränderung des Resultates ersichtlich, wodurch hiermit kein Einfluss erfolgen kann.

Die Lage und Position der Verkehrsschilder sind demzufolge wieder zu untersuchen. Als Grundlage dient die Bilddatenbank mit den entsprechenden Vorfahrtsschildern. Es stellt sich hierbei heraus, dass das gesuchte Objekt bei der vorgegebenen Grundeinstellung mit Filterung und einer Erkennungsschwelle von 500 auf dem Testbild erkannt wird. Der festgelegte Wert ist dabei iterativ zu ermitteln, was ab der Anzahl von 41 Fotos zu einem nutzbaren Ergebnis führt.

Ein Spiegeln des Trainingsdatensatzes um die vertikale Achse ruft ebenfalls nur minimale Veränderungen und Unterschiede hervor, was beispielhaft zu folgenden Ergebnissen in der Abbildung 6.17 führt [24].

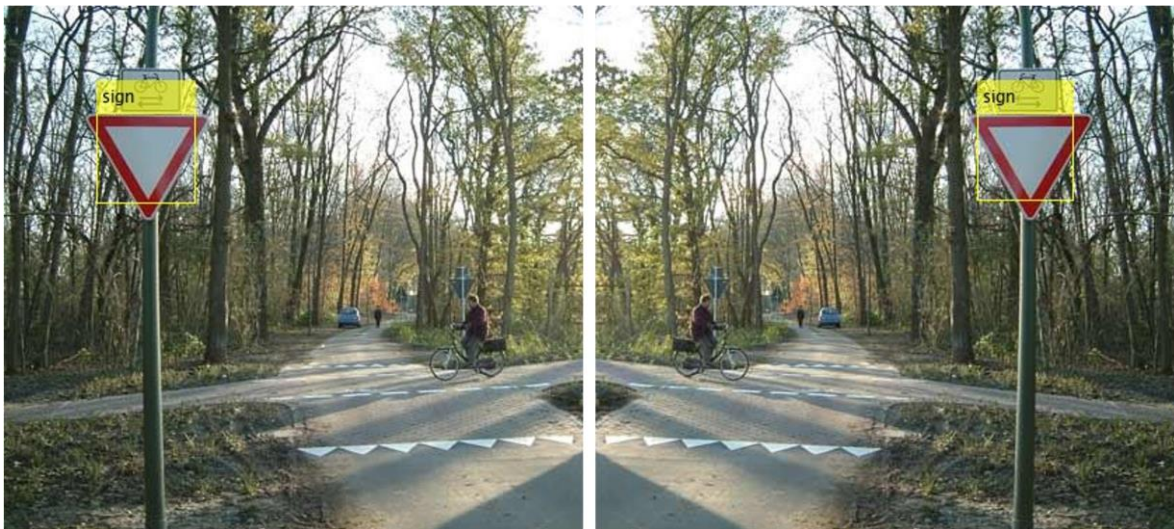


Abbildung 6.17: Einflussnahme Trainingsdaten mit erhöhten Schwellenwert [30]

Die Anzahl der Fotos aus den gewonnenen Aufnahmefahrten sind folglich zu untersuchen. Der voreingestellte Schwellenwert von 4 ist unverändert geblieben, wodurch die erste Erkennungsbox durch die beschriebene Festlegung zur Anzeige gebracht wird. Das Resultat mit dem gleich gebliebenen Testbild ist in der Abbildung 6.18 zu finden.

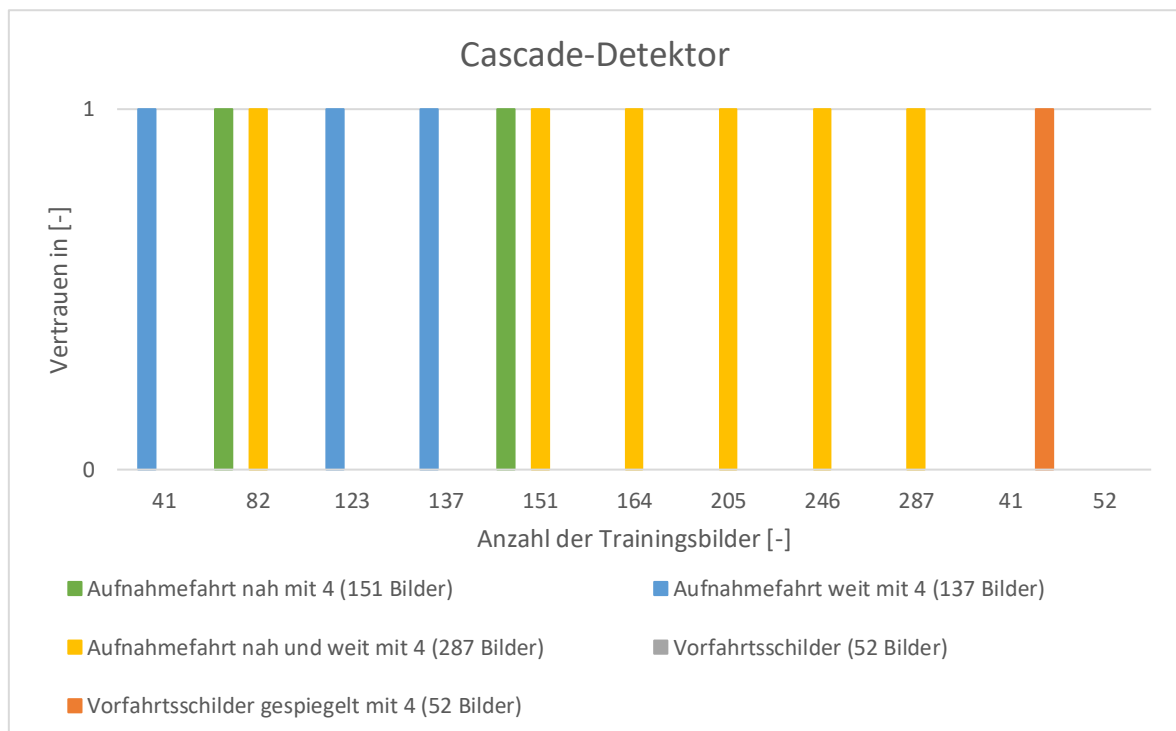


Abbildung 6.18: Resultat von Cascade-Detektor

Aufgrund des fehlenden Ergebniswertes ist eine erfolgreiche Detektion mit 1 festgelegt und so für den Testfall gekennzeichnet. Die entsprechenden falschen Markierungen sind infolgedessen mit 0 bewertet. Es ist dabei festzustellen, dass die Anzahl der Trainingsbilder zu einem Wechsel der Ergebnisse führen. Dies ist damit zu begründen, dass der Erkennungsschwellenwert immer wieder anzupassen ist. Ein optimiertes Ergebnis wäre die Folge, was mit dem derzeitigen Stand nicht automatisiert umzusetzen ist. Diese Erkenntnis wird vor allem bei der Detektion des Vorfahrtsschildes ersichtlich, was bei einem Erkennungsschwellenwert von lediglich 4 nur noch einmal erkannt wird. Das Stoppschild wird bei der maximalen Anzahl aller vorhandenen Datensätze erkannt, was vor allem an der vorzufindenden Entfernung im Testbild liegt. Die Filterung ist für diesen Fall daher sehr vorteilhaft, was bei einem kleineren Abstand schon wieder zu einem anderen Ergebnis führen kann.

6.4 Zwischenresultat der Untersuchung

Die verschiedenen Detektoren sind auf ihren Anlernprozess überprüft worden. Es stellen sich hierbei Besonderheiten heraus, die als Überblick in der nachfolgenden Tabelle zu finden sind.

Tabelle 6.2: Überblick des Zwischenresultates

	ACF-Detektor	Fast-R-CNN-Detektor	Cascade-Detektor
Welcher Wertebereich ist zu erreichen?	- zwischen minus unendlich und plus unendlich	- zwischen 0 und 1	- kein Wertebereich vorhanden
Was ist der zu erreichende Bestwert?	- möglichst hoher Wert, ohne genaue Vorgabe	- die 1 gibt eine sichere Erkennung wieder	- kein Bestwert vorhanden
Wie ist das Verhalten bei Änderung des ROI im Anlernprozess?	- Einfluss auf Ergebnis sichtbar	- kein Einfluss auf Ergebnis	- Einfluss vorhanden und ist mit Erkennungsschwellenwert anpassbar
Wie ist das Verhalten bei gespiegeltem Testbild?	- Unterschiede im Resultat ersichtlich	- minimale Unterschiede in den Resultaten ersichtlich	- minimale Unterschiede in den Resultaten ersichtlich
Wie verhält sich der Algorithmus bei gespiegelten Anlern-datensatz?	- Einflussnahme ist erkennbar	- Einflussnahme ist erkennbar	- Einflussnahme ist ohne Anpassung Erkennungsschwellenwert erkennbar
Welchen Einfluss nehmen die nahen Anlernbilder?	- bei 2 von 5 Prüfabläufen ist das Stoppschild erkannt wurden	- bei 4 von 5 Prüfabläufen ist das Stoppschild erkannt wurden	- bei 2 von 5 Prüfabläufen ist das Stoppschild erkannt wurden
Welchen Einfluss nehmen die weiten Anlernbilder?	- bei 3 von 4 Prüfabläufen ist das Stoppschild erkannt wurden	- bei 0 von 4 Prüfabläufen ist das Stoppschild erkannt wurden	- bei 3 von 4 Prüfabläufen ist das Stoppschild erkannt wurden

Welchen Einfluss nehmen die nahen und weiten Anlernbilder?	- bei 1 von 9 Prüfabläufen ist das Stoppschild erkannt wurden	- bei 8 von 9 Prüfabläufen ist das Stoppschild erkannt wurden	- bei 6 von 9 Prüfabläufen ist das Stoppschild erkannt wurden
---	--	--	--

Der ACF-Detektor besitzt einen unendlichen Wertebereich, wobei ein Bestwert nicht explizit angegeben ist. Das Resultat sollte daher so hoch wie möglich ausfallen. Eine Aussage zur Sicherheit kann wegen des fehlenden Schwellenwertes nicht getroffen werden. Der Erkennungsalgorithmus reagiert zusätzlich sehr empfindlich auf die Veränderung des ROI oder die Spiegelung des Trainingsdatensatzes. Eine ähnliche Beobachtung wird bei der Positionsveränderung des Objektes im Testbild ersichtlich, wodurch diese eine unterschiedliche Kennzeichnung und Auffindungswert erhalten. Die beste Erkennung des Testbildes von Matlab ist mit den weiten Anlernbildern zu erreichen.

