Konzeptionierung einer Objekterkennungs-Bibliothek für Multi-Sensor-Systeme für einen Einsatz in der Simulation und im Testbetrieb autonomer Fahrzeuge

Masterarbeit

im Studiengang Kognitionswissenschaft

Vorgelegt von:	Tanja Koch
Matrikelnummer:	4170415
Betreuer:	Dipl. Ing. Thomas Schu
Prüfer:	Erstprüfer: PD Dr. Gregor Hardieß
	Zweitprüfer: Prof. Dr. Oliver Bringmann

Stuttgart-Vaihingen, den 26. November 2019

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich diese schriftliche Abschlussarbeit selbstständig verfasst habe, keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe und wörtliche oder sinngemäße, aus anderen Werken übernommenen Aussagen, als solche gekennzeichnet habe.

Stuttgart-Vaihingen, den 26. November 2019 Tanja Koch

Kurzfassung

In der folgenden Masterarbeit wird eine Objekterkennungs-Bibliothek entworfen und verschiedene Sensoren darin integriert. Es erfolgt hierbei eine Implementierung von einzelnen Sensoren in das entworfene System. Das Einsatzgebiet bezieht sich auf autonome Fahrzeuge und Simulationen.

Durch die Integration verschiedener Sensoren in einer Bibliothek können die Vor- und Nachteile der Sensoren direkt miteinander verglichen, sowie passende Sensorfusionen herausgefunden und aktiv eingebunden werden. Wichtig dabei ist, den möglichen Einsatz der einzelnen Sensoren zu betrachten. Es ist relevant, welche Funktionen Sensor-spezifisch sind, welche Überschneidungen zwischen den Sensoren auftreten und somit unter Umständen in der gleichen Funktion weiterverarbeitet und ausgewertet werden können. Hierbei spielt es auch eine Rolle, welche Daten von den einzelnen Sensoren ausgegeben werden und wie viel Speicher das System benötigt.

Die Auswahl der Sensoren, welche in das System mitaufgenommen werden, ist ein weiterer Teil der Arbeit. Diese Auswahl ist erweiterbar, sodass auch andere Sensoren nachträglich, unter Betrachtung ihre Funktion, hinzugefügt werden können. Die Bibliothek soll in eine Rechen-Simulation eingebunden werden. Diese ermöglicht Szenarien eines Fahrzeugs und die darin integrierten Sensoren und Assistenzsysteme zu testen. Solches Szenarien wären zum Beispiel Überholmanöver und die Hinderniserkennung. Hierdurch kann schon vor der Anschaffung und dem Einbau der Sensoren, die Auswahl der richtigen Sensoren geprüft und Zeit und Geld eingespart werden.

Inhaltsverzeichnis

1 Einleitung

Als Pferdekutschen durch Automobile ersetzt wurden, verschwanden Fähigkeiten wie Hindernisvermeidung und auch 'autonome Fahrten'. Bei diesen Fahrten brachten die Pferde ihren Besitzer trotz Fahruntauglichkeit nach Hause. Alternativ wurde von den Pferden auch manchmal ein 'sicherer Zustand' hergestellt. Dies geschah zum Beispiel, indem sie den Wagen ins Gras neben der Straße gezogen hatten.[1, S. 2–3]

Autonomes Fahren spielt heute eine große Rolle und wird auch auf verschiedenen Ebenen, beispielsweise in den Medien, diskutiert. Die Präsenz wird von Automobilherstellern, Systempartnern und Unternehmen durch Erfolgsmeldungen weiter unterstützt. Projekte wie 'Autonomes Fahren-Villa Ladenburg' sollen zu weiteren Diskussionen anregen.[1, S. 1]

In diesem Projekt von der Daimler und Benz Stiftung, arbeiten Wissenschaftler mit unterschiedlichen Fachgebieten daran, Fragen rund um das autonome Fahren zu beantworten. Es geht hierbei zum Beispiel darum, wie sich das Fahrverhalten durch autonomes Fahren verändert. Des Weiteren wird die Frage gestellt, wann Personen autonome Fahrzeuge nutzen würden und wann sie sich dabei sicher fühlen. Interessant ist zudem die Frage, ob das Versprechen zur Entlastung der Straßen eingehalten werden kann. Können somit vielleicht auch die Städte dank automatischer Parkassistenten attraktiver gemacht werden?[2]

Es stellt sich natürlich auch die Frage, warum das autonome Fahren so eine wichtige Rolle spielt. Ein wichtiger Punkt ist hierbei die Unfallvermeidung. Im Jahre 2010 gab es weltweit laut WHO 1,24 Millionen Verkehrstote.[1, S. 4] 90 Prozent aller Verkehrsunfälle geschahen hierbei auf Grundlage von menschlichen Fehlern, laut Allianz-Vorstand Alexander Vollert.[2]

Auch im Fernverkehr bieten autonome Fahrzeuge Vorteile. So können Stausituationen frühzeitig erkannt und umfahren werden. Zusätzlich kann der Sicherheitsabstand zwischen den Fahrzeugen kleiner sein. Dadurch passen mehr Autos auf die Straße.[2]

Durch autonome Fahrten sollen die Straßen vor allem in den Innenstädten entlastet werden. Dabei zeigt sich oft das Problem, dass es zu wenige Parkplätze gibt und dass durch den Park-Such-Verkehr unnötige Strecken gefahren werden. Hier können die Autos entweder selbständig leere Parksilos anfahren oder wie Taxis gleich den nächsten Nutzer abholen. Wichtig ist dabei, dass die Personen aber durch die Nutzung dieser neuen Möglichkeiten nicht in ihrer Mobilität eingeschränkt werden. Ein Beispiel für die mögliche Entlastung des Verkehrsaufkommens in Innenstädten zeigt sich in einer Simulation von Singapur. Hierbei könnten zwei Drittel der Autos eingespart werden. In einem weiteren Beispiel könnte die Stadt New York City durch diese optimale Routen- und Fahrtenplanung 30 Prozent der Taxis einsparen. Dies ist ein sehr wichtiger Punkt in der Betrachtung von autonomen Fahrzeugen, da schon heute viele Menschen stundenlang in Staus und die Städte oft vor einem Verkehrskollaps stehen. Zudem wird damit gerechnet, dass 60 Prozent der Menschen bis 2030 in Städten leben werden.[2]

Es gibt viele Gründe, die für autonomes Fahren sprechen. In einer Umfrage vom Institut für Verkehrsforschung am Deutschen Zentrum für Luft- und Raumfahrt wurde hierzu im Juni 2014 eine repräsentative Online-Umfrage durchgeführt. Hierbei wurden die Privatpersonen explizit zu vier Möglichkeiten zur Nutzung des autonomen Fahrens befragt. Darin enthalten ist der Autobahnpilot, das Valet-Parken, das vollautomatisierte Fahrzeug und das Vehicle-on-Demand. Es zeigte sich, dass die Unterscheidung wichtig ist, da verschiedene Personen verschiedene Ideen bezüglich der Nutzung des autonomen Fahrens haben.[2]

Die wenigstens Personen konnten sich eine Fortbewegung ohne eigenes Fahrzeug vorstellen (Vehicle-on-Demand). Diese Möglichkeit bietet jedoch viele Vorteile. Zum Beispiel für Personen die sich kein eigenes Fahrzeug leisten können oder keinen Führerschein besitzen, sowie Kinder, Alte und Personen die aus gesundheitlichen Gründen nicht fahren können. Gründe für die negative Einstellung zu diesem Thema kann sein, dass sich viele der Befragten dieses Zukunftsszenario nicht vorstellen können.[2]

Der Autopilot hat in der Umfrage die größte Zustimmung. Zum Beispiel bei langen Überlandfahrten kann der Fahrer die Fahraufgabe an das System abgeben. Dieses Szenario ist wie das Valet-Parken, sehr gut vorstellbar und die Vorteile sind für jeden ersichtlich. Beim selbständigen Einparken des Fahrzeuges (Valet-Parken) kann der Fahrer einfach aussteigen wo er möchte und das Auto sucht sich selbstständig einen Parkplatz.[2]

Die Probanden wurden danach gefragt, welchen Vorteil sie beim vollautomatisierten Fahren sehen. Hierbei ist interessant zu sehen, dass nur die Wenigsten also ein Viertel, im Auto arbeiten wollen. Auch die Nutzung vom Internet und Dingen, wie Filme anschauen, entspannen und schlafen wurden nur selten genannt. Hingegen wollen 70 Prozent der Personen die Landschaft betrachten und fast genau so viele sind daran interessiert sich mit anderen während der Fahrt zu unterhalten.[2]

Jedoch gibt es nicht nur positive sondern auch negative Meldungen. Im US-Bundesstaat Kalifornien wurde hierzu eine Statistik aufgestellt. Dort sind 658 autonome Fahrzeuge unterwegs. Zum Jahresbeginn (2018) waren es nur halb so viele. Es gab 67 Kollisionen in den ersten elf Monaten des Jahres 2018. Jedoch sind die autonomen Fahrzeuge bei den meisten Unfällen unschuldig. Allein 72 Prozent der registrierten Unfälle seit 2016 waren Auffahrunfälle in denen der Fahrer im nachfolgenden Verkehr nicht aufmerksam war. Verbraucherschützer sehen das autonome Fahren trotzdem kritisch. Dies liegt daran, dass sich diese Fahrzeuge anders verhalten als Personen, was zu Fehlinterpretationen führen kann.[3]

Zudem gibt es noch viele Fragen auf die Antworten gefunden werden müssen, darunter fallen Themen wie die Sicherheit, Datenschutz, Haftung und die Frage der Freigabe von autonomen Fahrzeugen. Es zeigt sich, dass Menschen schnell verschiedene visuelle Wahrnehmungen ihrer Bedeutung zuordnen können. Dies ist für die maschinelle Wahrnehmung heute noch eine schwierige Aufgabe.[2]

Man erkennt wie wichtig das Thema 'autonomes Fahren' und die visuelle Wahrnehmung durch verschiedene Sensoren ist. In der folgenden Arbeit soll nun das Thema weiter ausgebaut werden. Hierzu wird eine Objekterkennungs-Bibliothek entwickelt. Auf deren Grundlage können verschiedene Sensoren miteinander verglichen und getestet werden. Des Weiteren kann so getestet werden, welche Sensordaten optimal zusammengefasst werden können. Hierbei soll eine exemplarische Implementierung der Bibliothek stattfinden, auf deren Basis das System weiter ausgebaut und ergänzt werden kann. Die Bibliothek soll später in einer Simulation helfen die richtigen Sensoren schon vor der Anschaffung auswählen zu können.

Die Arbeit ist in verschiedene Bereiche aufgeteilt. Im Kapitel 2 wird zunächst auf die Grundlagen eingegangen. Dabei wird gezeigt, welche Sensoren ausgewählt wurden. Die Sensoren werden dann im Einzelnen kurz erklärt und es wird auf ihre Vor- und Nachteile eingegangen. Zusätzlich wird erklärt, wo sie eingesetzt werden und welche Informationen die Sensoren bereitstellen können. Des Weiteren wird auf die Fusion der Sensoren eingegangen und durch welche Möglichkeiten die Objektklassifizierung stattfinden kann.

Die Konzeptionierung der Objekterkennungs-Bibliothek wird in Kapitel 3 thematisiert. Hier wird der Programmablauf und das Konzept der Bibliothek erklärt. Im nächsten Kapitel wird die Entwicklung der Bibliothek beschrieben. Hierbei wird auf den Aufbau und die Funktionsweise der Bibliothek eingegangen. Des Weiteren wird gezeigt wie die Fusion der Sensorwerte stattfindet. Weiterführend wird auf die graphische Benutzeroberfläche, welche als Ergänzung zur Bibliothek programmiert wurde, eingegangen. Um während der Programmierung einzelne Teilprogrammierungen zu testen, wurden zusätzlich Testdaten geschrieben. Es wird gezeigt, wie die Messdaten eingebunden werden können und die benutzerfreundliche Auslegung der Bibliothek aufgezeigt.

Als nächstes wird im Kapitel 5 auf die Evaluation eingegangen. Darin wird erklärt, welche Sensoren für das Testen der Bibliothek ausgewählt wurden. Die Bibliothek wurde in einem Versuchsaufbau getestet und die Ergebnisse evaluiert. Es wird dargestellt, welche Ergebnisse die GUI im Versuchsaufbau ausgibt und wie dieser Aufbau und die Ergebnisse zu bewerten sind.

Am Ende wird die Arbeit zusammengefasst und reflektiert. Zudem gibt es einen Ausblick, wie das weitere Vorgehen sein kann und welches Verbesserungspotenzial es noch gibt.

2 Grundlagen

Viele verschiedene Technologien werden schon heute im Bereich autonomes Fahren und Fahrerassistenzsysteme eingesetzt. Mit Hilfe von diesen sollen Autos ihre Umwelt erfassen und auf Veränderungen reagieren können. Dabei ist es wichtig, sowohl die Vorteile als auch die Nachteile der einzelnen Systeme zu beachten.[4]

Die möglichen Einsatzgebiete der Sensoren spielen bei der Betrachtung eine Rolle. Zudem muss man die Auswertung der Sensordaten mit einbeziehen.

2.1 Sensor-Auswahl

Es wurden sechs Systeme ausgewählt, welche sich in Ihren Vor-, Nachteilen und Einsatzgebieten ergänzen: Radio Detection and Ranging (Radar)[5], Ultraschall, Light Detection and Ranging (Lidar)[5], Laserscanner, Kamerasystem und Infrarotkamera.