Der Fast-R-CNN-Detektor hat einen Wertebereich von 0 bis 1. Die 1 widerspiegelt die sicherste Erkennung und ist demzufolge anzustreben. Die Kennzeichnung des ROI nimmt hierbei wenig Einfluss auf das Ergebnis. In der Positionsveränderung des Testbildes oder der Spiegelung der Verkehrszeichen im Anlernprozess sind Veränderungen der Auffindungswerte ersichtlich. Diese haben daher eine Auswirkung auf die Wirksamkeit des Algorithmus. Die nahen und weiten Anlernbilder haben die größte Wirkung auf dessen Effektivität, wodurch hiermit das Beispielbild am sichersten erkannt wird.

Der Cascade-Detektor besitzt lediglich einen Begrenzungsrahmen ohne die Angabe eines Erkennungswertes. Die Einflussnahme des ROI ist bei dem Testlauf ersichtlich, wobei mit dem Erkennungsschwellwert entgegengewirkt werden kann. Die Spiegelung des Trainingsdatensatzes erreicht ebenfalls unterschiedliche Resultate. Die Positionsveränderung ruft hingegen minimale Abweichungen hervor, was bei den anderen Algorithmen ausgeprägter ist. Das Testbild kann hierbei mit den weit entferntesten Anlernbildern am besten erkannt werden.

7 Auswertung

Es sind für die Bewertung der Algorithmen folglich weitere Fotosammlungen für den Prüfablauf nötig. Als Grundlage des Trainingsvorgangs dienen dabei die nahen und weiten Stoppschilder aus den Aufnahmefahrten, wodurch eine Datenbank von maximal 287 Bildern zur Verfügung steht. Eine Abstufung wird durch die Anzahl der genutzten Fotos erlangt, wodurch unterschiedliche Erkennungsalgorithmen für die Auswertung bereitstehen. Die Bewertung der gewählten und erzeugten Detektoren kann mit zwei unterschiedlichen Testdatensätzen vollzogen werden. Es sind demzufolge die benötigten zwei Ordner angelegt, wobei als Vorlage die übliche Aufteilung von 80 % Trainingsdaten zu 20 % Testdaten vorgenommen ist. Eine Bestimmung der genutzten Anlernbilder ist für jede Abstufung zu erstellen, was aber keinen direkten Vergleich derer zulassen würde. Der Anteil ist deshalb lediglich aus der maximalen Anzahl der Trainingsdaten zu bestimmen. Diese sind folglich für jeden Prüfdurchlauf vollständig zu nutzen, wodurch der Einfluss ersichtlich wird. Das Ergebnis liefert hierfür 58 Testbilder, was in Abbildung 7.1 nachzuvollziehen ist [32].

$$287 \text{ Bilder} \cdot \frac{20 \%}{100 \%} = 57,4 \approx 58 \text{ Testbilder}$$

Abbildung 7.1: Berechnung Testbilder für Auswertung

Für eine breitere Analyse von Testfällen sind diese in 20 Stopp- und 20 Verkehrsschilderfotos aufgeteilt, um eine richtige und falsche Detektion prüfen zu können. Die verbleibenden Fotografien weisen keine derartigen Umgebungszeichen auf, wodurch dieser Fall ebenso abgedeckt wird. Ein zweiter angelegter Prüfordner enthält ausschließlich Aufnahmen mit Stoppschildern. Die Wirksamkeit kann damit explizit mit einer höheren Anzahl untersucht werden. Der Testdatensatz des Institutes für Neuroinformatik und der Beispieldatensatz von Matlab liefern hierfür die benötigten Bildaufnahmen für die Erstellung [24].

Des Weiteren bleiben die meisten Standardeinstellungen der entsprechenden Erkennungsalgorithmen unverändert, um eine eventuell daraus resultierende bessere Anpassung auszuschließen. Eine Filterung erfolgt lediglich durch den maximalen Wert oder dem kleinsten Markierungsbereich, was eine entsprechende Bewertungsmöglichkeit der einzelnen Detektoren zulässt. Die Treffergenauigkeit, Wirksamkeit, Trennfähigkeit und der MCC-Wert sind nach den Grundlagen des Kapitels 3.2 berechnet, um die dabei erreichten Resultate vergleichen zu können.

7.1 Bewertung des ACF-Detektors

Folglich ist der ACF-Detektor bei den angelegten Testfällen zu überprüfen, wessen erreichte Ergebnisse in der nachfolgenden Abbildung 7.2 zu sehen sind.

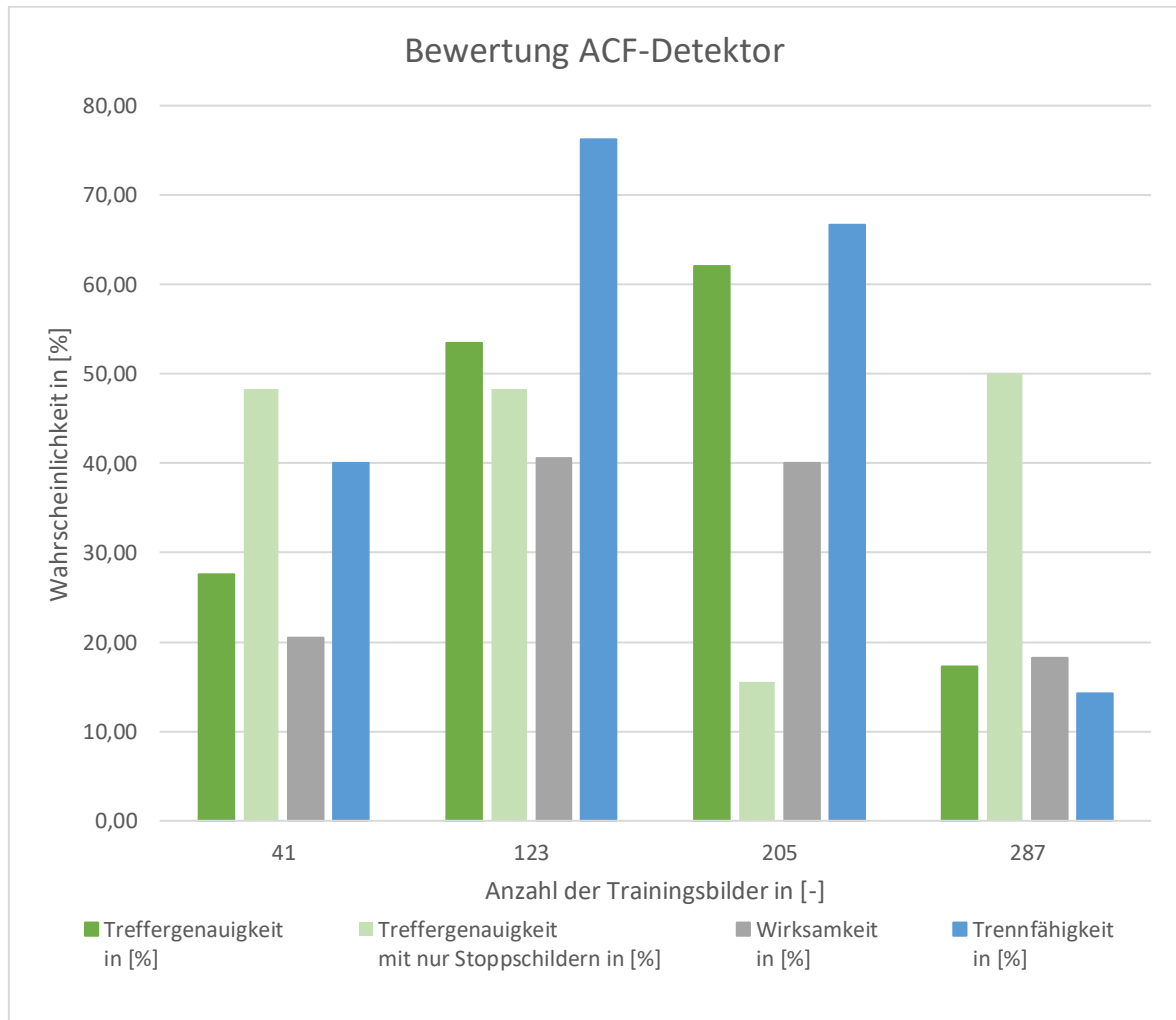


Abbildung 7.2: Bewertungsergebnis ACF-Detektor

Die berechneten Wahrscheinlichkeiten steigen kontinuierlich bis 123 Trainingsbildern an und sinken erst dann wieder ab. Ein Grund dafür können die eher weiten Aufnahmen beim größeren Datensatz sein, die sich negativ auf den Anlernprozess und der Erkennung dieses Detektors auswirken. Die Treffergenauigkeit erreicht einen maximalen Wert von circa 62 Prozent, was eine Auskunft zur richtigen Markierung der vorhandenen Stoppschilder ausgibt. In der Testdatenbank mit dauerhaft vorzufindendem Objekt ist maximal die Hälfte richtig gekennzeichnet. Die Wirksamkeit erzielt ebenfalls etwa 40 Prozent, was die korrekte Auffindung des entsprechenden Verkehrszeichens betrifft. Die Trennfähigkeit erreicht insofern den größten Wert mit über 76 Prozent, wobei ein nicht vorzufindendes Schild so vom Algorithmus interpretiert wird.

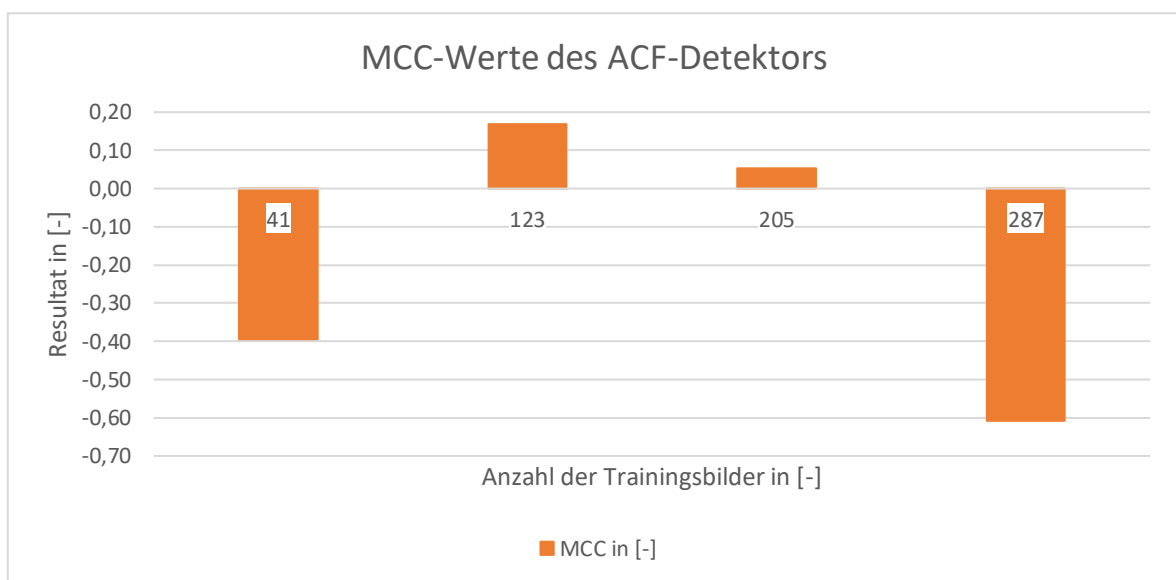


Abbildung 7.3: MCC-Werte des ACF-Detektors

Das Resultat des MCC-Wertes ist in Abbildung 7.3 vorzufinden, wobei ein maximaler Wert von 0,16 ersichtlich ist. Dieser sinkt mit zunehmender Anzahl von Trainingsbildern weiter ab, wodurch er sich als unzureichender Klassifikator auszeichnet.

Der entsprechende Positionsabstand ist zwischen Kamera und Objekt festgehalten, wodurch die Entfernungserkennung überprüft werden kann. In einer Aufnahmefahrt sind hierfür einzelne Bildsequenzen in einen Abstand von ca. 10 Meter extrahiert wurden.

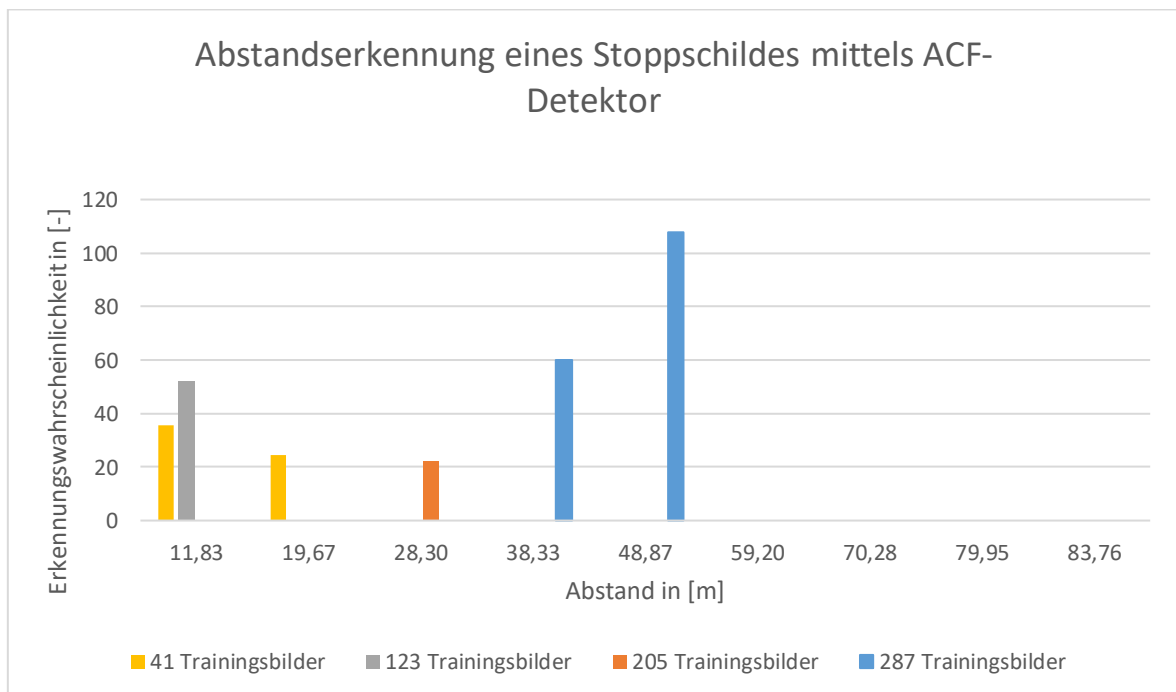


Abbildung 7.4: Abstandserkennung ACF-Detektor

Das Stoppschild wird im unmittelbaren Bereich erkannt, was in der Abbildung 7.4 ersichtlich ist. Erst mit steigender Anzahl von weit entfernten Trainingsbildern lässt sich die Distanz für eine erfolgreiche Detektion erhöhen. Der Nahbereich wird in diesem Fall wiederum schlechter, wodurch sich verschiedene Auslegungsmöglichkeiten ergeben. Die Bewertungspunkte sind damit geprüft, weswegen dieser mit den anderen verglichen werden kann.

7.2 Bewertung des Fast-R-CNN-Detektors

Der Fast-R-CNN Erkennungsalgorithmus wird mit den gleichen Testdatensätzen überprüft. In der nachfolgenden Abbildung 7.5 sind die erlangten Resultate vorzufinden.

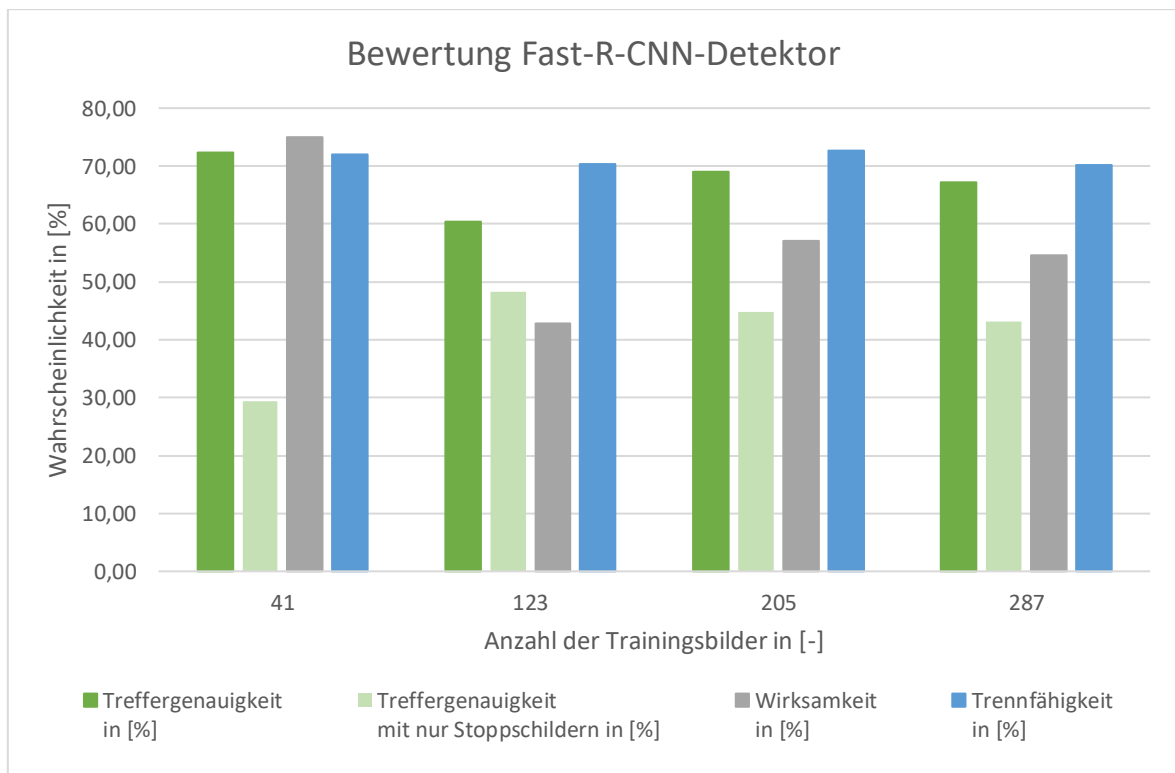


Abbildung 7.5: Bewertung Fast-R-CNN-Detektor

Die Treffergenauigkeit erlangt einen Wert von ungefähr 70 Prozent, wodurch das gesuchte Zeichen oft richtig markiert ist. Eine Verschlechterung der Ergebnisse ist bei der Betrachtung von reinen Stoppschilderbildern zu erkennen. Die Wirksamkeit erreicht am Anfang ein maximales Resultat von 75 Prozent, was sich dann auf näherungsweise 54 Prozent stabilisiert. Der Anteil an falschen Erkennungen steigt demzufolge und das Objekt wird weniger erkannt. Ein hohes Ergebnis von ungefähr 72 Prozent ist bei der Trennfähigkeit zu beobachten, was auf wenig fehlerhafte

Detektionen schließen lässt. Ist insofern kein Verkehrszeichen vorhanden, dann wird es dem Anwender auch so übermittelt.

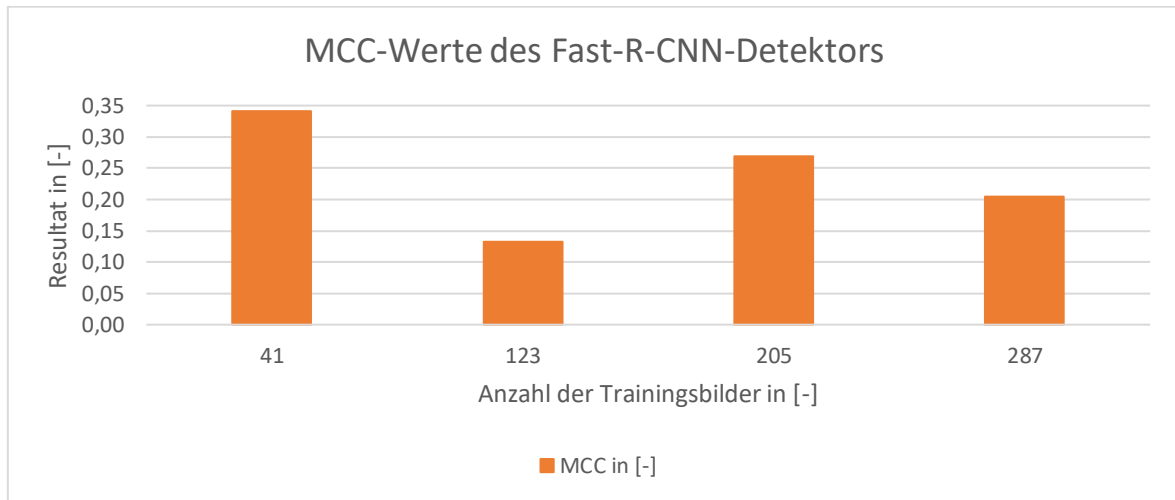


Abbildung 7.6: MCC-Werte des Fast-R-CNN-Detektors

In Abbildung 7.6 sind die erreichten MCC-Werte zu erkennen. Es wird hierbei ein maximaler Wert von 0,34 erreicht. Dieses Resultat sinkt mit Zunahme der Trainingsbilder, obwohl es stets im positiven Bereich verbleibt und folglich einen besseren Klassifikator widerspiegelt.