Die ausgewählten Sensoren sind der Kategorie Umfeldsensorik untergeordnet. In der Abbildung 2.1 wird die Unterteilung der Umfeldsensorik anhand des Funktionsprinzips in die einzelnen Sensorarten dargestellt.[5]

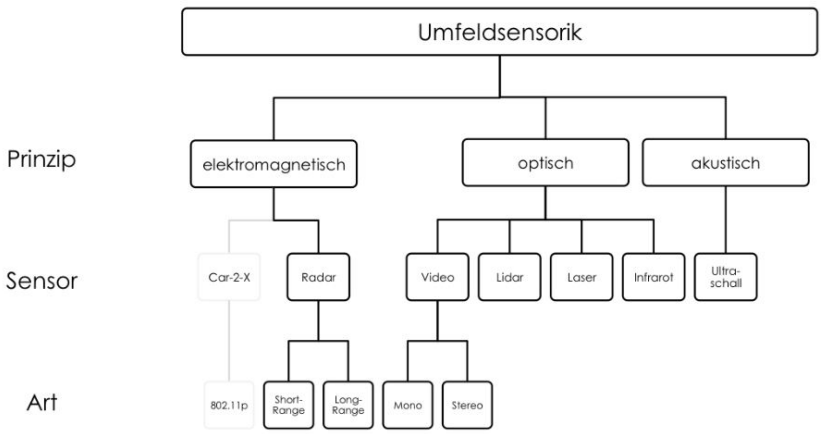


Abbildung 2.1: Übersicht Umfeldsensorik [5]

Durch die Auswahl der Sensoren sind alle drei Prinzipien abgedeckt. Auf elektromagnetischer Ebene durch den Radarsensor, auf der Optischen das Videosystem, Lidar, Laser und die Infrarotkamera und im akustischen Bereich mit dem Ultraschallsensor.[5]

2.1.1 Radar

Bei Radarsensoren werden durch einen Funksender und -empfänger Radiowellen emittiert und empfangen. Radarsensoren werden in unterschiedlichen Bereichen im autonomen Fahren eingesetzt. Sie dienen als Unterstützung zur adaptiven Geschwindigkeitsregelung, Abstandshaltung, Hinderniserkennung und werden bei Park- und Spurhalteassistenten eingesetzt.[5][6, S. 207]

Das System wird in drei Stufen unterteilt: die Nahbereichserkennung ($<$ 30 m), die Mittelbereichserkennung ($<$ 100 m) und die Fernbereichserkennung ($<$ 200 m).[6, S. 207] Das Long-Range Radar (typisch: 77 GHz) wird hierbei beispielsweise in der adaptiven Geschwindigkeitsregelung eingesetzt. Das Short-Range Radar wird (typischerweise 24 GHz) normalerweise im Nahbereich eingesetzt, also zum Beispiel für das Ein- und Ausparken.[5]

Das Radar hat viele Vorteile. Ein wichtiger Vorteil ist hierbei die Allwettertauglichkeit des Sensors. Das heißt, dass Wettereinflüsse nur eine geringe Auswirkung auf den Sensor haben. Zudem braucht er kein Tageslicht und funktioniert somit auch in der Dunkelheit. Er zeichnet sich durch eine kleine und kompakte Bauweise aus und ist sehr robust im Bezug auf Erschütterungen.[7][8, S. 20]

Das Radar weißt jedoch auch Nachteile auf. So werden nur Punktinformationen und eine grobe Auflösung geliefert. Zudem können weitere Radarsensoren störend auf das Signal wirken und bei Messungen in vertikaler Achse zeigen sich Ungenauigkeiten. Das heißt, dass es keinen Ausgleich von Fahrzeugneigungen gibt. Weitere Probleme zeigen sich bei der Objekterkennung. Zum Einen können Phantomobjekte erkannt werden, zum anderen brauchen die Objekte aber eine gewisse Größe um überhaupt erkannt zu werden. Ein anderes Problem zeigt sich bei der Reflektierung. So haben zum Beispiel Radfahrer und Fußgänger nur sehr geringe Reflexionseigenschaften. Dies kann dazu führen, dass diese gar nicht erkannt werden. [5][6, S. 207][7]

Mit dem Doppler-Effekt können zwei Messgrößen ermittelt werden. Zum Einen der Abstand und zum Zweiten die relative Geschwindigkeit. Um zu erkennen, wo sich im Radar das Objekt befindet, müssen mehrere Antennen halbmondförmig angeordnet werden, siehe Abbildung 2.2 auf der nächsten Seite. Da die Radarkeulen relativ breit sind, stellt sich die Winkelauflösung als eher schlecht heraus.[5]

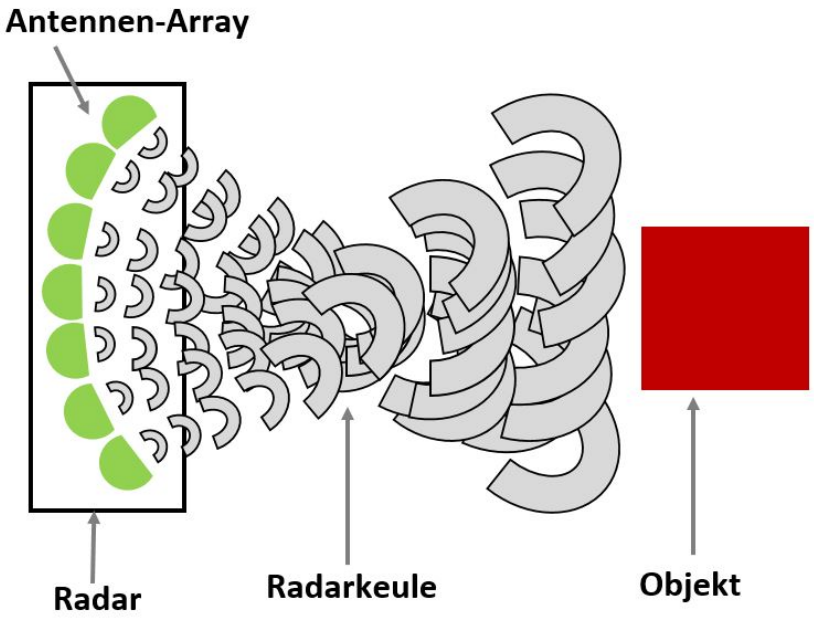


Abbildung 2.2: Aufbau Radar [5]

Es wird zwischen einem akustischen und einem relativistischen Dopplereffekt unterschieden. Der Unterschied liegt hierbei in der Ausbreitung. Elektromagnetische Wellen (relativistischen Dopplereffekt) breiten sich im Gegensatz zu mediengebundenen Wellen (akustischer Dopplereffekt) ohne Übertragungsmedium aus. Für die Radartechnik ist hierbei der relativistische Dopplereffekt von Bedeutung.[9, S. 12–13]

$$f' = f_0 * \sqrt{\frac{c_0 - v}{c_0 + v}} \tag{2.1}$$

Mit der Formel 2.1 wird der allgemeine relativistische Dopplereffekt berechnet. Hierbei steht f' für die resultierende Frequenz,. f_0 beschreibt die Sendefrequenz, c_0 die Lichtgeschwindigkeit und mit v wird die Relativgeschwindigkeit beschrieben.[9, S. 12–13]

2.1.2 Ultraschall

Ultraschallsensoren messen den Zeitraum vom Versenden der Ultraschallwelle bis zum Zurückkommen des reflektierten Schalls. In der Abbildung 2.3 wird die Funktionsweise aufgezeigt. Ultraschallsensoren sind für den Nahbereich geeignet ($<$ 3 m) und werden beispielsweise bei Parkassistenten, der Bestimmung von Fahrzeugen auf der Nebenspur und bei Überholassistenten eingesetzt.[5][6, S. 207]

Vorteile von Ultraschallsensoren ergeben sich im Preis und in der kompakten Bauweise. Zudem erkennen sie viele Objekte unabhängig des Materials und der Farben. Nur schalldämpfende Stoffe (zum Beispiel Watte) oder schräge Flächen können zum Problem werden.[8, S. 20][9, S. 21][10]

Die Nachteile zeigen sich beispielsweise in der geringen Reichweite, der langsamen Messung und ihrer Störanfälligkeit. Wie bei Radarsensoren können Sensoren des gleichen Typs zu Störeinflüssen führen. Zudem brauchen sie Kontakt zum Medium Luft und sind Wetterabhängig. So können sie bei Regen und Schnee Probleme in der Funktion aufweisen.[6, S. 207–208][8, S. 20]

Der Ultraschallsensor dient zur Ermittlung von Entfernungsinformationen. Informationen zur Geschwindigkeit werden hierbei, anders als beim Radarsensor, nicht ermittelt.[8, S. 20]

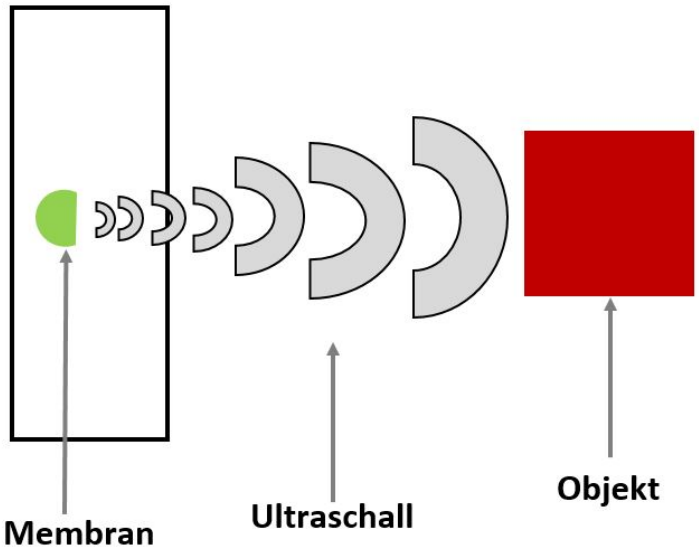


Abbildung 2.3: Aufbau Ultraschall [5]

2.1.3 Lidar

Lidar erzeugt eine 3D-Punktewolke indem Lichtwellen in Form eines Laserstrahls ausgesendet werden. Es können so, anders als beim Radar, Konturen der Objekte erkannt werden. Die Abbildung 2.4 zeigt das Funktionsprinzip von Lidar. Zwischen Emittieren und Empfangen der Lichtwelle wird die Zeit gemessen.[6, S. 207][7]

Die Entfernung wird mit Hilfe von der Laufzeit und Lichtgeschwindigkeit ermittelt. Es ist eine Bestimmung des Objekttyps und der Objektgröße möglich. Anders als beim Radar wird die Geschwindigkeit über nummerische Differentiation zweier Positionsmessungen ermittelt.[5]

Der Lidarsensor bietet viele Vorteile. So ist der Sensor unabhängig von Tageslicht und Störeinflüsse von Sensoren des gleichen Typs. Er weist eine hohe Messauflösung und einen hohen Erfassungsbereich im Bezug auf Entfernungen und Winkeln auf. Im Gegesatz zum Radarsensor ist ein Ausgleich von Fahrzeugneigungen möglich.[6, S. 207][7]

Als Nachteile zeigen sich die Anfälligkeit gegenüber Witterungsbedingungen, wie beispielsweise Nebel oder in Wassernähe die Gischt, und der Preis.[6, S. 207][9, S. 17–18]

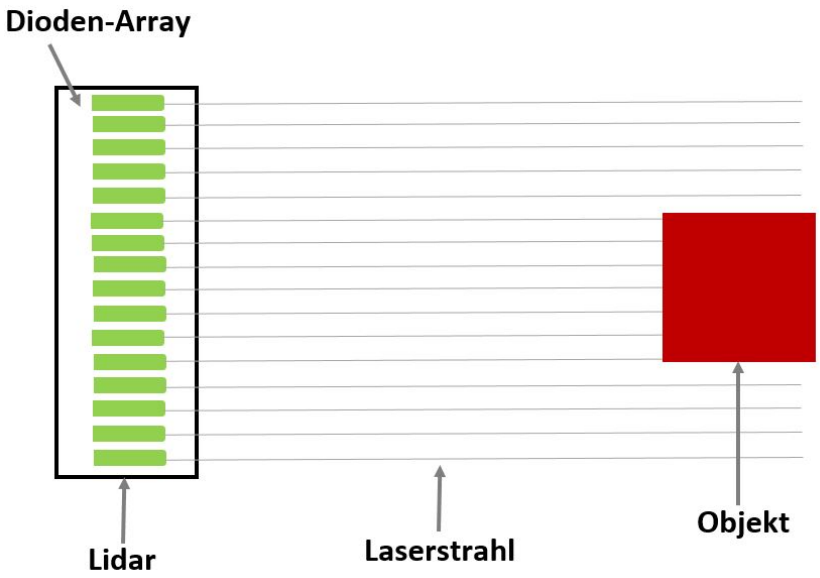


Abbildung 2.4: Aufbau Lidar [5]

2.1.4 Laserscanner

Der Laserscanner ist eine Weiterentwicklung zum Lidarsensor. Dabei werden ebenfalls Lichtimpulse versendet und die zurückgeworfenen Strahlen über ein Lichtlaufzeitverfahren ausgewertet. Jedoch werden, anders als bei Lidar, die Laserstrahlen mit Hilfe von einem drehenden Spiegel abgelenkt und über den zu scannenden Bereich verteilt, siehe Abbildung 2.5. Dabei entsteht eine Umgebungskarte.[6, S. 207][7]

Die Vorteile sind hierbei die Gleichen, wie beim Lidarsensor. Der größte Vorteil zeigt sich in der sehr guten Auflösung bei Entfernung und Winkel.[5] Die Umwelt kann in 3D mit einer bis zu 360 Grad Abdeckung erfasst werden.[6, S. 207]

Der Laserscanner zeigt auch die gleichen Nachteile, wie der Lidarsensor. Ein zusätzliches Problem zeigt sich jedoch bei der großen Bauform und den mechanisch beweglichen Teilen.[7]

Wie beim Lidarsensor beschrieben, erfolgt die Bestimmung der Entfernung über die Laufzeit und Lichtgeschwindigkeit. Die Geschwindigkeit wird, wie schon erwähnt, über nummerische Differentiation zweier Positionsmessungen ermittelt. Eine Bestimmung des Objekttyps und der Objektgröße ist möglich.[5]

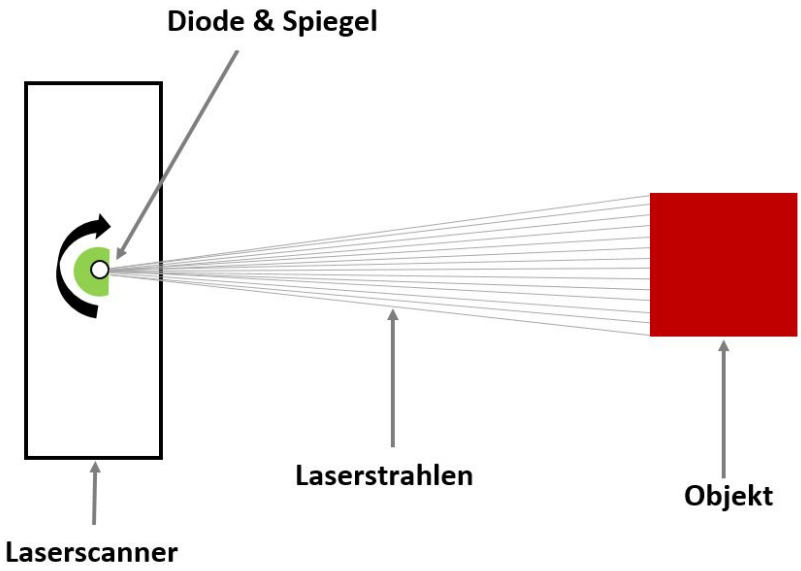
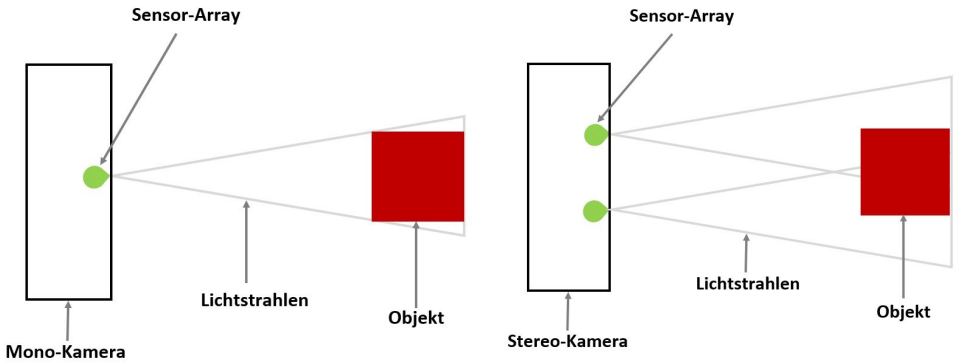


Abbildung 2.5: Aufbau Laserscanner [5]

2.1.5 Kamerasystem

Das Kamerasystem arbeitet auf Grundlage von Graustufenbilder. Diese werden über eine Videokamera erfasst und weiterverarbeitet. Mit Hilfe von Plausibilisierungsalgorithmen und Filtern werden Umfeldinformationen gewonnen.[5]

Es werden zwei Systeme unterschieden: einmal das Arbeiten mit einem Mono-Kamerasystem oder mit einem Stereo-Kamerasystem. Der Unterschied liegt im Aufbau und in den Informationen. So besteht ein Mono-Kamerasystem aus nur einer Kamera, welche Videos mit zweidimensionalen Darstellungen liefert. Beim Stereo-Kamerasystem kann hingegen eine dritte Dimension berechnet werden. Somit sind auch Informationen zu Abständen und Geschwindigkeiten ermittelbar.[5] In der Abbildung 2.6 wird der Aufbau der beiden Systeme dargestellt.