In der Bewertung der Abstandserkennung sind die extrahierten Bilder der Aufnahmefahrt zu nutzen. Das Ergebnis ist in Abbildung 7.7 zu finden.

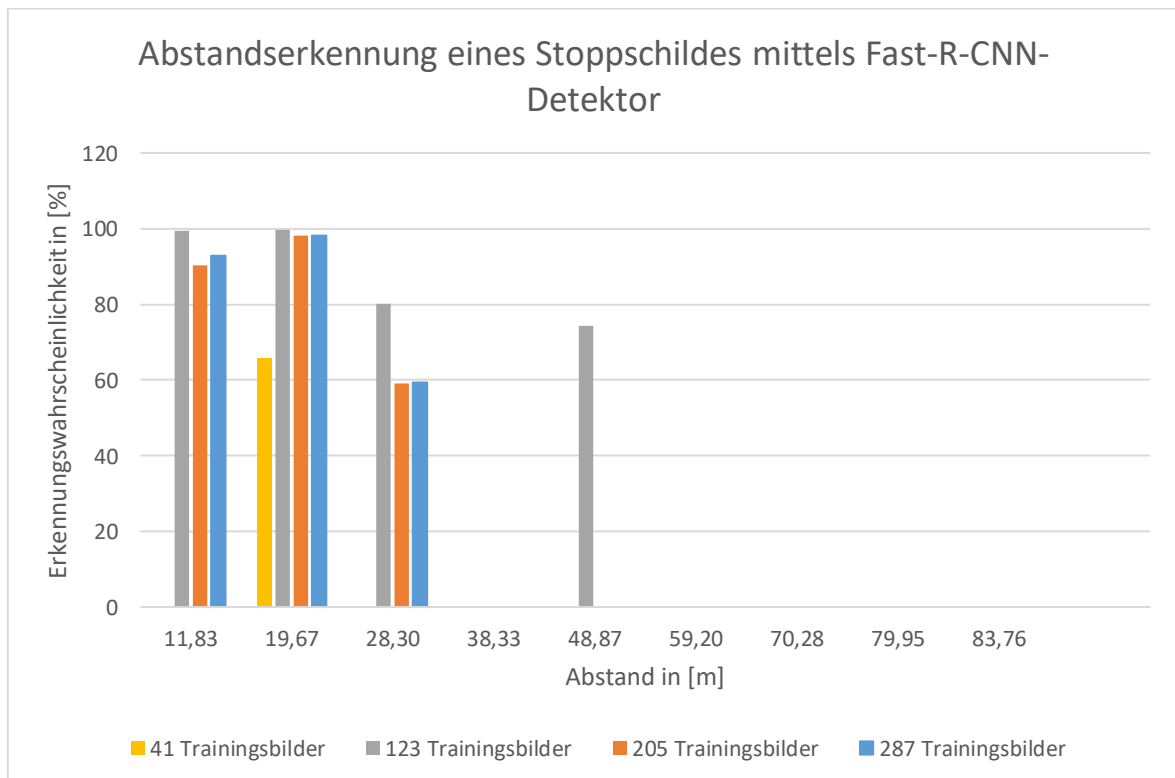


Abbildung 7.7: Abstandserkennung R-CNN Detektor

Im kurzen Bereich bis etwa 30 Metern wird das Verkehrszeichen dauerhaft erkannt, wobei ein Trainingsdatensatz von mindestens 123 Bildern benötigt wird. Der Algorithmus weist zudem eine beständig hohe Erkennungswahrscheinlichkeit auf, weswegen er für diesen Testfall gut geeignet scheint. Eine zweistellige Anzahl von Anlernbildern verschlechtert wiederum den Detektor enorm, weswegen eine größere Fotosammlung anzustreben ist.

7.3 Bewertung des Cascade-Detektors

Der Cascade-Detektor ist ebenfalls mit den beiden Testdatensätzen zu überprüfen, wobei ein Erkennungsschwellenwert von 40 festzulegen ist. Dieser Einstellwert ist damit zu begründen, dass zu viele Fehldetektionen im ersten Prüfdurchlauf zu erkennen sind. Eine Bewertung und Vergleich sind demzufolge nicht möglich, weshalb diese Erhöhung unumgänglich ist. Die Ergebnisse sind mit angepasster Standardeinstellung in Abbildung 7.8 vorzufinden.

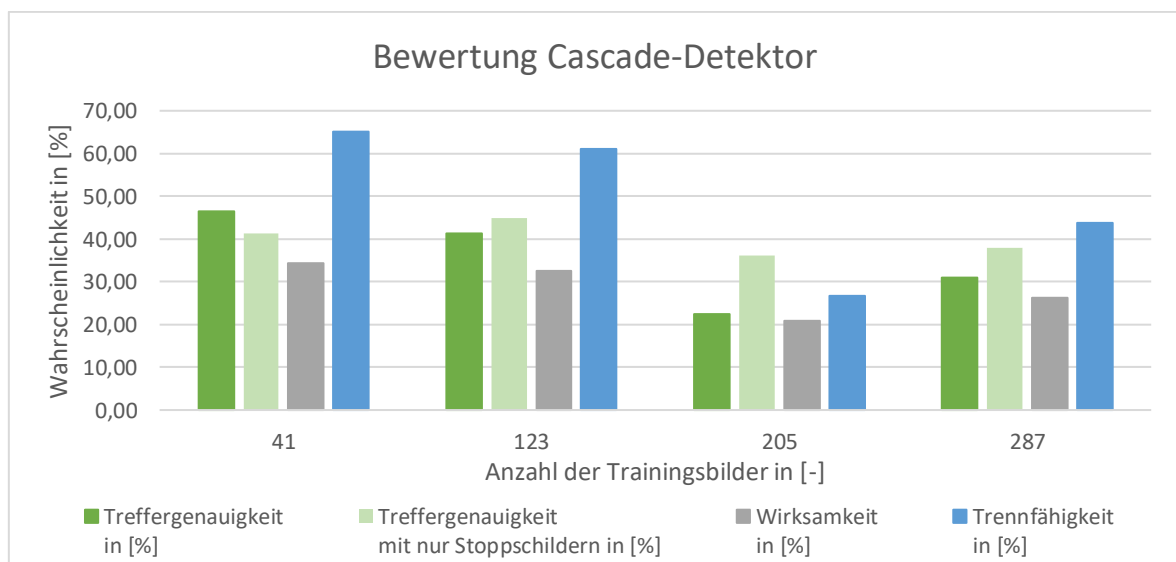


Abbildung 7.8: Bewertung Cascade-Detektor

Dieser weist eine Treffergenauigkeit von bestenfalls 46 Prozent auf, wodurch fast die Hälfte der Bilder richtig zugeordnet sind. Im Datensatz mit immer wieder vorzufindenden Stoppschildern ist ein maximaler Wert von 44 Prozent zu beobachten, obwohl mit der Zunahme an Trainingsbildern ein besseres Resultat erkenntlich wird. Die Wirksamkeit erreicht am Anfang ungefähr ein Drittel des angestrebten Zielergebnisses und sinkt dann leicht auf 26 Prozent. Grund dafür ist der steigende Anteil des nicht vorzufindenden Schildes im Testbild, wobei der Algorithmus fatalerweise eine Markierung setzt. Bei der Trennfähigkeit ist eine sehr stark abfallende Kurve zu erkennen, die erst bei der letzten Zunahme von

Anlernfotos wieder zunimmt. Das vorhandene Stoppschild wird folglich nicht mehr erkannt, weswegen der falsche negative Anteil steigt. Ein negativer Einfluss der weit entfernten Trainingsbilder ist zu erkennen, wodurch die Ergebnisse abfallen und erst mit Zunahme der Fotos wieder langsam steigen.

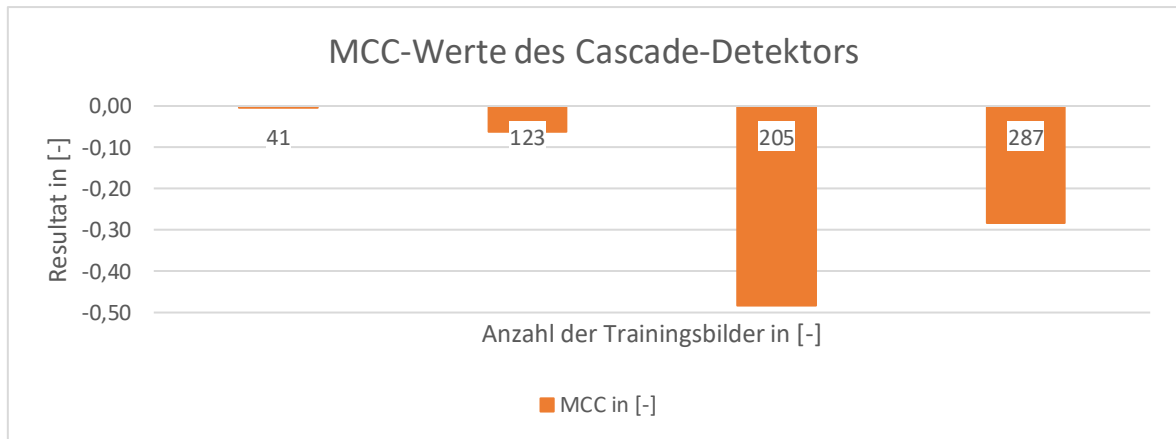


Abbildung 7.9: MCC-Werte des Cascade Detektors

Die berechneten MCC-Werte sind in Abbildung 7.9 zu finden, wobei dieser lediglich negative Resultate vorweist und daher für perfekte Klassifizierungen des Objektes ungeeignet ist.

Die Abstandserkennung ist ebenfalls näher zu betrachten, was zu nachfolgendem Ergebnis in der Abbildung 7.10 führt.