(a) Mono-Kamerasystem (b) Stereo-Kamerasystem

Abbildung 2.6: Vergleich Aufbau Mono-/Stereo-Kamerasystem [5]

Die Vorteile zeigen sich in einer hohen Auflösung und Reichweite. Leider zeigt das Kamerasystem auch Nachteile. Es ist wetterabhängig, also zum Beispiel bei Nebel nicht einsetzbar. Zudem müssen die Lichtverhältnisse passen. So kann eine tiefstehende Sonne zu Problemen führen.[8, S. 20][11, S. 9]

Das Kamerasystem bietet die Möglichkeit einer Objekterkennung und Objektkategorisierung. Über Programme, welche mit neuronalen Netzen arbeiten, besteht die Möglichkeit, Objekte zu definieren und in die Kategorien wie zum Beispiel Menschen oder Autos, einfach und schnell einzuteilen. Zudem können neue Kategorien dazu gelernt und alte ständig erweitert werden. Ein Beispiel hierfür ist das Arbeiten mit der Bibliothek OpenCV und dem neuronalen Netzwerk You Only Look Once (Yolo).[12]

Des Weiteren können indirekt über Bildfolgen Informationen zu Geschwindigkeiten und Distanzen ermittelt werden.[8, S. 20]

2.1.6 Infrarotkamera

Bei einer Infrarotkamera wird Licht im Infrarotspektrum ausgesandt und im Infrarotbereich von sensitiven Kameras aufgefangen und ausgewertet (siehe Abbildung 2.7 auf der nächsten Seite). Die Kamera wird zum Beispiel bei Nachtsichtassisten eingesetzt.[13, S. 66]

Hierbei wird in einen Nah-, Mittel- und Fern-Infrarot Bereich unterteilt. Der Unterschied besteht hierbei in der Wellenzahl des Lichts. Im Nahbereich beträgt die Wellenzahl des Lichts zwischen 10^6 cm^{-1} und $10^4 cm^{-1}$, im Mittelbereich zwischen 10^4 cm^{-1} und 100 cm^{-1} und im Fernbereich zwischen 10 cm^{-1} und 1 cm^{-1}.[14, S. 21]

Eine weitere Unterscheidung besteht bezüglich dem aktiven und passiven Infrarot-System. Beim passiven System werden keine Infrarotstrahlen ausgesandt, sondern nur empfangen. Hierbei werden die zeitlichen Veränderungen von integralen Wärmebildern detektiert. Im Gegensatz dazu wird beim aktiven System Licht im Infrarotspektrum nicht nur empfangen, sondern auch aktiv ausgesendet. Beide Systeme haben sowohl Vor- als auch Nachteile. Somit können zum Beispiel passive Infrarot-Systeme keine Informationen zu Geschwindigkeiten und Entfernung liefern. Im Gegensatz dazu sind aktive Infrarot-Systeme empfindlich gegenüber Umwelteinflüsse.[8, S. 20]

Infrarot-Kameras bieten viele Vorteile und werden schon heute eingesetzt. So können auch bei Nacht, Nebel, Schnee, Regen oder bei tiefstehender Sonne Menschen und Tiere erkannt werden. Bei Nacht strahlen die Scheinwerfer des Autos Infrarotwellen aus. Die Umgebung wird durch Fern-Infrarotkameramodule beobachtet. Bei Nachtsichtassisten werden die Bilder auf einem Bildschirm angezeigt und somit das Sichtfeld um den Faktor drei erweitert, ohne dass der Gegenverkehr geblendet wird.[5][13, S. 66]

Als Nachteile zeigt sich bei Infrarotkameras (Beispiel: Viper) die fehlenden Informationen zu Farben, Abständen und Geschwindigkeiten.[11, S. 9]

Die Weiterverarbeitung der Daten erfolgt ähnlich wie bei der Videosensorik und dient zur Unterstützung bei der Objekterkennung und Kategorisierung.[5]

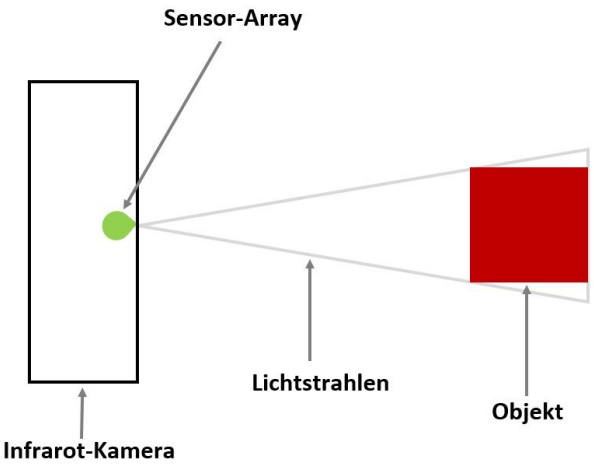


Abbildung 2.7: Aufbau Infrarot-Kamera [5]

2.2 Position und Dimension

Auch die Positionsbestimmung wird in der Bibliothek berücksichtigt. Hierbei geht es darum mitzuteilen, wo sich ein Objekt im Sichtfenster der Sensoren befindet. Durch die Informationen können zudem Bewegungsabläufe vorhergesagt werden.

Die Dimension bezieht sich hingegen auf die Größe des Objektes. Diese Information kann zum Beispiel dabei helfen zu bestimmen, ob das Objekt eine Gefahr darstellt. Des Weiteren kann die Information bei der Objektkategorisierung mit einbezogen werden.

2.3 Sensorfusion

Die Aufgabe der maschinellen Wahrnehmung ist es andere Verkehrsteilnehmer und Verkehrssituationen zu erkennen und richtig einzuordnen, sobald dies für die autonome Fahrt wichtig ist. Ein Beispiel hierfür sind Fußgänger, welche den Gehweg benutzen. Sie stellen eine andere Gefahr dar, wenn sie sich direkt am Straßenrand bewegen.[1, S. 422]

Um dies zu erreichen werden verschiedene Sensoren verwendet. Das Zusammenführen der Informationen von verschiedenen Sensoren wird als Sensorfusion bezeichnet. Diese Fusion ist aus unterschiedlichen Gründen wichtig.[15]

Ein Punkt ist hierbei, dass durch die Fusionierung sowohl bewegte als auch statische Objekte erfasst werden können. Dies zeigt sich beispielsweise beim Erkennen und Vermessen von Fahrbahnmarkierungen.[1, S. 422]

Jeder Sensor weißt spezifische Vor- und Nachteile auf. Zum Beispiel bietet ein lidarbasierter Sensor eine hohe Messauflösung, allerdings bestehen Einschränkungen bei Nebel. Im Gegensatz hierzu zeigt das Radar zwar eine schlechte Auflösung, aber Wettereinflüsse haben nur einen geringen Einfluss auf den Sensor.[15]

Des Weiteren zeigt sich die Wichtigkeit der Sensorfusion beim Betrachten der zu ermittelten Distanz. So kann mit einem Stereo-Kamerasystem die Distanz zu Objekten ermittelt werden. Jedoch nimmt der Messfehler dieser Messung quadratisch mit der Entfernung zu. Bei modernen Kamerasystemen liegt die Messreichweite bei ca. 50 m. Im Gegensatz dazu bieten Radar- und Lidarsensoren genaue Informationen zur Distanzmessung mit nahezu gleichbleibender Genauigkeit. Auf Grundlage schlechter Winkelauflösung können bei beiden Sensorarten die Maße der Objekte nur schlecht erfassen. Es zeigt sich, dass eine Sensordatenfusion für die maschinelle Wahrnehmung ein wichtiger Punkt ist.[1, S. 422]

Des Weiteren haben die Sensoren jeweils andere Anwendungsgebiete und liefern unterschiedliche Ausgaben. So kann aus den Rohdaten eines Radarsensors Informationen zur Geschwindigkeit und Distanz gewonnen werden. Eine passive Infrarotkamera hingegen wird bei der Objektbestimmung eingesetzt.

2.4 Objektklassifizierung mit Hilfe neuronaler Netzwerke

Neuronale Netze gelten als informationsverarbeitende Systeme. Diese besitzen eine grobe Analogie zum Gehirn von Säugetieren. Im Gehirn findet die Informationsverarbeitung über viele Nervenzellen statt. Diese Nervenzellen sind einfach und leiten über Nervenbahnen den Grad ihrer Erregung weiter. Neuronale Netze bestehen auch aus einfachen Einheiten, wie Zellen und Neuronen. Die Informationen werden über gerichtete Verbindungen und indem Zellen aktiviert werden übermittelt.[16, S. 23]

Ein wichtiger Punkt der künstlichen neuronalen Netze ist deren Lernfähigkeit. Das bedeutet eine Aufgabe kann selbstständig mit Trainingsspielen erlernt werden. Hierfür findet keine explizite Programmierung des neuronalen Netzes statt. Neuronale Netze spielen in vielen verschiedenen Gebieten eine wichtige Rolle. Hierzu zählen die Biologie, Medizin, Psychologie, Elektrotechnik, Psychologie, Mathematik und Informatik.[16, S. 23–24]

2.4.1 Grundlagen neuronaler Netze

Bei den neuronalen Netzen gibt es im Allgemeinen verschiedene Komponenten. Dazu gehören zum einen die Zellen (Neuronen, Elemente, units). Sie bestehen aus dem Aktivierungszustand (activation) ($a_i(t)$), welcher den aktuellen Zustand der Zelle angibt. Des Weiteren gehört die Aktivierungsfunktion (f_{act}) dazu. Mit dieser Funktion wird die neue Aktivierung aus den verschiedenen Parametern berechnet (sieht Formel 2.2).[16, S. 72]

$$a_j(t+1) = f_{\text{act}}(a_j(t), net_j(t), \Theta_j) \tag{2.2}$$

In der Formel 2.2 wird gezeigt, wie für das Neuron (j) der neue Aktivierungszustand ($a_j(t+1)$)aus der alten Aktivierung ($a_j(t)$) und der Netzeingabe (net input) ($net_j(t)$) mit Hilfe der Aktivierungsfunktion (f_{act}) berechnet wird. Θ_j steht hierbei für den Schwellenwert des Neurons.[16, S. 72]

Ein weiteres Element der Zelle ist die Ausgabefunktion (f_{out}). o_j steht hierbei für die Ausgabe der Zelle j. Diese wird mit Hilfe der Ausgabefunktion (f_{out}) und der Aktivierung der Zelle (a_j) berechnet. Die Berechnung wird in der Formel 2.3 dargestellt.[16, S. 72]

$$o_j = f_{\text{out}}(a_j) \tag{2.3}$$

Zwei weitere wichtige Punkte sind das Verbindungsnetzwerk der Zellen und die Propagierungsfunktion. Beim Verbindungsnetzwerk wird das neuronale Netz als gerichteter und gewichteter Graph betrachtet. Die Kanten stellen hierbei die gewichteten Verbindungen zwischen den Neuronen dar. w_{ij} beschreibt hierbei das Gewicht (weight) der Verbindung der Zellen i und j. Hierbei findet die Gewichtung von Zelle i nach Zelle j statt. Die Reihenfolge der Indizes muss hierbei beachtet werden. W steht für die Gewichtsmatrix, also für die Verbindung aller Zellen.[16, S. 72–73]

$$net_j(t) = \sum\nolimits_i o_i(t) w_{\mathrm{ij}} \tag{2.4}$$

Die Propagierungsfunktion zeigt, wie die Netzeingabe eines Neurons $net_j(t)$ mit Hilfe der Summe der Ausgaben der Vorgängerzellen $o_i(t)$ und den Verbindungsgewichten w_{ij} berechnet wird. Die Berechnung wird in der der Formel 2.4 gezeigt.[16, S. 72–73]

Auch die Lernregel ist eine Komponente der neuronalen Netze. Hierbei handelt es sich um einen Algorithmus, anhand dessen das neuronale Netz lernt, aus einer vorgegebenen Eingabe eine gewünschte Ausgaben zu entwickeln. Gelernt wird hierbei meistens, indem die Stärke der Verbindungen angepasst wird. Dies geschieht durch die wiederholte Präsentation von Trainingsmustern. Ziel ist es oft, dass die erwartete Ausgabe und die tatsächlichen Ausgabe für die Trainingsmuster so nah beieinander liegen wie nur möglich. Dies beutet, dass der Fehler minimiert wird.[16, S. 73]

2.4.2 feedforward-Netzwerk

Viele der Verbindungen von neuronalen Netzen gehen nur in eine Richtung. Das heißt, sie verlaufen vom Eingabeneuron (input units) zum Ausgabeneuron (output units). Dadurch können die Zellen durch die Positionierung im Netzwerk klassifiziert werden.[16, S. 73]

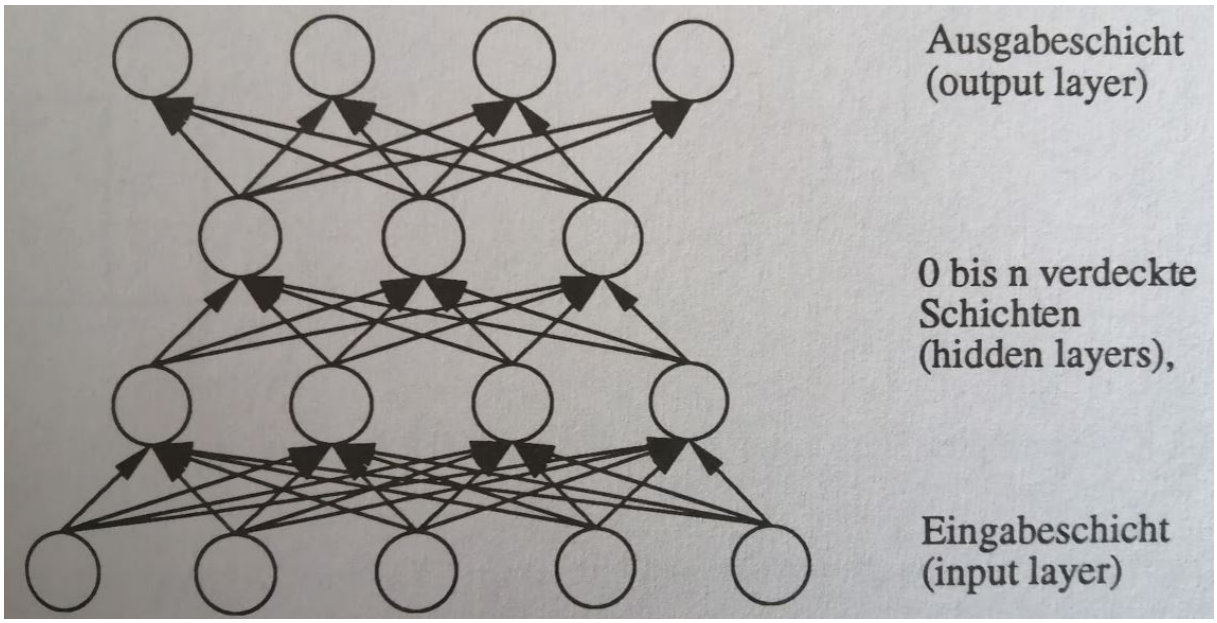


Abbildung 2.8: feedforward-Netzwerk [16, S. 73]

In der Abbildung 2.8 wird dieser häufig verwendete Spezialfall aufgezeigt. Das Netz besteht hierbei aus drei Stufen. Somit besitzt es drei Schichten mit trainierbaren Verbindungen. Die Eingabe wird von den Eingabeneuronen (input units) in das Netz weitergeleitet (Eingabeschicht). In der mittleren Schicht (verdeckte Schicht) findet die Informationsverarbeitung innerhalb des Netzes statt. Da diese Neuronen von außen nicht zu sehen sind, werden diese Neuronen als verdeckte Neuronen (hidden units) bezeichnet. In der letzten Schicht (Ausgabeschicht) wird von den Ausgabeneuronen (output units) die Ausgabe nach Außen getätigt.[16, S. 73–74]

2.4.3 computerbasierte Klassifizierung

Die Objektklassifizierung thematisiert die Objektunterscheidung und eindeutige Objektbezeichnung. Diese Unterscheidung und Bezeichnung fällt auch oft Menschen nicht leicht. Um diese Bestimmung von einer Maschine durchführen zu lassen gibt es verschiedene Ansätze.

Convolutional Neural Network

Einer der ersten Ansätze in der computerbasierten Klassifizierung ist das Convolutional Neural Network (CNN). Das CNN stellt eine Unterart des feedforward-Netzwerks dar. Es unterscheidet sich jedoch durch spezielle verborgene Ebenen, welche für unterschiedliche Aufgaben zuständig sind. Hierbei werden verschiedenen Methoden verwendet, um lokale Merkmale zu erkennen und zu kombinieren.[17]

Zu den Bestandteilen eines CNN gehören Filter (Convolutional Layer), Aggregations-Schichten(Pooling-Layer) und eine oder mehrere Standard-Schichten (Dense/Fully Connected Layer) mit vollständig verbundenen Neuronen. Die Filter und Aggregations-Schichten werden hierbei abwechselnd wiederholt.[18]

Im Convolutional Layer ist jedes Neuron für ein Areal der Eingangsdaten verantwortlich. Diese Bereiche werden Receptive Field genannt. Neuronen mit gleichen Gewichtungen werden hierbei jedoch auf alle Areale angewendet. Dies liegt daran, dass Merkmale nicht immer an den gleichen Stellen auftreten. Feature Maps sind die gesammelten Zustände von Neuronen mit den gleichen Gewichtungen. Es werden mehrere Feature Maps benötigt um einzelne Merkmale von Objekten zu verbinden.[17]

Das Convolutional Layer kann durch verschiedene Parameter verändert werden. Hierzu gehört zum Einen die Größe des Rezeptive Fields (F). Durch die Größe wird die Menge der Daten zugeteilt, welche jedes Neuron verarbeiten muss. Die Eingangsdaten sind hierbei als dreidimensional zu betrachten. Des Weiteren gehört die Schrittweite des Rezeptive Fields (S) dazu. Hierbei wird die Pixelanzahl der Feldverschiebung nach dem Wechseln zum nächsten Neuron gezählt. Ein weiterer Punkt ist die Tiefe des Layers. Hierbei wird die Anzahl der Merkmale vorgegeben, auf die die Eingangsdaten untersucht werden sollen. Des Weiteren gehört der Input Padding (P) dazu. Dabei werden die Eingangsdaten mit Nullwerten erweitert, wodurch das gewünschte Volumen erzeugt werden kann.[17]

Beim Pooling Layer werden schwache Signale aussortiert und nur die stärksten Signale weitergegeben. Durch das Pooling werden so nur die wichtigen Signale zur nächsten Schicht weitergegeben. Zudem soll eine abstrakte Repräsentation erreicht und die Parameter des Netzes reduziert werden.[18]

Das Dense/Fully Connected Layer besitzt eine normale neuronale Netz-Struktur. In dieser Struktur sind alle Inputs und Outputs mit den Neuronen verbunden. Die Output-Signale der Filter-Schichten besitzen keine Positionsmerkmale. Im Gegensatz dazu besitzen sie jedoch ortsunabhängige Objektinformationen. Diese Informationen werden dann in ein/ mehrere Fully Connected Layer übergeben und anschließend wird mit einem Output-Layer eine Verbindung hergestellt.[18]

Das Convolutional Neural Networks funktioniert somit, indem es mit Hilfe seiner Filter ortsunabhängige Strukturen in den Eingangsdaten erkennt. Hierbei führen in den ersten Ebenen einfache Strukturen wie Linien, Kanten und Farbtupfer zur Aktivierung der Filter. Ein wichtiger Punkt ist hierbei, dass das Netz selbständig die Art der Filter erlernt. Auf der nächsten Ebene werden dann kompliziertere Strukturen, wie Kombinationen aus den ersten Fällen, erlernt. Ein Beispiel hierfür wären Kurven. So wird das Abstraktions-Level immer weiter erhöht. Die Tiefe dieser Abstraktionen wird dabei über die charakteristischen Merkmalen der vorgegebenen Klassen bestimmt. In der Abbildung 2.9 wird so eine stetige Vertiefung der Abstraktionen gezeigt. Die Vertiefung wird von Layer 1 bis Layer 5 stetig abstrakter.[18]

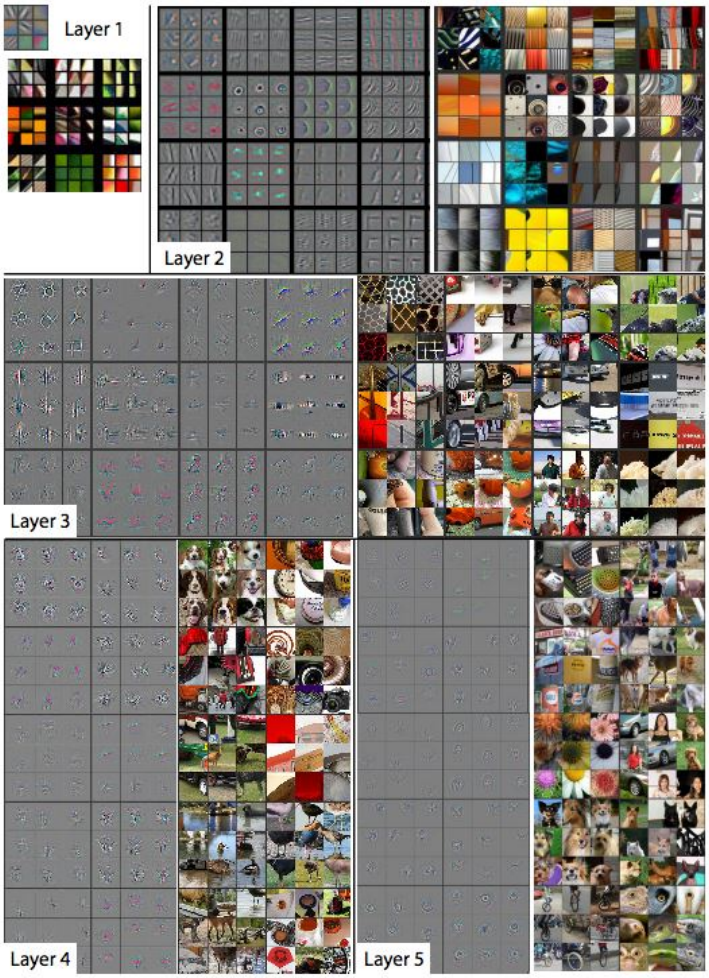


Abbildung 2.9: verschiedene Ebenen der Abstraktion beim CNN [18]

You Only Look Once

Eine Möglichkeit für eine schnelle und einfache Objektklassifizierung bietet sich durch Yolo. Wie der Name schon sagt, wird hier nur ein einziges CNN auf das Bild angewendet. Dabei wird das Bild in verschiedene Bereiche unterteilt. Es werden Begrenzungsrahmen und Wahrscheinlichkeiten für die einzelnen Bereiche vorausgesagt. Die Gewichtung der einzelnen Begrenzungsfelder findet über die vorhergesagten Wahrscheinlichkeiten statt. Die Vorteile zeigen sich in der Geschwindigkeit der Auswertung, da wie schon erwähnt eine einzelne Netzauswertung zur Vorhersage reicht. Zusätzlich kann es dadurch einfacher optimiert werden als mehrstufige Verfahren. Des Weiteren ist so auch eine Echtzeit-Objekterkennung möglich. In der Abbildung 2.10 wird die graphische Auswertung eines Bildes mit Hilfe von Yolo gezeigt. Hierbei werden die Objekte direkt erfasst und die Kategorisierungen aufgeführt.[18][19]

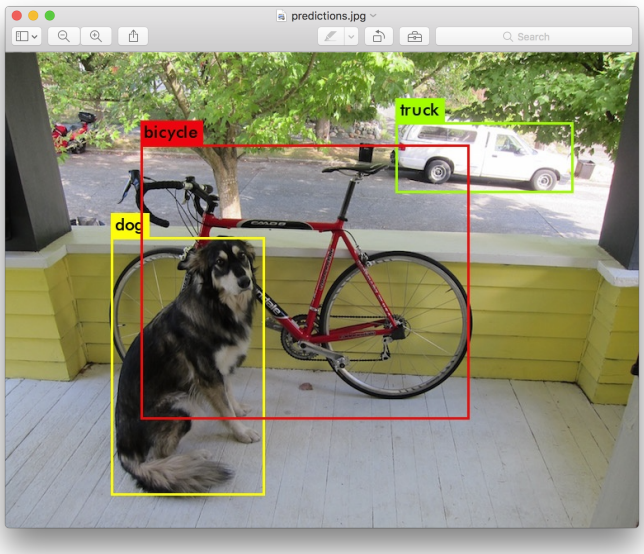


Abbildung 2.10: Beispielbild bei der Auswertung über Yolo [12]

3 Konzeptionierung der Objekterkennungs-Bibliothek

Die Programmbibliothek wird definiert als eine Sammlung verschiedener Programmteile, wie zum Beispiel Unterprogramme und Klassen. In der objektorientierten Programmierung wird die Programmbibliothek als Klassenbibliothek, also eine Sammlung verschiedener Klassen, bezeichnet. Diese können dann in verschiedene Programmcodes eingebunden und verwendet werden.[20, S. 6]

3.1 Konzept des Programmablaufs

Im ersten Schritt wurde ein Konzept zur Darstellung des Projektablaufs, unterteilt in einzelne Ablaufschritte, erstellt, siehe Abbildung 3.1.

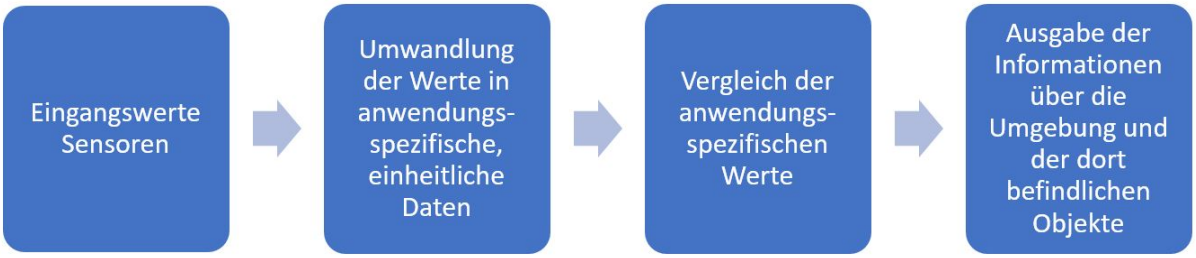


Abbildung 3.1: allgemeiner Ablaufplan

Zuerst liefern die Sensoren verschiedene Eingangswerte. Diese müssen dann in den jeweiligen Anwendungen vereinheitlicht werden. Danach können die einzelnen Daten verglichen und zu einem Ergebnis zusammengefasst werden. Im letzten Schritt sollen die Informationen über die erfasste Umgebung ausgegeben werden.

3.2 Konzept der Bibliothek

Die Abbildung 3.2 zeigt eine Übersicht der Informationen, welche zum Einen der Bibliothek übergeben und zum Anderen von ihr ausgegeben werden können. Die einzelnen Sensorenausgaben werden hierbei je nach Anwendungsgebiet unterschiedlich eingebunden.

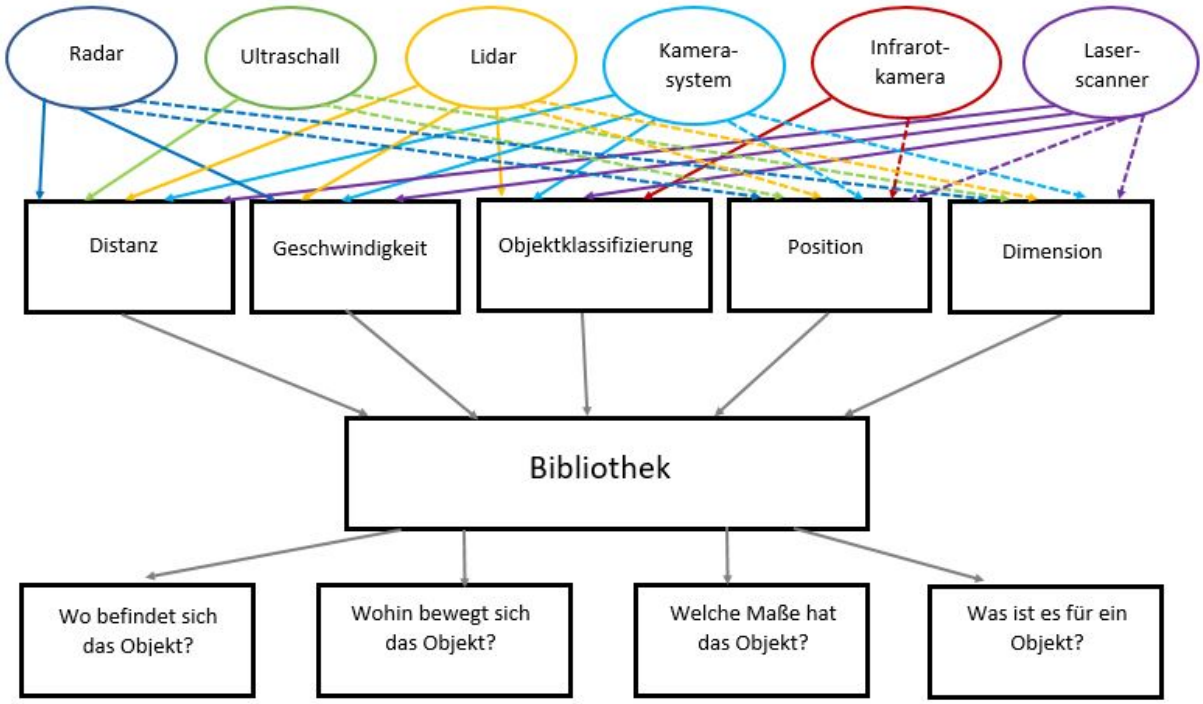


Abbildung 3.2: Übersicht Integration und Anwendung der Sensoren

Zum Ermitteln von Distanzen können sowohl Informationen von Radar-, Ultraschall- und Lidarsensoren eingebunden werden, als auch das Kamerasystem und der Laserscanner. Die Geschwindigkeiten können ebenfalls durch Radar- und Lidarsensoren ermittelt werden. Auch hier können Informationen vom Kamerasystem und dem Laserscanner verwendet werden. Für die Objektbestimmung und Kategorisierung können das Kamerasystem, die Infrarotkamera und der Laserscanner verwendet werden.

Je nach Sensorauswahl können theoretisch alle Sensoren, wenn sie die richtige Dimensionierung besitzen, zur Positionsbestimmung verwendet werden. Ob die Maße der Objekte ermittelbar sind, hängt ebenfalls von der Dimensionierung ab. Diese können durch alle Sensoren, außer der passiven Infrarotkamera, bei der richtigen Auslegung ausgegeben werden.

4 Entwicklung der Objekterkennungs-Bibliothek

Die Bibliothek wurde in Visual Studio als statische Bibliothek geschrieben. Als Programmiersprache wurde C++ ausgewählt. Sie ist nicht eigenständig lauffähig, sondern wird in einem Programm komplett oder teilweise aufgerufen. Durch die Definition als statische Bibliothek ist die Einbindung schnell und einfach möglich.