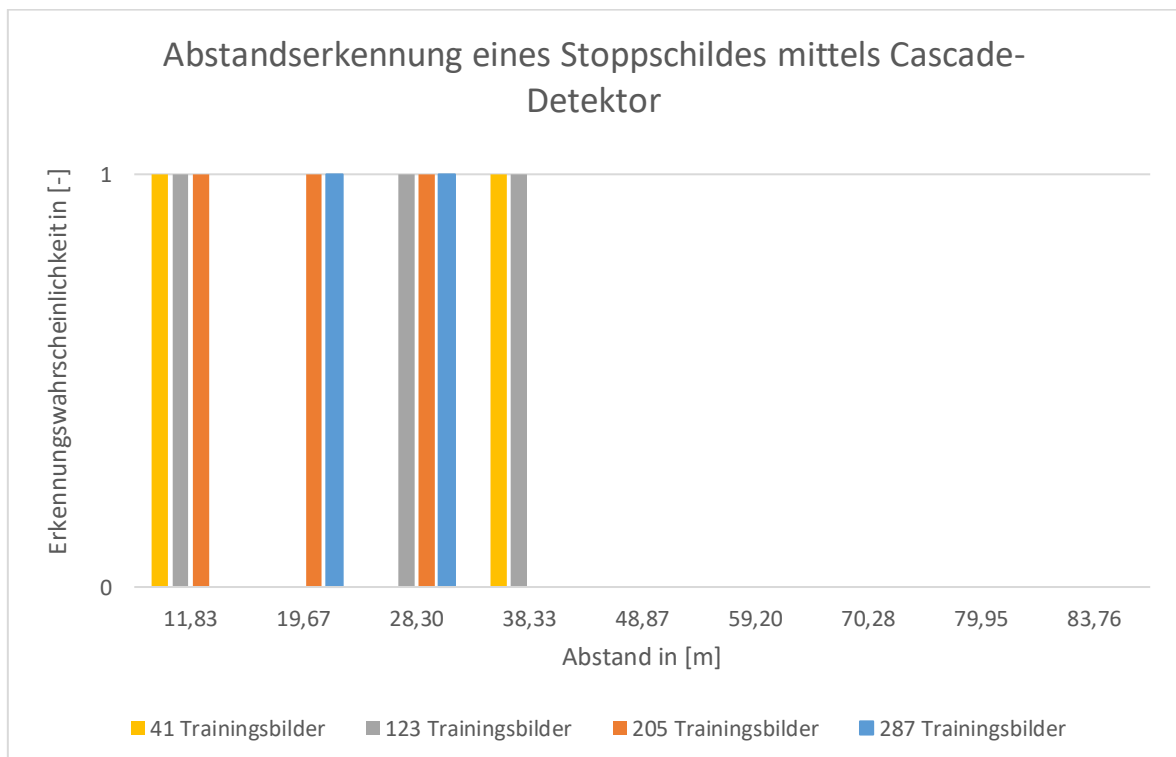


Abbildung 7.10: Abstandserkennung Cascade-Detektor

Das beste Ergebnis wird mit 205 Trainingsbildern erzielt, wobei sehr viele falsche Detektionen im ungefilterten Prüferesultat erkennbar sind. Das zu markierende Stoppschild wird bis zu einem Abstand von circa 40 Metern stellenweise erkannt. Die in diesem Bereich falsch detektierten Objekte sind eine Folge der Filterung. Das entsprechende Resultat ist daher sehr zufallsbehaftet, weil es sich auf den kleinsten Kennzeichnungsbereich konzentriert. Bei kurzen Distanzen ist eher das Maximum zu betrachten, was folglich mit einer ständigen Anpassung verbunden wäre. Dies ist zu dem derzeitigen Stand nicht anders zu realisieren, weswegen der kleinste Markierungsrahmen als dauerhafte Filterung dient.

7.4 Resultat der Auswertung

Der Cascade-Detektor ist für die Auffindung eines Stoppschildes nicht geeignet. Es fehlen zum einen die entsprechenden Zahlenwerte der Detektionen und zum anderen ermöglicht dieser nur eine Filterung über den Begrenzungsrahmen. In der Auswertung der entsprechenden Ergebnisse wird dies ebenfalls ersichtlich.

Der ACF-Detektor weist ein ähnliches Resultat auf, obwohl dieses mit der Zunahme von weit entfernten Trainingsbildern weiter sinkt. Ein Einfluss der Reihenfolge ist ebenfalls zu beobachten, wofür es keine Vorgabe gibt und daher in dieser Auswertung erneut hervorzuheben ist. Ein Mindestwert ist bei diesem großen Erkennungswertebereich schwer festlegbar, wodurch dieser ebenfalls nicht für die Markierung eines Verkehrsschildes einzusetzen ist.

Die im Vergleich besten und kontinuierlichsten Ergebnisse liefert der Fast-R-CNN-Detektor. Dieser erkennt in den meisten Fällen das davor befindliche Objekt und erreicht bei der MCC-Betrachtung den Bestwert im Prüfprozess. Demzufolge ist dieser als Klassifikator geeignet und weist überdies weitere Verbesserungsmöglichkeiten auf. Es sind dabei die vielen vorzufindenden Trainingsoptionen zu nennen, welche zu einer Einflussnahme des Testergebnisses führt. Ein Mindestwert für den Anwendungsfall lässt sich ebenfalls näherungsweise bestimmen, wodurch dieser für die Erkennung eines Verkehrszeichens weiter betrachtet werden sollte.

8 Zusammenfassung und Ausblick

Die vorliegende Diplomarbeit beinhaltet die Erstellung einer geeigneten Bilddatenbank für maschinelle Lernverfahren mittels eigener Aufnahmefahrten. Es wurde dabei eine Möglichkeit der Fusion eines Lidar-Sensors erarbeitet, um den Abstand zum entsprechenden Objekt auszugeben. Die aus den Routen resultierende Bilddatenbank dient als Grundlage einer genaueren Untersuchung von verschiedenen Erkennungsansätzen, die als Beispiele in Matlab vorzufinden sind. Infolgedessen sind diese mit verschiedenen Anzahlen von Bildern trainiert und getestet wurden, wodurch ein Vergleich und eine Auswertung durchführbar sind. Die besten Ergebnisse liefert der Fast-R-CNN-Detektor, woraus sich eine optimale Nutzung für die Erkennung eines Verkehrszeichens ableitet. Des Weiteren bietet dieser die Chance für weitere Verbesserungsmöglichkeiten, um das schon erzielte Resultat noch weiter zu optimieren. Es sind dabei die zusätzliche Filterung durch einen Schwellenwert und die weiteren Trainingsoptionen zu nennen. Es ist zudem eine Erweiterung des Anlern Datensatzes vorstellbar, um eine Beeinflussung von verschmutzten, beschädigten oder gar verdeckten Objekten zu überprüfen. Dies ist bei den Aufnahmefahrten ersichtlich geworden, wo durch Vandalismus bei einer Vielzahl von Stoppschildern der beispielhafte Aufkleber „EATING ANIMALS“ zu entdecken war. Diese sind bewusst nicht im Trainingsdatensatz enthalten, um keine zusätzlichen Beeinflussungsgrößen einzubringen. Die Auswirkung solcher Bilder und deren Auffindung im Alltag ist demzufolge noch zu untersuchen. Dies gilt gleichermaßen für die verschiedenen Witterungsverhältnisse oder gar Nachtfahrten, welche ebenfalls noch nicht überprüft sind. Ein weiterer Bilddatensatz aus zusätzlichen Aufnahmefahrten ist deshalb zu erstellen. Die Algorithmen ermöglichen überdies lediglich die Auffindung eines bestimmten Verkehrszeichens, was zugleich keinen Einsatz in einem zukünftigen Mobilitätsangebot zulässt. Die Eingabe- und Ausgabeschicht sollte daher erweitert werden, um mehrere Straßenschilder detektieren zu können. Die aufbereitete Datenbank aus dem Internet ist hierfür nutzbar, aber deckt dennoch nicht alle vorzufindenden Zeichen in Deutschland ab.

Literatur- und Quellenverzeichnis

- [1] Koch, Alexander:
Diese Systeme sind ab 6. Juli 2022 im Auto Pflicht; Bauer Xcel Media Deutschland KG; Hamburg; <https://www.autozeitung.de/automatische-bremssysteme-181544.html>; 2022; Abruf: 02.08.2022.
- [2] Baumann, Uli; Wittich, Holger:
Die wichtigsten Antworten zur EU-Tempobremse; Motor Presse Stuttgart GmbH & Co. KG; Stuttgart; <https://www.auto-motor-und-sport.de/verkehr/isa-automatisches-tempolimit-ab-2022/>; 2021; Abruf: 02.08.2022.
- [3] o.V.:
Verkehrszeichenerkennung: Orientierungshilfe im Schilderwald; VFR Verlag für Rechtsjournalismus GmbH; Berlin;
<https://www.bussgeldkatalog.org/verkehrszeichenerkennung/>; 2022; Abruf: 02.08.2022.
- [4] o.V.:
Unterschied zwischen Clustering und Klassifizierung; Sawakinome; o.O.;
<https://www.sawakinome.com/articles/database/difference-between-clustering-and-classification.html>; 2022; Abruf: 20.07.2022.
- [5] Erhardt, Angelika:
Einführung in die digitale Bildverarbeitung; Auflage 1.; Vieweg + Teubner; Wiesbaden; 2008.
- [6] Aust, Holger:
Das Zeitalter der Daten; Springer Berlin Heidelberg; Berlin, Heidelberg; 2021.
- [7] Stuart, Herbert A.; Klages, Gerhard:
Kurzes Lehrbuch der Physik; Springer Berlin Heidelberg; Berlin, Heidelberg; 2010.
- [8] Döllinger, Christian; Aufschnaiter, Monika:
Optische Linsen in der Fotografie; Bayerischer Rundfunk; München;
<https://www.br.de/alphalernen/faecher/physik/2-optische-linsen-fotografie100.html>; 2021; Abruf: 01.07.2022.
- [9] Werner, Martin:
Digitale Bildverarbeitung; Springer Fachmedien Wiesbaden; Wiesbaden; 2021.
- [10] Grasel, Edgar:
Die Streifen am Bildschirm und das Histogramm; Wieselfilm Videoproduktion; Linz; <https://www.wieselfilm.net/tipps-tricks/die-streifen-am-bildschirm-und-das-histogramm/>; o.J.; Abruf: 01.07.2022.