4.1 Aufbau und Funktionsprinzip der Bibliothek

Die Bibliothek besteht aus einer Basisklasse und Unterklassen mit deren Attributen. In der Abbildung 4.1 wird dieser Aufbau mithilfe von sechs verschiedene Beispielsensoren mit unterschiedlichen Informationen dargestellt.

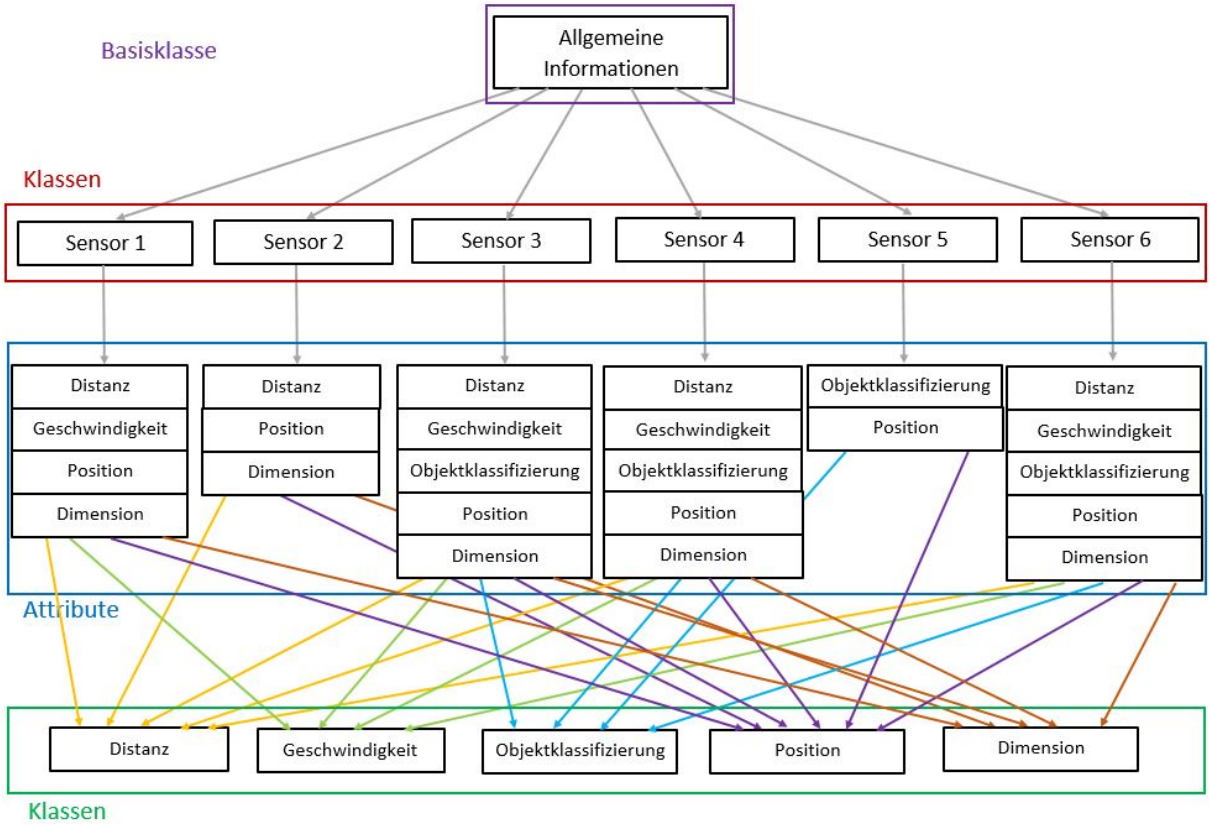


Abbildung 4.1: Aufbau der Bibliothek

In der Basisklasse werden alle Informationen vordefiniert und eingebunden. Dazu gehören die vordefinierten Variablen und Funktionen für die Distanz, Geschwindigkeit, Objektbestimmung, Position und Dimension. Zudem werden auch Variablen zur Temperatur und Geschwindigkeit des Fahrzeugs vordefiniert.

Die einzelnen Klassen sind der Basisklasse untergeordnet und beziehen sich auf die einzelnen Sensoren. Jeder Sensor der eingebunden wird, besitzt eine eigene ID, sodass auch die Verwendung von mehreren gleichen Sensoren einfach zu realisieren ist. Falls weitere Systeme eingebunden werden sollen, kann die Bibliothek mit weiteren Klassen erweitert werden. In den einzelnen Klassen gibt es verschiedene Attribute. Diese geben die Grundinformationen der einzelnen Sensoren einheitlich weiter.

Jede Information zum Fahrzeug wird zusätzlich in einer eigenen Klasse definiert und der Basisklasse untergeordnet. Mit Hilfe dieser Informationen können in der Bibliothek auch Faktoren, wie die Temperatur des Fahrzeuges, berücksichtigt werden.

Mit Hilfe von Klassen zur Bestimmung der jeweiligen Werte zu Distanz, Geschwindigkeit, Objekterkennung, Position und Dimension findet eine Sensorfusion und -ergänzung statt. In der Abbildung 4.2 wird aufgezeigt, wie die Klassen aufgeteilt sind.

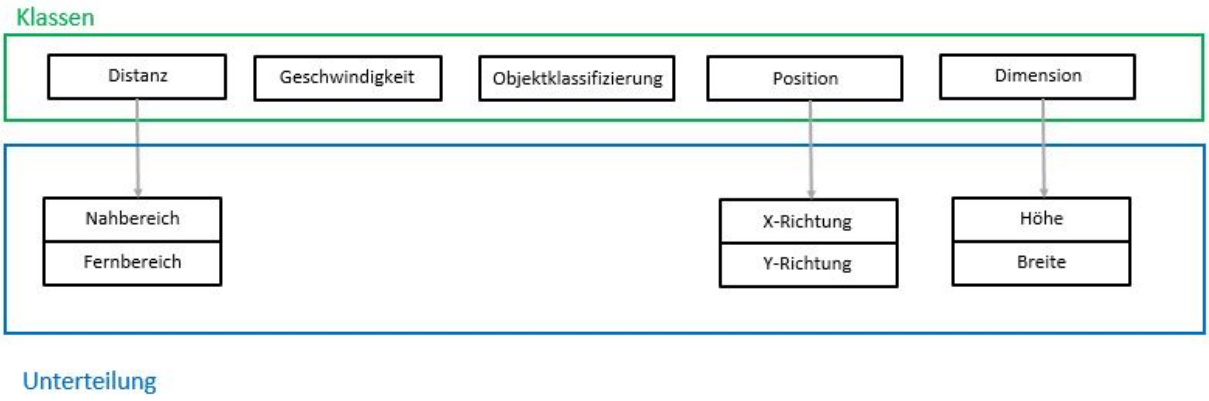


Abbildung 4.2: Unterteilung der Klassen

Wie schon erwähnt, haben bei der Distanz verschiedene Sensoren ihre Stärken in verschiedenen Distanzbereichen. Deshalb wird die Distanz in einen Nah- und einen Fernbereich unterteilt. Die Position des Objektes wird mit Hilfe eines Koordinatensystems dargestellt. Hierfür wird ein X- und Y-Wert benötigt. Die Dimension soll die Höhe und Breite des Objektes darstellen.

4.2 Fusion der Sensorwerte

Um die Sensoren miteinander fusionieren zu können, müssen die verschiedenen Sensorwerte miteinander verglichen und verrechnet werden. Die Distanzen und Geschwindigkeiten sind nach ihrer Position aufgeteilt. In der Abbildung 4.3 wird diese Aufteilung aufgezeigt.

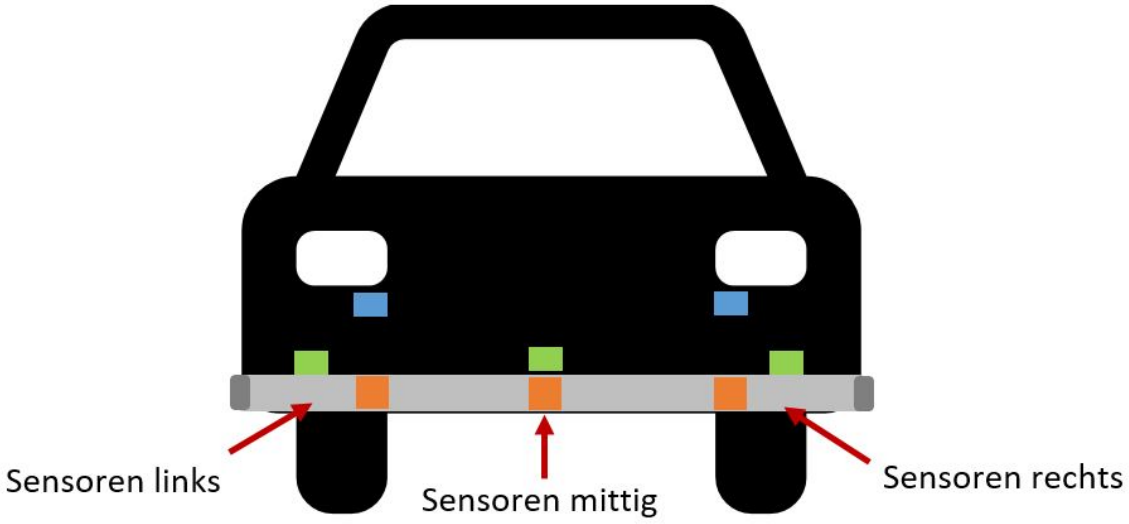


Abbildung 4.3: Anordnung der Sensoren

Die Berechnungen der Distanzen, Geschwindigkeiten, Positionen und Dimensionen finden über das arithmetische Mittel statt, siehe Formel 4.1. Hierfür ist eine eigene Klasse implementiert, sodass auf diese Berechnung einfach zugegriffen werden kann.

$$x = \frac{x_1 + x_2 + ... + x_n}{N} \tag{4.1}$$

4.2.1 Distanz

Die Sensoren werden je nach Einsatzgebiet in den Nah- oder Fernbereich eingebunden, siehe Abbildung 4.4. Hierbei werden sie dann jeweils noch einmal nach ihrer Lage unterteilt. Also in Sensoren die sich auf der linken und rechten vorderen Seite befinden oder die mittig platziert sind.

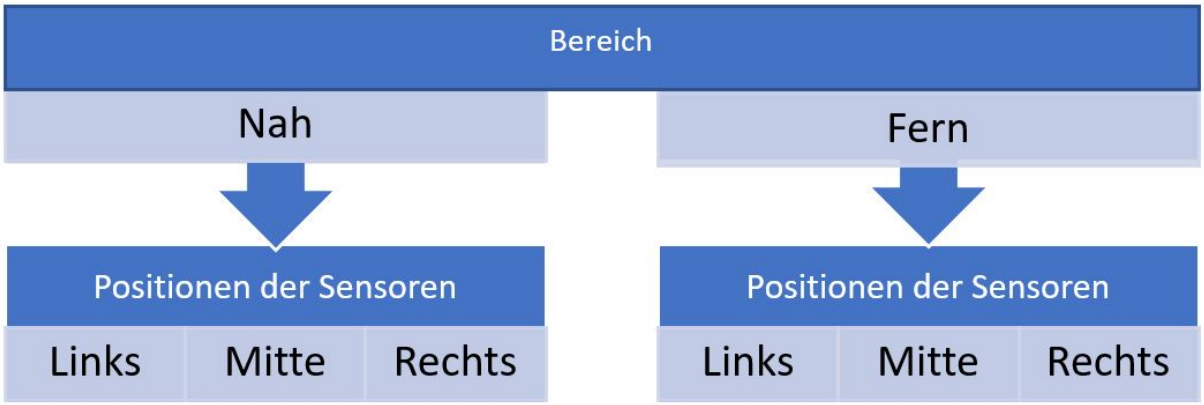


Abbildung 4.4: Aufbau Distanz

Aus den Werten der verschiedenen Sensoren werden dann die arithmetischen Mittel berechnet, siehe Formel 4.1 auf der vorherigen Seite. Die Berechnungen finden jeweils für links, Mitte und rechts im Nah- und Fernbereich statt.

4.2.2 Geschwindigkeit

Bei der Geschwindigkeit werden die Sensoren ebenfalls nach ihrer Lage aufgeteilt. Die Aufteilung wird in der Abbildung 4.5 dargestellt.

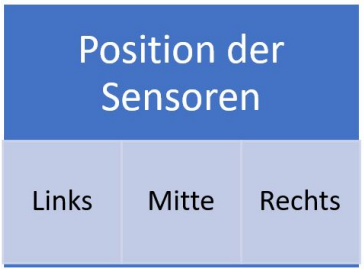


Abbildung 4.5: Aufbau Geschwindigkeit

Die Berechnung erfolgt auch hier für die jeweilige Unterteilung über den Mittelwert.

4.2.3 Objektklassifizierung

Die Klassifizierung findet über einen Vergleich der verschiedenen Sensorausgaben statt. Hierbei wird unterschieden, ob nur ein oder ob mehrere Objekte erkannt werden können, siehe Abbildung 4.6.

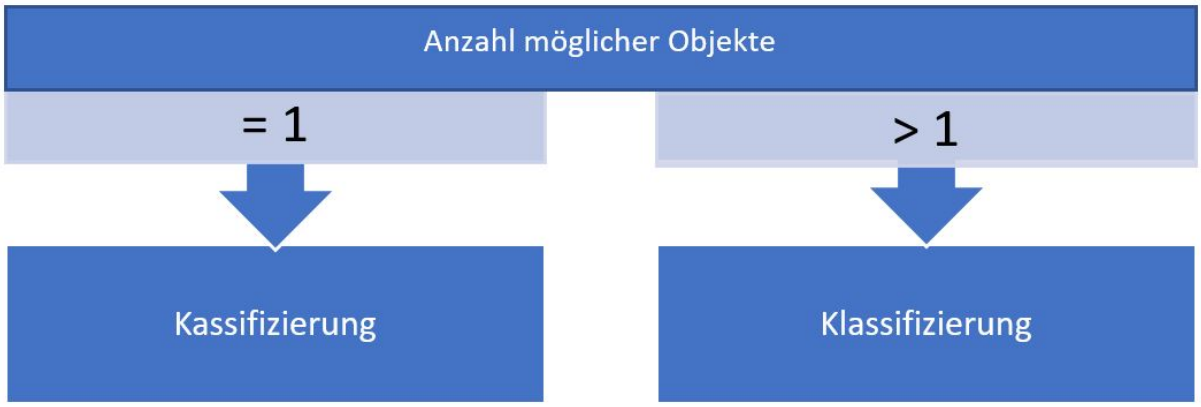


Abbildung 4.6: Aufbau Klassifizierung

Bei der einfachen Betrachtung wird erfasst, wie viele Sensoren das gleiche Objekt erkennen. Wird das Objekt von mehren Sensoren erkannt und gleich deklariert, wird die Klassifizierung ausgegeben.

Beispiel: Es sind drei Sensoren vorhanden. Zwei dieser Sensoren erkennen einen Hund und einer eine Katze, somit wird der Hund ausgegeben. Wenn das Objekt hingegen nicht eindeutig erkannt wird, dann kommt die Ausgabe 'no match'. Dies ist zum Beispiel der Fall, wenn drei Sensoren vorhanden sind und jeder eine andere Klassifizierung weiter gibt.

Bei der mehrfachen Betrachtung wird erfasst, wie viele Objekte erkannt werden. Des Weiteren werden alle der erkannten Objekte aufgeführt. Zudem wird erfasst, von wie vielen Sensoren die einzelnen Objekte erkannt werden.

4.2.4 Position

Die Positionsbestimmung wird, wie bei der Klassifizierung nach der Anzahl der Objekte unterteilt. Hier gibt es aber noch eine weitere Unterteilung, um die Lage des Objektes darstellen zu können. Dazu werden die Werte noch in einen X- und Y-Anteil aufgeteilt.

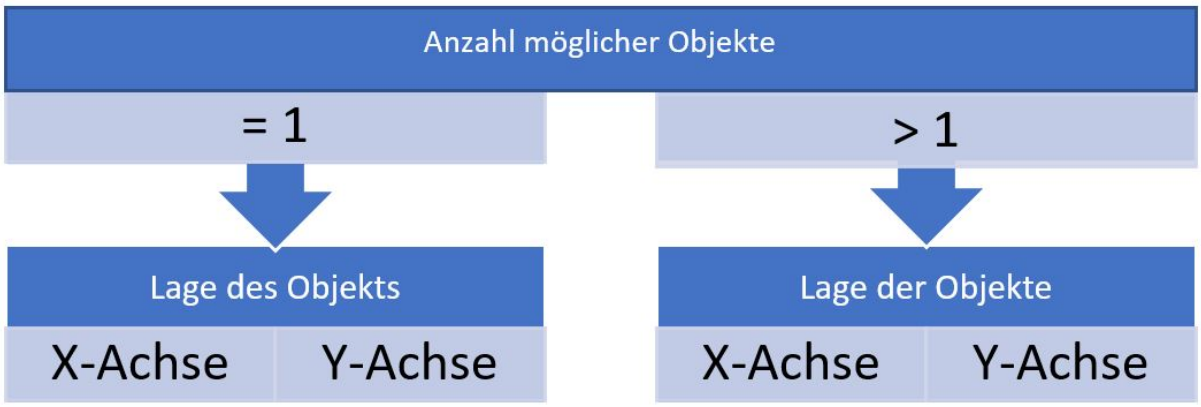


Abbildung 4.7: Aufbau Position

Zur Bestimmung der einzelnen Werte bei der einfachen Bestimmung wird mit dem arithmetischen Mittel gearbeitet. Die Mehrobjekt-Erkennung ist hierbei aktuell vorgemerkt und muss, sobald geeignete Sensordaten vorhanden sind, noch integriert werden.

4.2.5 Dimension

Die Aufteilung der Dimension ist ähnlich der Aufteilung der Position. Es wird nach der Objektbestimmung unterteilt und des Weiteren werden die Maße des Objektes in Höhe und Breite aufgeteilt.

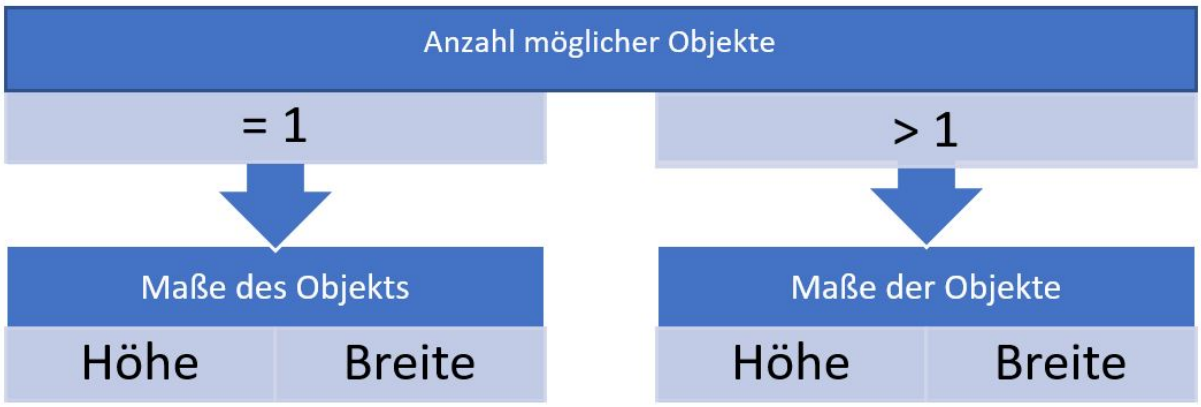


Abbildung 4.8: Aufbau Dimension

Die einzelnen Werte bei der einfachen Bestimmung werden über den Mittelwert berechnet. Auch hier ist die Mehrobjekt-Erkennung aktuell vorgemerkt und muss, sobald Sensordaten vorhanden sind, noch integriert werden.

4.3 Aufbau und Funktionsprinzip der graphischen Benutzeroberfläche

Es wurde ein Graphical User Interface (GUI) entworfen um die Ausgabe graphisch darzustellen. In der Abbildung 4.9 wird der Aufbau der GUI gezeigt.

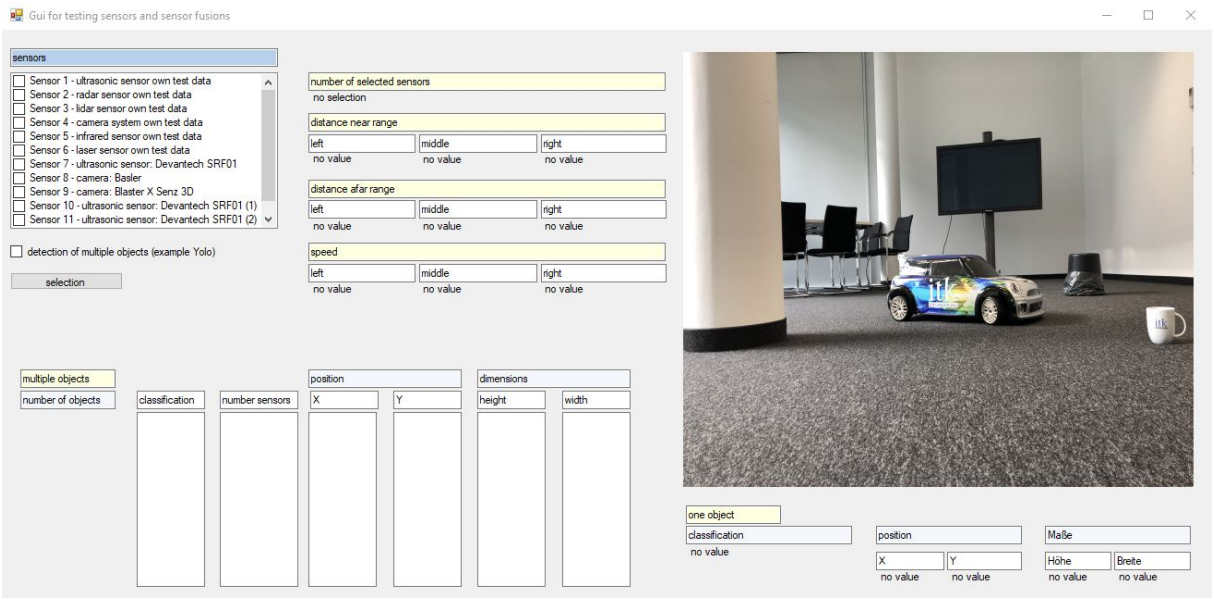


Abbildung 4.9: Aufbau der graphischen Benutzeroberfläche

In der GUI können die einzelnen Sensoren ausgewählt werden. Es wird angezeigt, wie viele Sensoren ausgewählt sind. Des Weiteren findet eine Ausgabe der berechneten Werte statt. Zuerst erfolgt die Distanz aufgeteilt in Nah- und Fernbereich, dann die Geschwindigkeit, jeweils aufgeteilt in Links, Mitte und Rechts. Es kann durch Aktivierung eines Kontrollkästchens entschieden werden, ob ein oder mehrere Objekte erkannt werden sollen. Bei mehreren Objekten wird die linke Einheit genutzt, bei einem Objekt die Rechte. Somit kann unterschieden werden, ob die Sensorausgabe nur auf einer Ebene stattfindet oder ob man mehrere Objekte erkennen und aufführen will. Sowohl in der einfachen, als auch in der Mehrobjekterkennung wird jeweils in die Klassifizierung, Position und Maße unterteilt. Bei der Erkennung von mehreren Objekten gibt es noch zusätzlich die Ausgabe der Anzahl von Sensoren, von denen das Objekt erkannt und klassifiziert wurde.

4.3.1 Auswahl der Sensoren und Objekterkennung

Das Element (Abbildung 4.10) bietet die Möglichkeit verschiedene Sensoren auszuwählen und deren Werte einzulesen. Des Weiteren kann ausgewählt werden, ob nur ein Objekt oder ob mehrere Objekte im Sensorbereich erkannt werden sollen. Mit klicken auf den Button 'Auswählen' startet der Aufruf der Klassen und die Berechnungen.

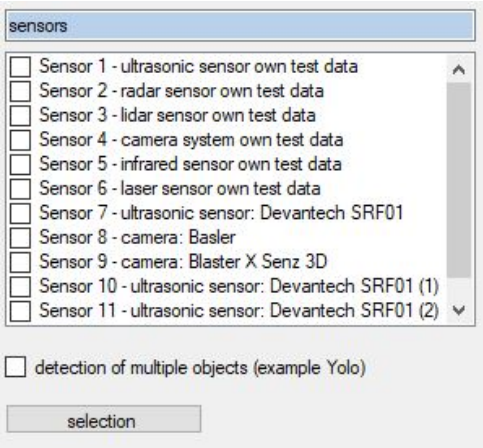


Abbildung 4.10: Auswahl der Sensoren und Objekterkennung

4.3.2 Ausgabe der Anzahl an Sensoren, Distanzen und Geschwindigkeiten

Unter 'number of selected sensors' in der Abbildung 4.11 wird angezeigt, wie viele Sensoren ausgewählt wurden. Darunter werden die berechneten Distanzen und Geschwindigkeiten gezeigt. Die Distanzen werden in Nah- und Fernbereich unterteilt. Auch wird in der Darstellung die Unterteilung in linke, rechte Seite und Mitte, sowohl bei den Distanzen als auch bei den Geschwindigkeiten, abgebildet.

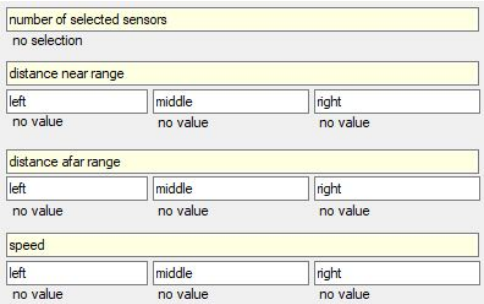


Abbildung 4.11: Ausgabe der Anzahl an Sensoren, Distanzen und Geschwindigkeiten

4.3.3 Ausgabe mit mehreren Objekten

Wenn die Auswahl zur Erkennung mehrerer Objekte aktiviert wurde, wird die Abbildung 4.12 aktiv. Hierbei wird ausgegeben, wie viele Objekte erkannt wurden und in der Spalte Klassifizierung, welche Objekte dies sind. Unter der Anzahl Sensoren wird angezeigt, wie vielen Sensoren die einzelnen Objekten erkannt haben. Bei der Position wird die Lage und bei den Maßen die Höhe und Breite der Objekte angegeben. Sowohl die Position als auch die Dimensionierung sind nur vorgemerkt, da keine geeigneten Testdaten vorhanden sind, und müssen noch in die Bibliothek integriert und mit geeigneten Sensoren getestet werden.

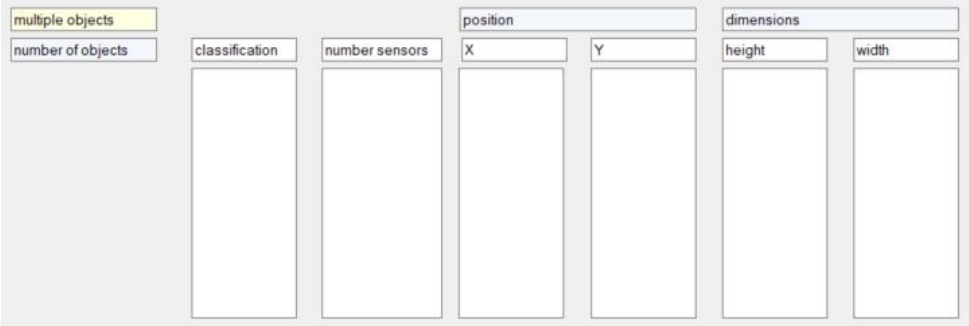


Abbildung 4.12: Ausgabe mit mehreren Objekten

4.3.4 Ausgabe mit einem Objekt

Falls die Auswahl zur Erkennung mehrerer Objekte nicht aktiviert wurden, wird die Abbildung 4.13 ausgewählt. Im Gegensatz zur Auswahl mit mehreren Objekten, wird hier nur ein Objekt dargestellt. Es wird das Objekt unter Klassifizierung angegeben und sowohl die Position, aufgeteilt in X und Y, als auch die Dimension, aufgeteilt in Höhe und Breite, ausgeben.

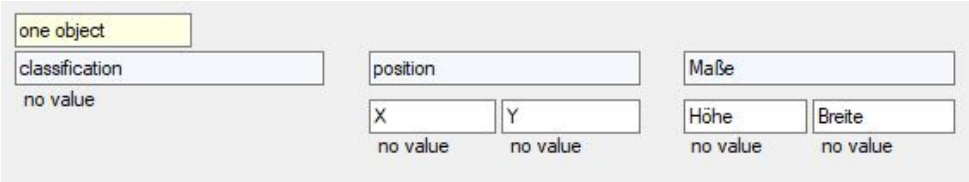


Abbildung 4.13: Ausgabe mit einem Objekt

4.4 Testdaten für die Entwicklung der Bibliothek

Um die Funktion der Bibliothek während der Programmierung testen zu können, wurden Testdaten für die einfache Bestimmung in Textdateien geschrieben. In der Abbildung 4.14 sieht man den Aufbau. Diese Testdaten sind anhand der aufgeführten möglichen Einsatzgebiete ausgelegt und jeweils 50 Zeilen lang. Es gibt für Ultraschall, Radar, Lidar, Laserscanner, Infrarotkamera und Kamerasystem jeweils eigene Textdateien.

Zeile	Daten

Abbildung 4.14: Allgemeiner Aufbau Testdaten

Die Werte des Ultraschallsensor wurden hierbei in fünf Sparten unterteilt, siehe 4.15. Die Aufteilung beinhaltet die Distanz, Position und Dimension.

Zeile	Distanz	X-Achse	Y-Achse	Höhe	Breite

Abbildung 4.15: Allgemeiner Aufbau Testdaten Ultraschall

Zu den Daten vom Ultraschallsensor kommen beim Radarsensor noch die Geschwindigkeit hinzu. Dies ist in der Abbildung 4.16 zu sehen.

Zeile	Distanz	Geschwindigkeit	X-Achse	Y-Achse	Höhe	Breite

Abbildung 4.16: Allgemeiner Aufbau Testdaten Radar

Beim Lidar kommt im Vergleich zum Radar die Klassifizierung hinzu. Wie schon erwähnt ist der Laserscanner eine Erweiterung zum Lidar, der Testdatenaufbau ist konform. Das Stereo-Kamerasystem bietet, wie Lidar und der Laserscanner, Informationen zur Distanz, Geschwindigkeit, Klassifizierung, Position und Dimension. In der Abbildung 4.17 werden die Unterteilungen aufgezeigt.

Zeile	Distanz	Geschwindigkeit	Klassifizierung	X-Achse	Y-Achse	Höhe	Breite

Abbildung 4.17: Allgemeiner Aufbau Testdaten Lidar

Die passive Infrarotkamera bietet Informationen zur Klassifizierung und zur Position, sieht Abbildung 4.18.