- [11] Pietrzyk, Johannes; Worpitz, Benjamin:
Histogramme & Punktoperatoren; o.O.; https://wwwpub.zih.tu-dresden.de/~photo/teachlets/Histogramme_Punktoperatoren/index.htm; 2014;
Abruf: 02.07.2022.
- [12] Choo, Kenny; Greplova, Eliska; Fischer, Mark H.; Neupert, Titus:
Machine Learning kompakt; Springer Fachmedien Wiesbaden; Wiesbaden;
2020.
- [13] Matzka, Stephan:
Künstliche Intelligenz in den Ingenieurwissenschaften; Springer Fachmedien
Wiesbaden; Wiesbaden; 2021.
- [14] Chicco, Davide; Jurman, Giuseppe:
The advantages of the Matthews correlation coefficient (MCC) over F1 score
and accuracy in binary classification evaluation; o.O.;
<https://bmcbgenomics.biomedcentral.com/articles/10.1186/s12864-019-6413-7>;
2020; Abruf: 19.06.2022.
- [15] Dollár, Piotr; Appel, Ron; Belongie, Serge; Perona, Pietro:
Fast Feature Pyramids for Object Detection; Auflage 36.; IEEE Transactions
on Pattern Analysis and Machine Intelligence; o.O.; 2014.
- [16] o.V.:
Get Started with Cascade Object Detector; o.O.;
<https://de.mathworks.com/help/vision/ug/train-a-cascade-object-detector.html>;
o.J.; Abruf: 16.05.2022.
- [17] o.V.:
Cascade Classifier Training; o.O.;
https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html; o.J.;
Abruf: 10.08.2022.
- [18] o.V.:
Funktionsweise und Einsatz von Rolling Shutter CMOS-Kameras; o.O.;
<https://www.baumer.com/ch/de/service-support/funktionsweise/funktionsweise-und-einsatz-von-cmos-rolling-shutter-kameras/a/CMOS-rolling-shutter-cameras>; o.J.; Abruf: 15.04.2022.
- [19] o.V.:
Gigabit Ethernet und GigE Vision; Basler AG; Ahrensburg;
<https://www.baslerweb.com/de/vision-campus/schnittstellen-und-standards/gigabit-ethernet/>; o.J.; Abruf: 18.05.2022.
- [20] Charlene:
Was ist ein PoE-Switch?; FS.COM GmbH; München;
<https://community.fs.com/de/blog/what-is-a-poe-switch-and-how-does-it-work.html>; o.J.; Abruf: 14.07.2022.

- [21] o.V.:
16 Port Unmanaged Gigabit Ethernet PoE+ Switch mit FlexPoE (76W); o.O.;
<https://www.netgear.com/de/business/wired/switches/unmanaged/gs116lp/>;
o.J.; Abgerufen: 09.08.2022.
- [22] Eichfelder, Marius:
Ab wie viel FPS läuft ein Computerspiel flüssig?; BurdaForward GmbH;
München; https://praxistipps.chip.de/ab-wie-viel-fps-laeuft-ein-computerspiel-fluessig_13580; 2013; Abruf: 12.05.2022.
- [23] o.V.:
A Look Inside the BMW i3 Interior Design!; Modern in Denver; Denver;
<https://www.modernindenver.com/2014/12/i3-interior/>; 2014; Abruf
23.06.2022.
- [24] o.V.:
The German Traffic Sign Detection Benchmark; Bochum;
https://benchmark.ini.rub.de/gtsdb_dataset.html; 2019; Abruf: 04.05.2022.
- [25] o.V.:
trainACFObjectDetector; o.O.;
<https://de.mathworks.com/help/vision/ref/trainacfobjectdetector.html>; o.J.;
Abruf: 02.05.2022.
- [26] o.V.:
Choose an Object Detector; o.O.;
<https://de.mathworks.com/help/vision/ug/choose-an-object-detector.html>; o.J.;
Abruf: 02.05.2022.
- [27] o.V.:
Train ACF-Based Stop Sign Detector; o.O.;
<https://de.mathworks.com/help/vision/ug/train-acf-based-stop-sign-detector.html>; o.J.; Abruf: 02.05.2022.
- [28] o.V.:
Train Fast R-CNN Stop Sign Detector; o.O.;
<https://de.mathworks.com/help/vision/ug/train-fast-r-cnn-stop-sign-detector.html>; o.J.; Abruf: 02.05.2022.
- [29] o.V.:
trainCascadeObjectDetector; o.O.;
<https://de.mathworks.com/help/vision/ref/traincascadeobjectdetector.html>; o.J.;
Abruf: 02.05.2022.
- [30] o.V.:
Änderung der Vorfahrtsregelung zugunsten der Radfahrer; o.O.;
<https://nationaler-radverkehrsplan.de/de/praxis/aenderung-der-vorfahrtsregelung-zugunsten-der-radfahrer>; o.J.; Abruf: 05.05.2022.
- [31] Krizhevsky, Alex:
Der CIFAR-10-Datensatz; o.O.; <https://www.cs.toronto.edu/~kriz/cifar.html>;
o.J.; Abruf: 20.05.2022.

[32] Gholamy, Afshin; Kreinovich, Vladik; Kosheleva, Olga:
Why 70/30 or 80/20 Relation Between Training and Testing Sets: A
Pedagogical Explanation; El Paso;
https://scholarworks.utep.edu/cs_techrep/1209/; 2018; Abruf: 01.08.2022.

Eidesstattliche Erklärung

Das vorliegende Dokument wurde an der Hochschule für Technik und Wirtschaft Dresden unter der Leitung der Herren Prof. Dr. rer. nat. Toralf Trautmann (Hochschule für Technik und Wirtschaft Dresden) und Dipl.-Ing. (FH) Dirk Engert (Hochschule für Technik und Wirtschaft Dresden) angefertigt.

Hiermit erkläre ich, dass ich die vorliegende Arbeit zu dem Thema

„Entwicklung einer dynamischen Bilddatenbank für maschinelle Lernverfahren“

selbstständig und ohne Benutzung anderer Quellen und Hilfsmittel als angegeben angefertigt habe. Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht habe.

Ferner gestatte ich der Hochschule für Technik und Wirtschaft Dresden, die vorliegende Diplomarbeit unter Beachtung insbesondere urheber-, datenschutz- und wettbewerbsrechtlicher Vorschriften für Lehre und Forschung zu nutzen.

Ort, Datum

Name, Vorname

Anlagenverzeichnis

Anlage 0	Überblick Symbole Skriptedarstellung	Seite 80
Anlage 4.1	Datenblatt Kamera Basler acA1920-40gc	Seite 81
Anlage 4.2	Skript Marvin_4_Aufnahme_Video.mlx	Seite 82
Anlage 5.1	Skript Auswertung_1_TabelleFuerVideoErstellen.mlx	Seite 83
Anlage 5.2	Skript Auswertung_2_Videodateien_Aufnahmefahrt.mlx	Seite 84
Anlage 5.3	Skript Labeler1_BildLabeler.mlx	Seite 85
Anlage 6.1	Resultate Untersuchung Detektoren	Seite 86
Anlage 7.1	Bewertung Detektoren	Seite 87
Anlage 7.2	Abstanderkennung Detektoren	Seite 88
Anlage 9.1	Datenträger Teil 1	Seite 89
Anlage 9.2	Datenträger Teil 2	Seite 90

Anlagen

Anlage 0



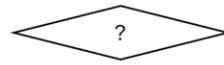
Ein-/Ausgabe



allgemeine Operation



Unterprogramm



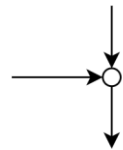
Verzweigung



Grenzstellen; Start/Ende



Verzweigung



Zusammenführung

Anlage 4.1

Sensor:

Sensorhersteller	Sony
Sensor	IMX249
Shutter-Technologie	Global Shutter
Sensorformat	1/1.2"
Sensortypen	CMOS
Sensorgröße	11,3 mm x 7,1 mm
Auflösung (HxV)	1920 px x 1200 px
Auflösung	2.3 MP
Pixelgröße (H x V)	5,86 µm x 5,86 µm
Bildrate	42 fps
Mono/Farbe	Color

Kameradaten:

Schnittstelle	GigE
Pixel Bit Depth	10 or 12 bits
Synchronisierung	▪ hardware trigger ▪ free-run ▪ Ethernet connection
Belichtungssteuerung	▪ programmable via the camera API ▪ hardware trigger
Digitaler Input	1
Digitaler Output	1
General Purpose I/O	1
Stromversorgung	▪ PoE or 12-24 VDC
Leistung (typisch)	2,8 W
Leistung PoE	3 W

Bauform:

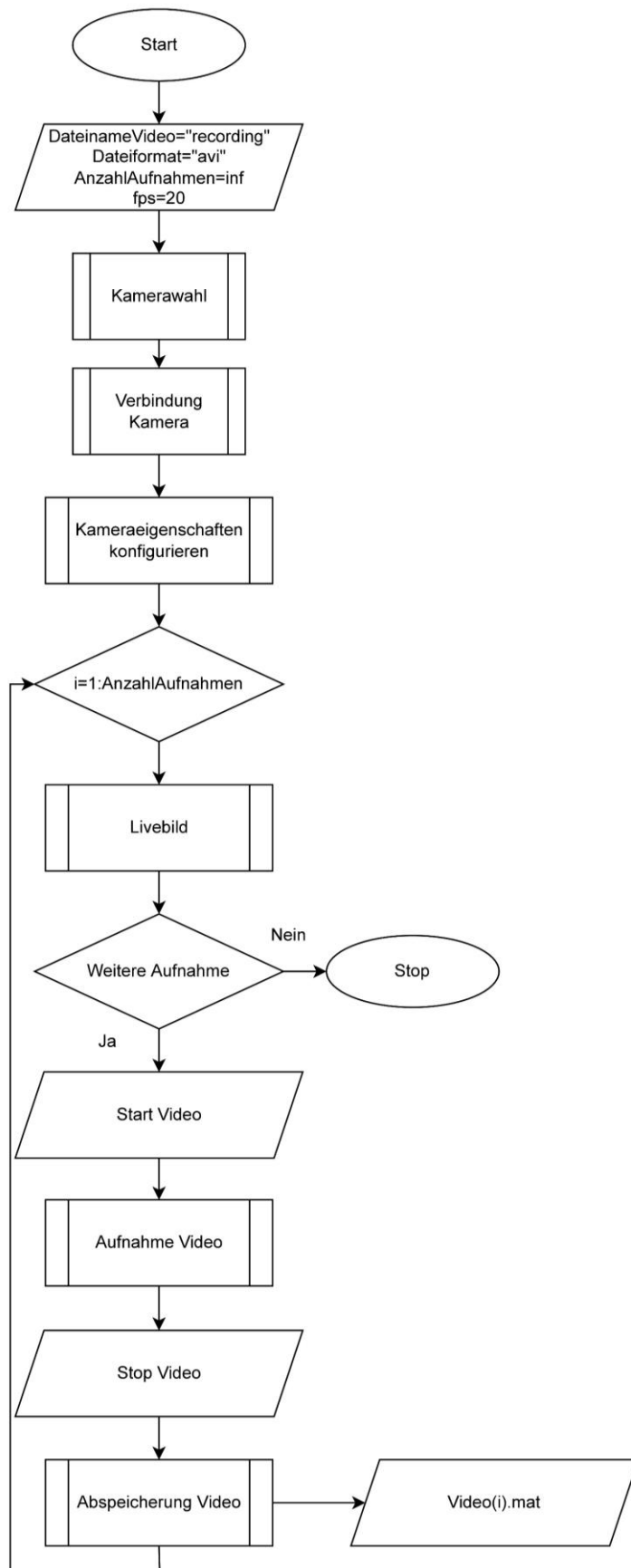
Housing Typ	Box
Größe (L x B x H)	42 mm x 29 mm x 29 mm
Objektivanschluss	▪ C-mount
Betriebstemperatur	0 - 50°C
Gewicht (typisch)	90g