Zeile	Klassifizierung	X-Achse	Y-Achse

Abbildung 4.18: Allgemeiner Aufbau Testdaten Infrarotkamera

4.5 Einbindung von Messdaten in die Bibliothek

Um mit Messdaten arbeiten zu können, müssen diese zuerst mit den Sensoren aufgenommen werden. Diese Messdaten können dann entweder direkt in die Bibliothek eingebunden oder müssen vor der Einbindung noch bearbeitet werden. Ein Beispiel hierfür ist die Objektklassifizierung mit Hilfe von Yolo. Hierbei werden mit Hilfe von aufgenommen Daten, zum Beispiel durch eine Kamera, Objekte klassifiziert. Diese Ergebnisse können dann gespeichert und der Bibliothek übergeben werden. Der Ablauf wird in Abbildung 4.19 aufgezeigt.

Eine direkte Einbindung der Daten in die Bibliothek ist hierbei aktuell nicht möglich. Die Daten müssen erst erfasst und als Textdatei gespeichert werden, bevor die Bibliothek sie weiterverarbeiten kann. Dies dient als einheitliche Schnittstelle, da verschiedene Sensoren die Werte unterschiedlich ausgeben bzw. abspeichern. Zudem kann somit die Bibliothek Daten Schritt für Schritt einbinden und verarbeiten. Dadurch können einzelne Szenen erfasst und ausgewertet werden.

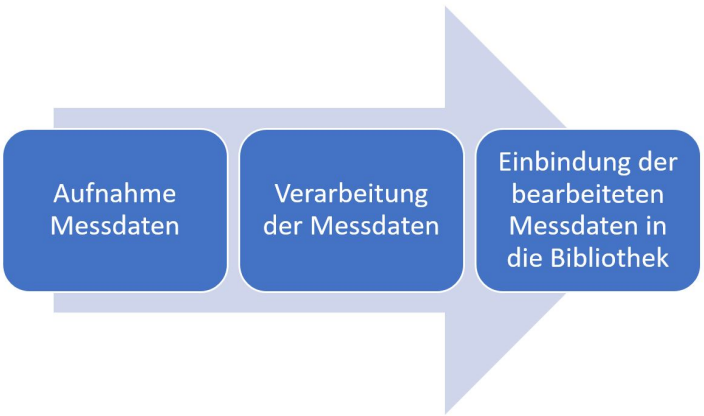


Abbildung 4.19: Ablauf Einbindung von Messdaten

4.6 Benutzerfreundliche Auslegung

Um die Erweiterbarkeit der Bibliothek, welche gefordert ist, zu realisieren, wurde für jeden Sensor eine eigene Klasse angelegt. Des Weiteren wurde für die einfache Erweiterung eine Kurzanleitung entworfen. Diese ist im Anhang angefügt. In der Anleitung wird erklärt, wie man Sensoren Schritt für Schritt dem System hinzufügen kann. Die Anleitung ist zusätzlich der Bibliothek als Textdatei beigefügt und sowohl in Deutsch als auch in Englisch verfasst.

Durch die Visualisierung über die GUI ist ein einfaches und verständliches Arbeiten mit der Bibliothek möglich. So können auch Personen ohne Programmiergrundlagen mit den eingebunden Sensoren arbeiten und verschiedenen Sensorfusionen testen. Durch verschiedene Farben wird zudem die Arbeit mit der Bibliothek weiter unterstützt. Auch können hierdurch die Werte leicht abgelesen und verstanden werden.

Zusätzlich ist durch das Abspeichern aller Werte in einer Textdatei auch möglich nach der Ausführung des Programms die Werte auszuwerten und zu vergleichen.

5 Evaluation

Bei der Evaluation wird mit Hilfe eines Testaufbaus und der Integration einzelner Versuchssensoren die Bibliothek mit der dazugehörigen GUI getestet. Zudem kann dadurch Optimierungspotential identifiziert werden.

5.1 Auswahl der Sensoren

Es wurden zwei verschiedene Sensorarten ausgewählt: Ultraschaltsensoren und Kamerasysteme. Diese Sensoren wurden verwendet, da sie schon im Unternehmen vorhanden sind. Zudem kann durch diese Kombination die Sensorfusion getestet werden. Die Sensoren sind in der Abbildung 5.1 dargestellt.

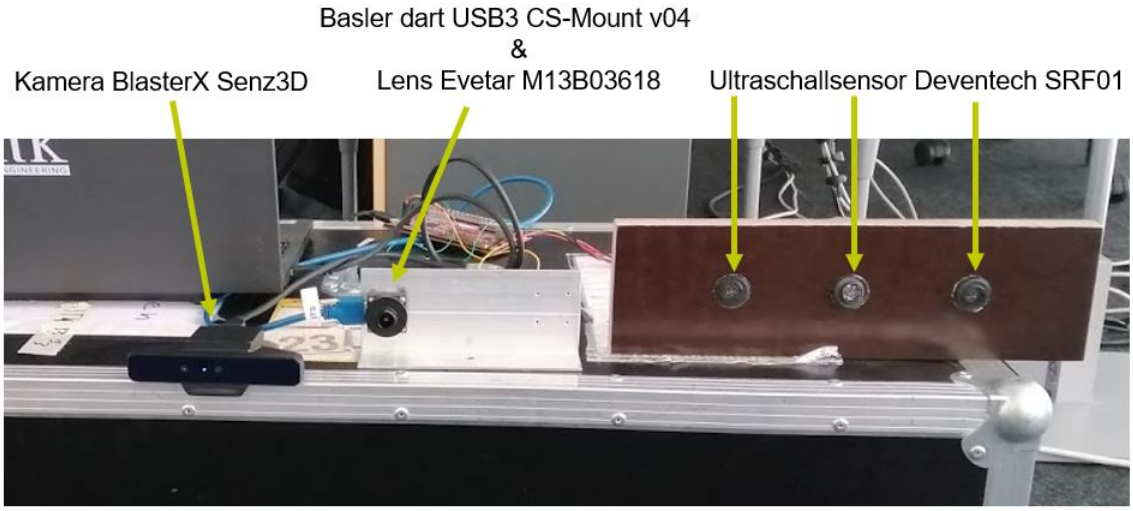


Abbildung 5.1: Aufbau Auswahl Sensoren

5.1.1 Ultraschallsensor Deventech SRF01

Der Ultraschallsensor Deventech SRF01 hat eine maximale Reichweite von 600 cm.[21] Es wurden drei dieser Sensoren verwendet, um die drei Bereiche auf der linken, mittleren und rechten Seite erfassen zu können.

Mit Hilfe des Code Composer Studio 8.3.0 können die Sensordaten der drei Sensoren abgerufen und gespeichert werden. Das Programm zur Ausgabe der Daten wurde in einer vorherigen Bachelorarbeit schon aufgestellt. Die Daten werden hierbei in Zentimeter ausgegeben.[22]

5.1.2 Kamerasystem BlasterX Senz3D

Die Webcam BlasterX Senz3D besitzt eine Tiefenwahrnehmung. Mit der Kamera ist es möglich Videos mit einer hohen Bildfrequenz zu streamen. Die Kamera besitzt drei Objektive. Zum einen ist eine RGB-Kamera eingebaut, des Weiteren eine Infrarotkamera und zusätzlich noch ein Laserprojektor.[23]

Für den folgenden Versuch wird auf die RGB-Kamera zugegriffen. Die Videoausgabe und das Abspeichern der Videos erfolgt über das OBS Studio.

5.1.3 Kamerasystem von Basler

Das Kamerasystem besteht aus einem 'Basler dart USB3 CS-Mount v4' mit einem 'Lens Evetar M13B03618 F1.8 f3.6mm 1/3"′ Objektiv. Hierbei wird mit einer mono RGB-Kamera gearbeitet.[24] Das Objektiv besitzt eine feste Brennweite, mit 3,6 mm zudem einen IR-Cut Filter und eine feste Blende mit 1,8.[25]

Die Ausgabe und Aufnahme der Kamera erfolgt über die Basler Video Recording Software.

5.2 Aufbau des Versuchs

Der Versuchsaufbau ist in Abbildung 5.2 dargestellt. Auf dem Bild sind verschiedene Objekte zu erkennen. Hierzu gehören Standard-Einrichtungsgegenstände, wie Stühle, Tische und ein Monitor. Zusätzlich zu den Einrichtungsgegenständen wurde eine Auto, eine Tasse und ein Mülleimer im Raum platziert. Ein Weiterer interessanter Punkt sind hierbei die weißen Säulen, da diese farblich mit der Wand verschmelzen.

Abbildung 5.2: Versuchsaufbau Ansicht Testobjekte

Die Sensoren sind wie in Abbildung 5.3 zu sehen ist, auf dem Boden platziert. Die Ultraschallsensoren sind dabei so ausgerichtet, dass sie auf verschiedene Objekte zeigen. Problematisch zeigt sich hierbei jedoch das vorgefertigte Brett für die Ultraschallsensoren, da hierbei ein großes Spiel bei der Passgenauigkeit vorhanden war. Die beiden Kameras wurden so ausgerichtet, dass sie jeweils die selben Objekte erfassen. Somit konnte man testen, ob beide Kameras die verschiedenen Objekte gleich gut erfassen. Des Weiteren ist es wichtig festzustellen, ob die Bibliothek die richtige Anzahl an Sensoren, welche die Objekte erkennen, auch ausgibt.

Abbildung 5.3: Versuchsaufbau Ansicht Sensoren

5.3 Ergebnisse

Die verschiedenen Daten der Sensoren müssen ausgewertet und verglichen werden, um eine erfolgreiche Funktion der Bibliothek nachweisen zu können. Hierbei liegt der Fokus vor allem auf der Objektklassifizierung. Dies liegt zum Einen daran, dass aufgrund des Zeitaufwandes keine Testdaten geschrieben wurden und zum Anderen an der Auswertung mit Hilfe von Yolo. Zudem ist es wichtig zu erkennen ob die Sensorfusion funktioniert.

Erwartet wird, dass die Kameras alle Objekte aufnehmen und über Yolo die verschiedenen Objekte erkannt werden. Des Weiteren sollen die Ultraschallsensoren aufgrund ihrer Ausrichtung drei verschiedene Objekte zeigen und bei der Distanz, Aufgeteilt in links, Mittel und rechts, drei verschiedene Werte ausgeben.

5.3.1 Messdaten

Distanzbestimmung

Die Werte der Ultraschallsensoren lassen sich in CSV-Dateien speichern. Die gespeicherte CSV-Datei kann dann als Textdatei umgewandelt und in die Bibliothek integriert werden. In der ersten Spalte wird hierbei der Zeitstempel angezeigt. In der zweiten Spalte gibt es eine Ausgabe der Daten in Abhängigkeit zum Zeitpunkt.

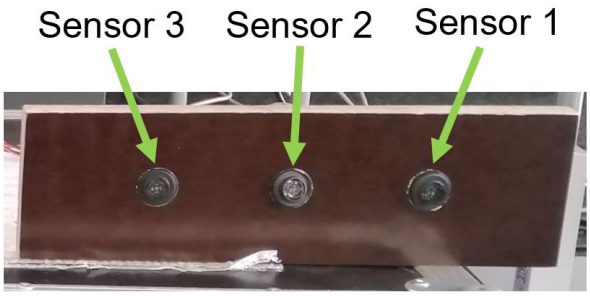


Abbildung 5.4: Anordnung der Ultraschallsensoren

In der Abbildung 5.4 wird die Anordnung der Ultraschallsensoren dargestellt. Der erste Ultraschallsensor ist hierbei, von hinten betrachtet, auf der linken Seite des Brettes eingesetzt und nach links ausgerichtet. Der zweite Sensor befindet sich in der Mitte und der Dritte auf der rechten Seite mit der Ausrichtung, nach rechts.

Ein Ausschnitt der gespeicherten Werte des ersten Ultraschallsensors wird in Abbildung 5.5 dargestellt. Es zeigt sich, dass der Sensor im Abstand von 50 cm ein Objekt gefunden hat. Auf Grundlage der Ausrichtung, wird hierbei wahrscheinlich der Pfosten erkannt.

	A	B
1	Sample:float	Data:float
2	0.0	50.0
3	1.0	50.0
4	2.0	50.0
5	3.0	50.0

Abbildung 5.5: Werte des ersten Ultraschallsensors

Auch für den zweiten Ultraschallsensor sind die Werte gespeichert, siehe Abbildung 5.6. Der Sensor zeigt ein Objekt in 83 cm Entfernung. Hierbei kann es sich aufgrund des Abstandes und der Ausrichtung um die Distanz zum Auto handeln.

	A	B
1	Sample:float	Data:float
2	0.0	83.0
3	1.0	83.0
4	2.0	83.0
5	3.0	83.0

Abbildung 5.6: Werte des zweiten Ultraschallsensors

Der dritte Ultraschallsensor weist ein Objekt in einem Abstand von 49 cm auf. Ein Ausschnitt der Aufnahme ist in Abbildung 5.7 zu sehen. Hierbei wird wahrscheinlich der Abstand zur Tasse gemessen.

	A	B
1	Sample:float	Data:float
2	0.0	49.0
3	1.0	49.0
4	2.0	49.0
5	3.0	49.0

Abbildung 5.7: Werte des dritten Ultraschallsensors

Klassifizierung

Bei den Kamerasystemen erfolgt die Aufnahme jeweils über die aufgeführte Software. Diese Aufnahmen werden dann mit Yolo für die Objektklassifizierung ausgewertet.

Ein Bild vom aufgenommen Video des Kamerasystems BlasterX Senz3D wird in Abbildung 5.8 gezeigt. Es ist zu erkennen, dass alle Objekte gut sichtbar sind, somit wird eine erfolgreiche Klassifizierung der Objekte erwartet.

Abbildung 5.8: Bild des Szenarios mit dem BlasterX Senz3D Kamerasystem

In der Abbildung 5.9 wird ein Ausschnitt der Auswertung mit Yolo gezeigt. Hierbei werden die einzelnen Objekte und die Wahrscheinlichkeit der richtigen Kategorisierung aufgeführt. Auch hier müssen die Daten wieder als Textdatei abgespeichert werden, sodass sie von der Bibliothek ausgewertet werden können.

```
Objects:

tvmonitor: 97%
chair: 97%
chair: 53%
cup: 70%
chair: 48%
car: 95%

FPS:0.0

 cvWriteFrame
```

Abbildung 5.9: Werte des BlasterX Senz3D Kamerasystems ausgewertet über Yolo

Das Kamerasystem von Basler weist hierbei eine schlechtere Bildqualität auf, wie in Abbildung 5.10 zu sehen ist. Die Objekte sind zum Teil schwer zu erkennen und zu kategorisieren.

Abbildung 5.10: Bild des Szenarios mit dem Basler Kamerasystem

Ein Ausschnitt der Auswertung über Yolo ist in Abbildung 5.11 aufgezeigt. Auch hier sind die erkannten Objekte mit ihrer Wahrscheinlichkeit aufgeführt. Das Abspeichern in eine Textdatei ist wieder Voraussetzung für die Auswertung über die Bibliothek.

```
Objects:

tvmonitor: 96%
chair: 31%
chair: 61%
car: 98%
chair: 84%

FPS:0.0

 cvWriteFrame
```

Abbildung 5.11: Werte des Basler Kamerasystems ausgewertet über Yolo

5.3.2 Ausgabe der GUI

Die einzelnen Textdateien mit den Messwerten werden anschließend in die Bibliothek integriert. Das heißt, sie werden jeweils in die passende Sensorklasse eingefügt. Die Abbildung 5.12 zeigt eine Übersicht zur Ausgabe der GUI für den Versuchsaufbau.