Konformität:

Konformität	▪ CE, RoHS, GenICam ▪ GigE Vision, IP30, FCC ▪ UL, KC, EAC ▪ IEEE 802.3af (PoE)
-------------	--

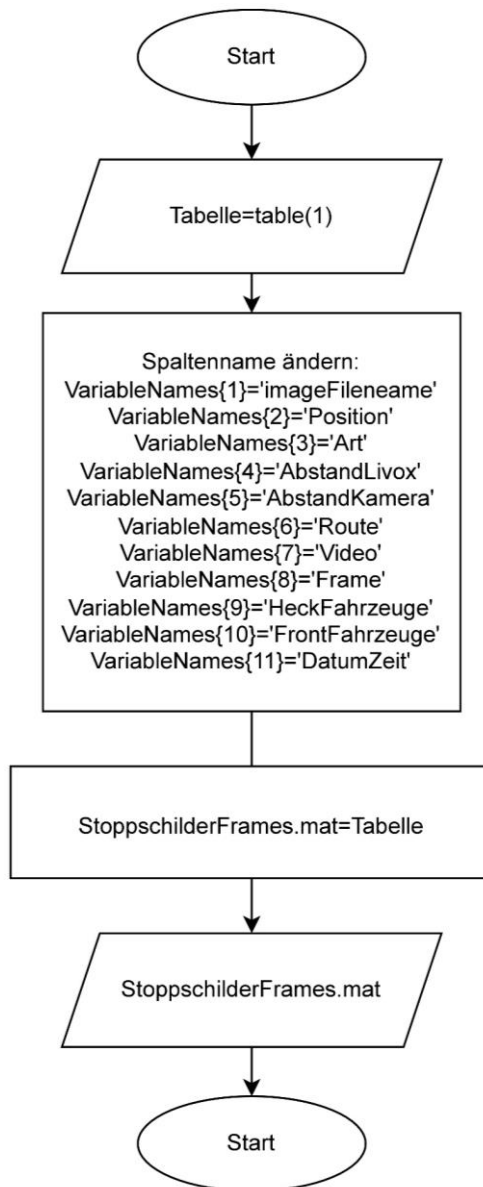
Anlage 4.2

Marvin_4_Aufnahme_Video.mlx



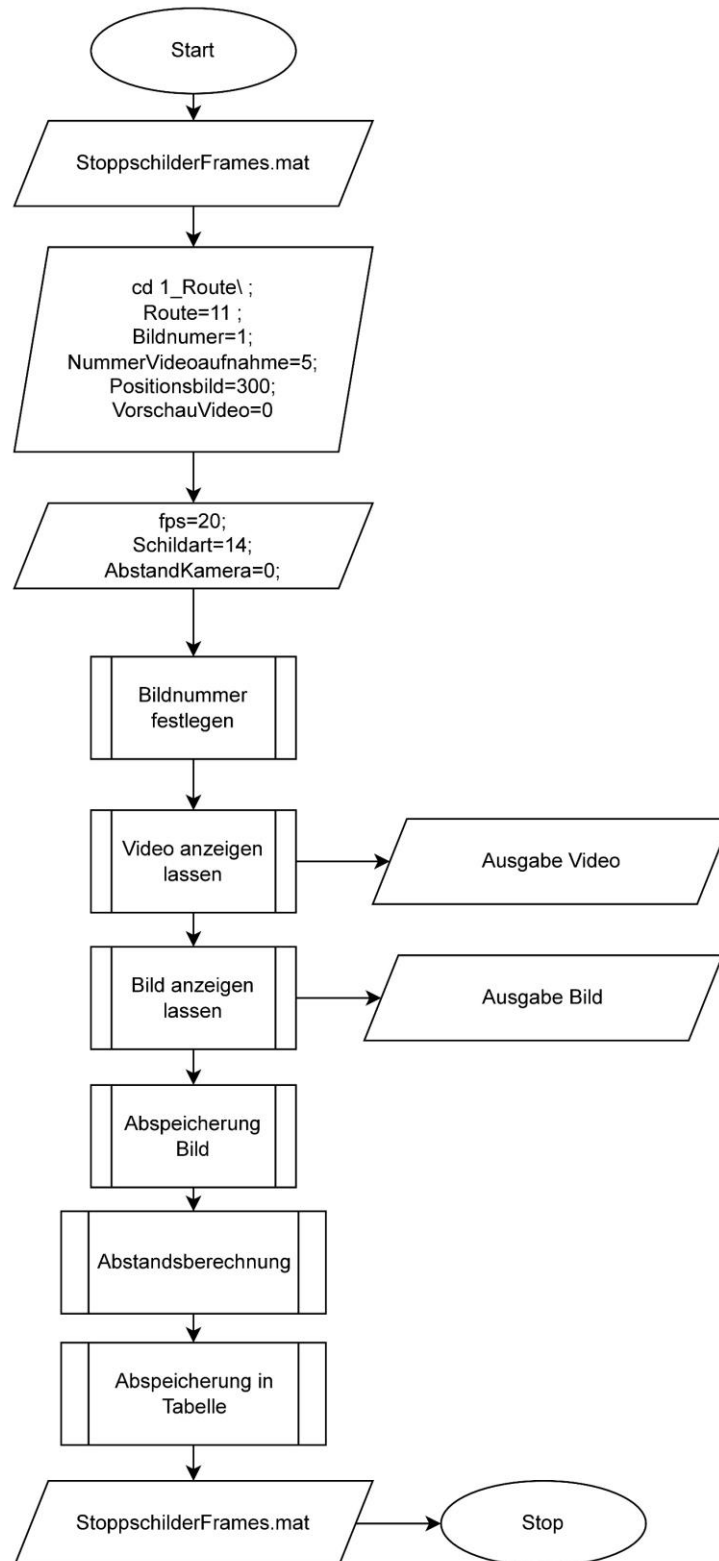
Anlage 5.1

Auswertung_1_TabelleFuerVideoErstellen.mlx

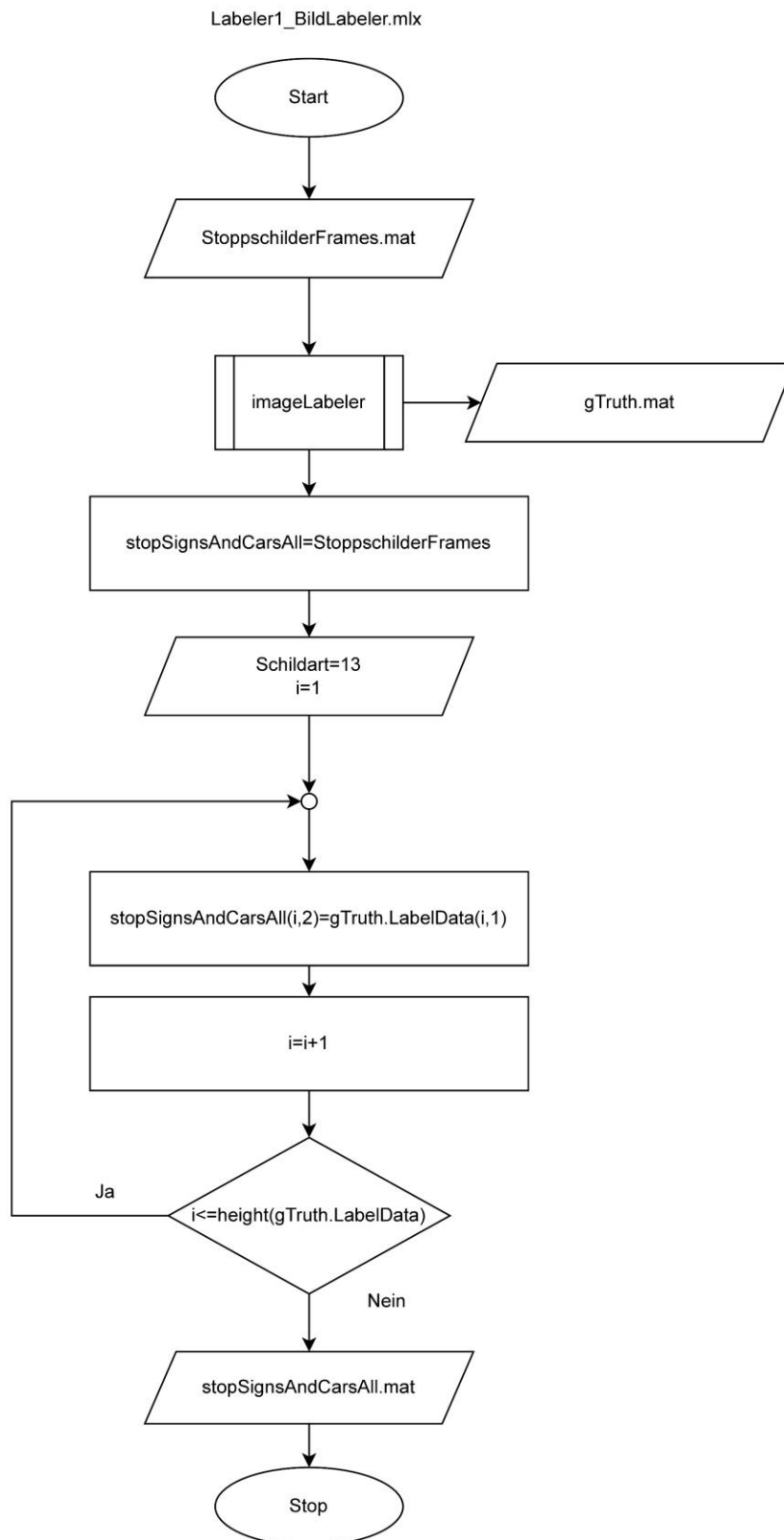


Anlage 5.2

Auswertung_2_Videodateien_Aufnahmefahrt.mlx



Anlage 5.3



Anlage 6.1

Bilder	Aufnahmefahrt nah (151 Bilder)	Bilder	Aufnahmefahrt weit (137 Bilder)	Bilder	Aufnahmefahrt nah und weit (287 Bilder)	Bilder	Vorfahrtsschilder (52 Bilder)	Bilder	Vorfahrtsschilder gespiegelt (52 Bilder)
41	0	41	74,9	41	0	41	0	41	97,4
82	0	82	12,4	82	0	52	111,3	52	86,7
123	27,4	123	0	123	27,4				
137	51,8	137	32	137	51,8				
151	0			151	0				
				164	0				
				205	0				
				246	0				
				287	0				

Fast-R-CNN-Detektor

Bilder	Aufnahmefahrt nah (151 Bilder)	Bilder	Aufnahmefahrt weit (137 Bilder)	Bilder	Aufnahmefahrt nah und weit (287 Bilder)	Bilder	Vorfahrtsschilder (52 Bilder)	Bilder	Vorfahrtsschilder gespiegelt (52 Bilder)
41	0	41	0	41	0	41	0,96716	41	0,9437
82	0,99299	82	0	82	0,99774	52	0,88248	52	0,93314
123	0,99709	123	0	123	0,99877				
137	0,99502	137	0	137	0,99248				
151	0,98989			151	0,99666				
				164	0,99721				
				205	0,98116				
				246	0,98135				
				287	0,98909				

Cascade-Detektor

Bilder	Aufnahmefahrt nah mit 4 (151 Bilder)	Bilder	Aufnahmefahrt weit mit 4 (137 Bilder)	Bilder	Aufnahmefahrt nah und weit mit 4 (287 Bilder)	Bilder	Vorfahrtsschilder (52 Bilder)	Bilder	Vorfahrtsschilder gespiegelt mit 4 (52 Bilder)
41	0	41	1	41	0	41	0	41	1
82	1	82	0	82	1	52	0	52	0
123	0	123	1	123	0				
137	0	137	1	137	0				
151	1			151	1				
				164	1				
				205	1				
				246	1				
				287	1				
				41					
				52					

Anlage 7.1

ACF-Detektor

Anzahl Bilder	Treffergenauigkeit in [%]	Wirksamkeit in [%]	Trennfähigkeit in [%]	Treffergenauigkeit mit nur Stoppschildern in [%]	MCC in [-]
41	27,59	20,51	40,00	48,28	-0,39
123	53,45	40,54	76,19	48,28	0,17
205	62,07	40,00	66,67	15,52	0,05
287	17,24	18,18	14,29	50,00	-0,61

Fast-R-CNN-Detektor

Anzahl Bilder	Treffergenauigkeit in [%]	Wirksamkeit in [%]	Trennfähigkeit in [%]	Treffergenauigkeit mit nur Stoppschildern in [%]	MCC in [-]
41	72,41	75,00	72,00	29,31	0,34
123	60,34	42,86	70,27	48,28	0,13
205	68,97	57,14	72,73	44,83	0,27
287	67,24	54,55	70,21	43,10	0,20

Cascade-Detektor

Anzahl Bilder	Treffergenauigkeit in [%]	Wirksamkeit in [%]	Trennfähigkeit in [%]	Treffergenauigkeit mit nur Stoppschildern in [%]	MCC in [-]
41	46,55	34,29	65,22	41,38	-0,01
123	41,38	32,50	61,11	44,83	-0,06
205	22,41	20,93	26,67	36,21	-0,48
287	31,03	26,19	43,75	37,93	-0,28

Anlage 7.2

ACF-Detektor

Abstand	287 Trainingsbilder	205 Trainingsbilder	123 Trainingsbilder	41 Trainingsbilder
11,83	0,00	0,00	52,20	35,70
19,67	0,00	0,00	0,00	24,40
28,30	0,00	22,50	0,00	0,00
38,33	60,00	0,00	0,00	0,00
48,87	107,70	0,00	0,00	0,00
59,20	0,00	0,00	0,00	0,00
70,28	0,00	0,00	0,00	0,00
79,95	0,00	0,00	0,00	0,00
83,76	0,00	0,00	0,00	0,00

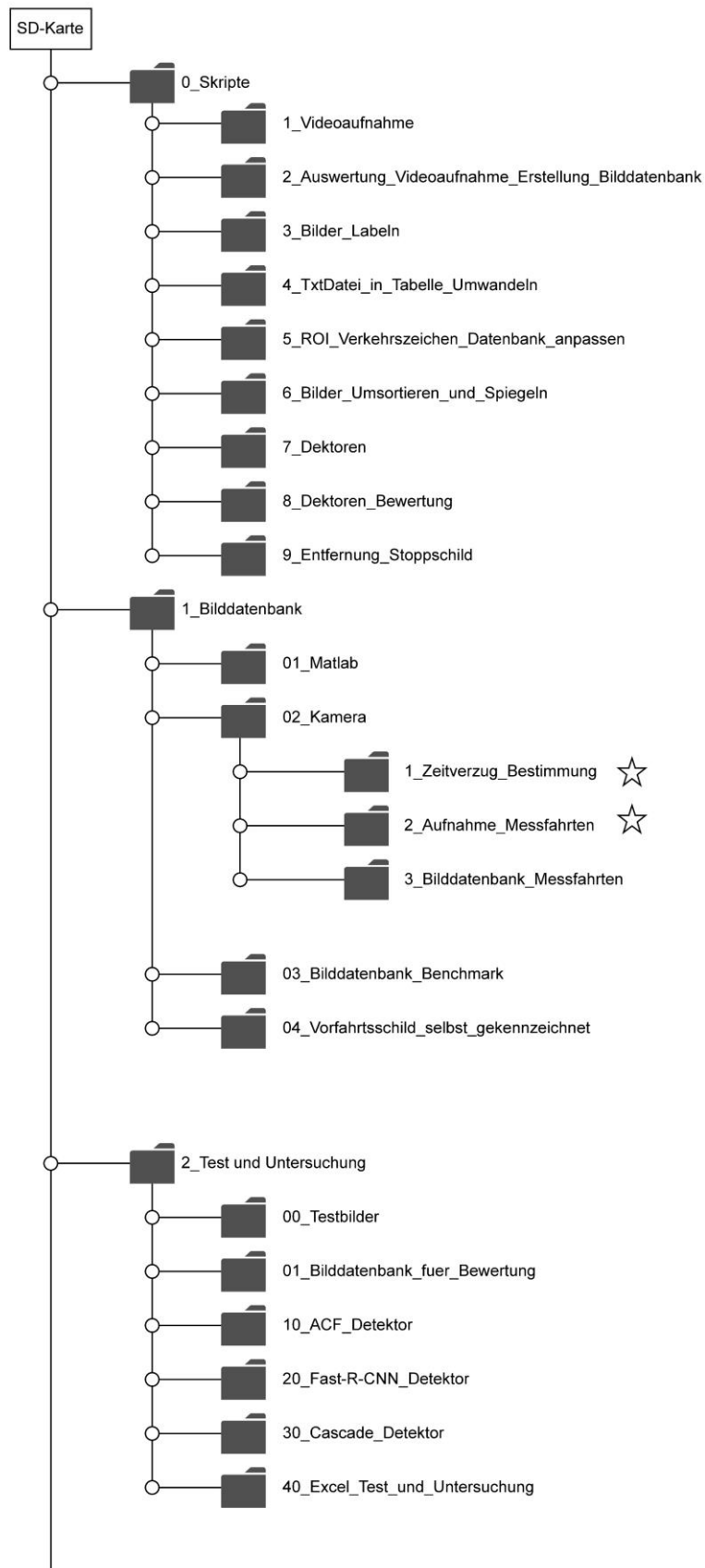
R-CNN-Detektor

Abstand	287 Trainingsbilder	205 Trainingsbilder	123 Trainingsbilder	41 Trainingsbilder
11,83	92,80	90,31	99,35	0,00
19,67	98,05	98,06	99,69	65,57
28,30	59,30	59,02	80,19	0,00
38,33	0,00	0,00	0,00	0,00
48,87	0,00	0,00	74,39	0,00
59,20	0,00	0,00	0,00	0,00
70,28	0,00	0,00	0,00	0,00
79,95	0,00	0,00	0,00	0,00
83,76	0,00	0,00	0,00	0,00

Cascade-Detektor

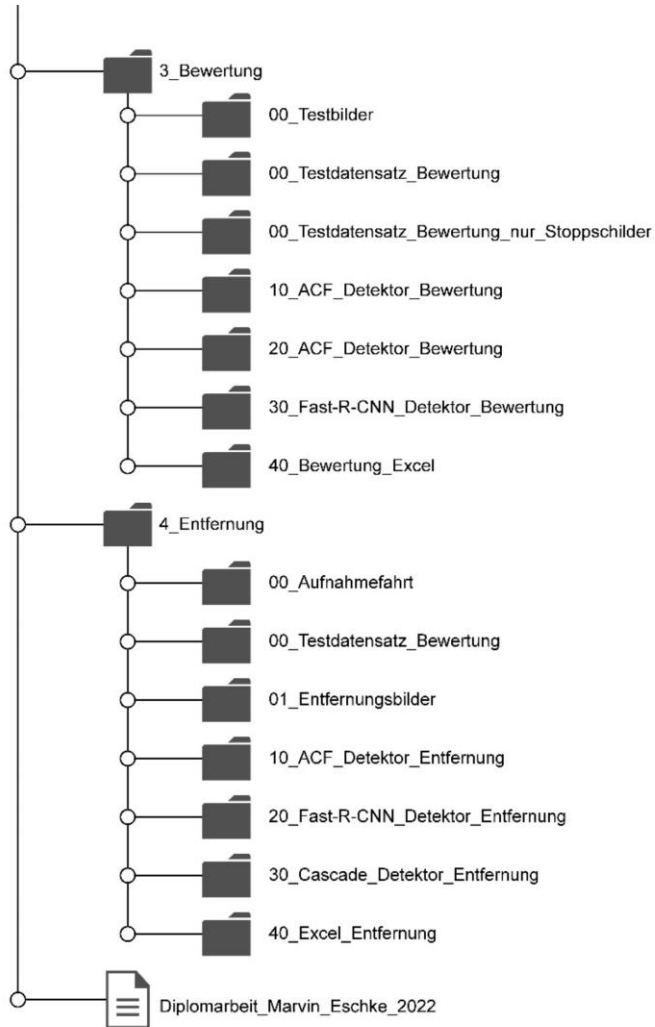
Abstand	287 Trainingsbilder	205 Trainingsbilder	123 Trainingsbilder	41 Trainingsbilder
11,83	0,00	1,00	1,00	1,00
19,67	1,00	1,00	0,00	0,00
28,30	1,00	1,00	1,00	0,00
38,33	0,00	0,00	1,00	1,00
48,87	0,00	0,00	0,00	0,00
59,20	0,00	0,00	0,00	0,00
70,28	0,00	0,00	0,00	0,00
79,95	0,00	0,00	0,00	0,00
83,76	0,00	0,00	0,00	0,00

Anlage 9



...Fortsetzung auf nächster Seite

...Fortsetzung Datenträger



Hinweis:

☆ In Absprach mit Herrn Professor Dr. rer. nat. Trautmann sind die mit "Stern" gekennzeichneten Ordner auf einem zusätzlichen Datenträger für diese Diplomarbeit übergeben wurden.