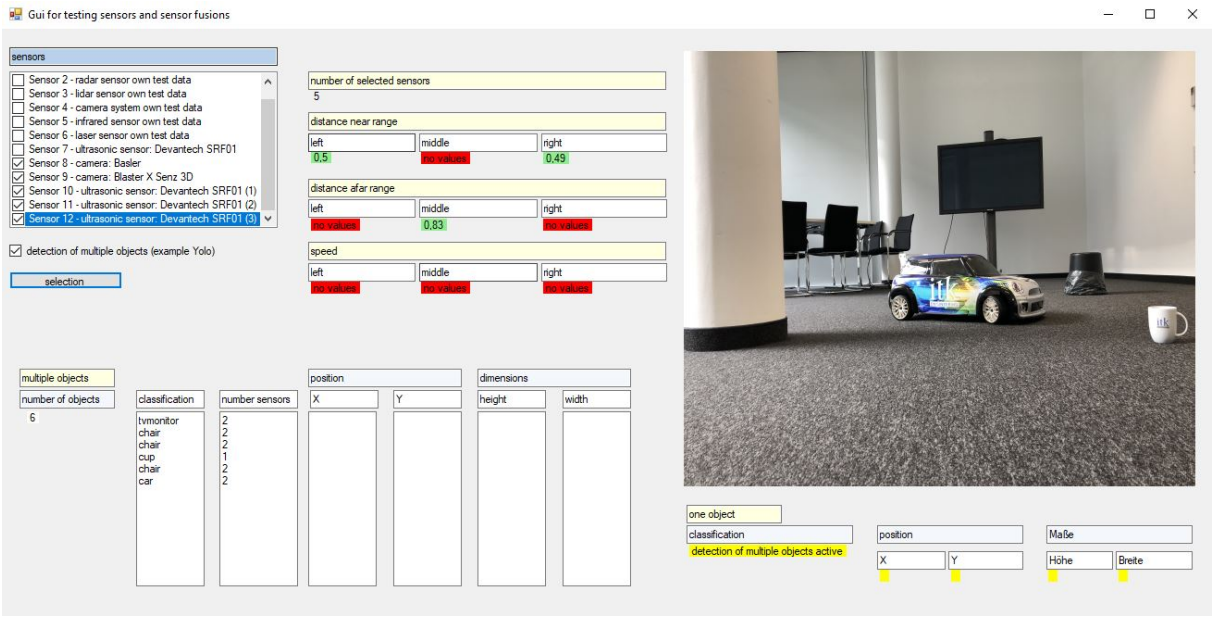


Abbildung 5.12: Ausgabe GUI beim Szenario

Es wurden, wie in Abbildung 5.13 dargestellt, die drei Ultraschallsensoren und die zwei Kamerasysteme ausgewählt. Zusätzlich fand eine Aktivierung für die Mehrobjekterkennung über das Feld 'detection of multiple objects (example Yolo)' statt.

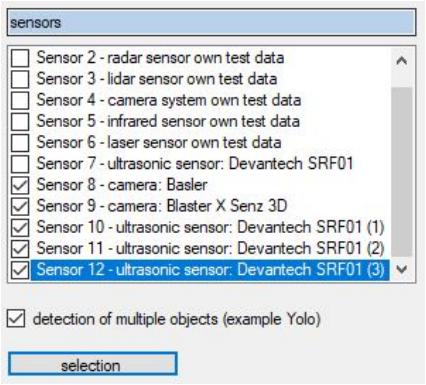


Abbildung 5.13: Auswahl der Sensoren und Objekterkennung beim Szenario

Die Ergebnisse der drei Ultraschallsensoren sind in Abbildung 5.14 aufgezeigt. Die Werte sind in den verschiedenen Distanzbereichen aufgeführt. Zum Testen der Nah- und Fernbereichserkennung wurde die Grenze zwischen den Bereichen auf 0,6 m gesetzt. Zusätzlich wird angezeigt das fünf Sensoren ausgewählt wurden.

Abbildung 5.14: Ausgabe der Anzahl an Sensoren, Distanzen und Geschwindigkeiten beim Szenario

In der Abbildung 5.15 ist die Ausgabe der Mehrobjekterkennung dargestellt. Es zeigt sich, dass sechs Objekte erkannt worden sind. Von diesen Objekten wurde jedoch die Tasse nur von einem der beiden Sensoren erkannt. Die anderen Objekte sind jeweils von beiden Sensoren erfasst worden.

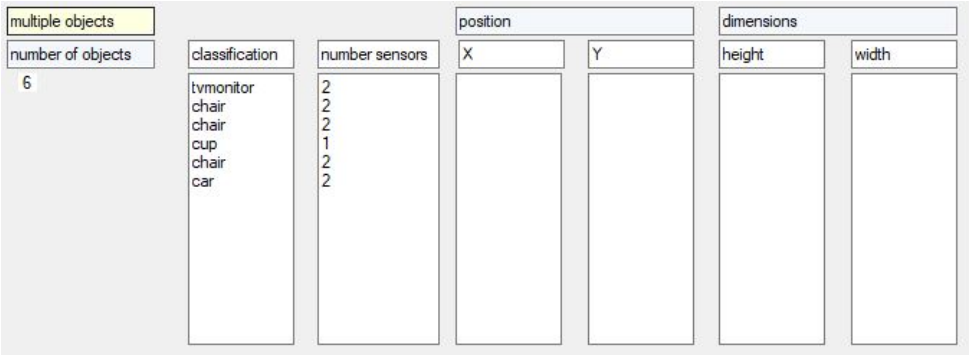


Abbildung 5.15: Ausgabe mit mehreren Objekten beim Szenario

5.4 Bewertung

Im Versuchsaufbau konnten einige Funktionen der Bibliothek und die Sensoren getestet werden. Es zeigt sich, dass bei den Ultraschallsensoren die Zuordnung der Werte zu den einzelnen Objekten schwer ist. Dies liegt daran, dass schon leichte Verkantungen der Sensoren nach links oder rechts die Ultraschallwellen in eine Richtung ablenken.

Das Kamerasystem BlasterX Senz3D zeigt nach der Auswertung über Yolo ein Objekt (die Tasse) mehr an als das Kamerasystem von Basler. Dies wird auf die gute Bildqualität zurückgeführt. Es ist interessant zu sehen, dass beide Systeme den Mülleimer nicht erkennen. Es wird vermutet, dass dies an der Tüte und an der Tatsache liegt, dass der Eimer verkehrt herum steht. Somit kann er nicht identifiziert werden. Auch wird der weiße Pfeiler nicht erkannt. Hierbei stellt sich jedoch die Frage, ob dies an der Farbe, die konform mit der Wandfarbe ist, liegt.

Durch die Auswertung erfolgt die Erkenntnis, dass mit der Bibliothek erfolgreich eine Sensorfusion stattfinden kann. Die Fusion und Ergänzung der einzelnen Sensoren zeigt sich als wichtig, da zum Beispiel der Pfeiler vom ersten Ultraschallsensor erkannt, jedoch von beiden Kamerasystemen nicht erkannt wurde.

6 Zusammenfassung und Ausblick

Die Masterarbeit thematisiert die Konzeptionierung einer Objekterkennungs-Bibliothek für Multi-Sensor-Systeme. Die Bibliothek soll in Simulationen und im Testbetrieb von autonomen Fahrzeugen eingesetzt werden.

Zu Beginn wurde sich in die Grundlagen eingearbeitet und die verschiedenen Sensoren mit ihren Vor-, Nachteilen und Einsatzgebieten kennen gelernt. Es wurde entschieden, welche Sensorsysteme für das autonome Fahren eine besondere Rolle spielen und welche Sensordaten aus ihnen gewonnen werden können. Des Weiteren war die Sensorfusion ein großes Thema. Hierbei stellte sich die Frage, wie die Sensordaten vereinheitlicht und miteinander verrechnet werden können.

Als nächstes wurde die Bibliothek konzeptioniert und entwickelt. Hierbei war die erste Überlegung zur Funktionsweise der Bibliothek. Es wurde im nächsten Schritt eine Klassenbibliothek konzeptioniert und überlegt, wie die Daten der unterschiedlichen Sensoren einheitlich in die Bibliothek integriert werden können. Hierbei fiel die Wahl auf die Einführung einheitlicher Textdateien. In diese können die Werte jedes einzelnen Sensors gespeichert werden.

Es musste zudem beachtet werden, welche Informationen die Bibliothek am Ende liefern soll und welche Werte die verschiedenen Sensoren ausgeben. Am Ende soll beantwortet werden können, wo sich das Objekt befindet, wohin es sich bewegt, welche Maße es hat und was für ein Objekt es ist. Im nächsten Schritt ging es darum die Programmierung nach den aufgestellten Anforderungen durchzuführen. Hierfür wurden auf Grundlage der objektorientierten Programmierung Klassen verwendet. Hierdurch sollte der Aufbau einfach gehalten werden und eine spätere Integration weiterer Klassen zu den Sensoren oder für weitere Informationen einfach sein.

Um später die Bewegungsrichtung ermitteln zu können und um auch im eindimensionalen Bereich eine grobe Positionsbestimmung zu haben, wurden die Sensoren in drei Gruppen aufgeteilt: linke Seite, Mitte und rechte Seite. Für jeden Bereich kann über das arithmetische Mittel aus verschiedenen Sensorwerten, die Distanz und Geschwindigkeit ermittelt werden. Zusätzlich kann über den Mittelwert die Berechnung der Position und Dimension stattfinden. Um dies einfach zu realisieren, wurde eine eigene Klasse für die Berechnung des arithmetischen Mittels programmiert. Die Dimension, Position und Kategorisierung wird in eine Ein- und Mehrobjekterkennung unterteilt. Somit kann der Benutzer selber entscheiden, ob er eine einfache oder vielfältige Erkennung haben möchte.

Die Objektklassifizierung muss hierbei gesondert betrachtet werden und stellte sich als größere Herausforderung dar. Hierbei können nicht einfach Sensorwerte verwendet werden, sondern es muss auf Grundlagen von Bildern zuerst eine Bestimmung der einzelnen Objekte stattfinden. Für die Bestimmung wurde das Programm Yolo ausgewählt. Hierbei findet die Klassifizierung mit Hilfe eines neuronalen Netzes statt.

Um die Ausgabe und Auswahl der Bibliothek benutzerfreundlich zu gestalten, wurde eine GUI entwickelt. In dieser kann man die verschiedenen Sensoren und die Ein- und Mehrobjekterkennung auswählen. Zusätzlich werden alle wichtige Informationen, wie zum Beispiel die Anzahl der ausgewählten Sensoren, ausgegeben. Als Zusatz werden alle Daten noch in einer Textdatei gespeichert. Mit Hilfe von selbst geschriebenen Testdateien konnten Tests während der Implementierung der Bibliothek stattfinden.

Um die Grundfunktion der Bibliothek nachzuweisen, wurde ein Versuchsaufbau entwickelt und mit drei Sensoren getestet. Es zeigte sich, dass die Bibliothek alle Werte fehlerfrei ausgibt und auch die Kategorisierung ohne Einschränkungen bei zwei Sensorwerten funktioniert. Alle bereits am Anfang gestellten Fragen können mit Hilfe verschiedener Sensorwerte durch die Bibliothek beantwortet werden. So kann die Lage und die Maße eines Objekts bestimmt und die Mehrobjekterkennung noch ergänzt werden. Auf Grundlage der Einteilung in drei Sensorgruppen kann man die Fragen, wohin sich Objekte bewegen können, beantworten. Was es für ein Objekt ist, wird auf Grundlage neuronaler Netze bestimmt und in der Bibliothek ausgewertet. Auch findet eine Ausgabe der Distanz und Geschwindigkeit statt.

Die Bibliothek ist noch nicht am Ende ihrer Entwicklung. So muss die Programmierung der Position und Dimension in der Mehrobjekterkennung noch ergänzt werden. Des Weiteren können noch Tests stattfinden, ob eine direkte Einbindung von Sensordaten mit den richtigen Sensoren möglich wäre. Auch ist es möglich die Bibliothek durch weitere Daten zu ergänzen, wie zum Beispiel durch die Temperatur des Motors oder die Eigengeschwindigkeit. Zusätzlich wäre es interessant zu sehen, ob eine Ausgabe der Bewegungsrichtung innerhalb eines Bildes möglich wäre. Somit könnten die Positionen der einzelnen Sensoren verfolgt werden.

Des Weiteren muss sie im realen Einsatz noch getestet werden, das heißt in der Simulation und im Testbetrieb autonomer Fahrzeuge. Hierfür kann die statische Bibliothek zum Beispiel in eine virtuelle Umgebung eingebunden werden und mit Hilfe von Testdaten dabei helfen verschiedene Verkehrssituation mit unterschiedlichen Daten und Sensoren zu simulieren.

Auch bietet die Klassifizierung noch viele interessante Möglichkeiten. So kann auch mit anderen Programmen als Alternative zu Yolo getestet oder mehrere Alternativen mit in die Programmierung aufgenommen werden. So ist es beispielsweise möglich, wenn eine schnelle oder genaue Erfassung nötig ist, die passende Lösung auszuwählen.

Es ist fraglich, ob eine Unterteilung der Sensoren in drei Bereiche (links, Mitte, rechts) für den Einsatz auf der Straße ausreicht. Hierbei kann es nötig sein weitere Unterteilungen vorzunehmen. Zudem wird aktuell nur die Frontseite des Autos simuliert. Da ein Auto aber auch an den Seiten und Hinten seine Umgebung wahrnehmen muss, müssen diese Punkte in der Weiterentwicklung zusätzlich berücksichtigt werden.

Wie am Anfang beschrieben, gab es schon in früheren Zeiten eine Art des 'autonomen Fahrens'. Ziel ist es mit Hilfe maschineller Wahrnehmung nun auch ohne Pferde autonome Fahrten zu realisieren. Dies zeigt sich auf einigen Gebieten schon als sehr erfolgreich.[1, S. 2–3]

Die Objekterkennungs-Bibliothek soll ein weiterer Schritt in diese Richtung sein und bei der Weiterentwicklung der maschinellen Wahrnehmung unterstützen.

Quellenverzeichnis

[1] Markus Maurer u. a. *Autonomes Fahren: technische, rechtliche und gesellschaftliche Aspekte*. Springer-Verlag, 2015.

[2] Eckard Minx und Rainer Dietrich. *Autonomes Fahren*. Piper ebooks, 2015.

[3] sp-x. *Unfallstatistik: Autonome Autos in den meisten Fällen nicht Schuld*. Hrsg. von kfz-betrieb. 17. Dez. 2018. URL: https://www.kfz-betrieb.vogel.de/unfallstatistik-autonome-autos-in-den-meisten-faellen-nicht-schuld-a-785596/ (besucht am 18. 10. 2019).

[4] Diana Künstler. *Sensoren für autonome Fahrzeuge: Rigorose Real-World-Tests sind ein Muss*. Hrsg. von funkschau. 28. Feb. 2019. URL: https://www.funkschau.de/telekommunikation/artikel/162962/ (besucht am 13. 06. 2019).

[5] Paul Balzer. *Fahrzeugumfeldsensorik: Überblick und Vergleich zwischen Lidar, Radar, Video*. Hrsg. von MOTORBLOG. 29. Juli 2014. URL: https://www.cbcity.de/fahrzeugumfeldsensorik-ueberblick-und-vergleich-zwischen-lidar-radar-video (besucht am 20. 05. 2019).

[6] Harry Wagner und Stefanie Kabel. *Mobilität 4.0–neue Geschäftsmodelle für Produkt- und Dienstleistungsinnovationen*. Springer, 2018.

[7] Christian Rohde. *Fachvortrag "Umfeldsensorik mit dem Lidar: Vor- und Nachteile ggü. Radar"*. Hrsg. von Hochschule für Technik und Wirtschaft Dresden. 19. Nov. 2012. URL: http://www.htw-mechlab.de/index.php/fachvortrag-umfeldsensorik-mit-dem-lidar-vor-und-nachteile-ggu-radar/ (besucht am 20. 05. 2019).

[8] Ferdinand Seidel. "Implementierung und Evaluierung einer radarbasierenden Abstandsmessung für einen Quadrocopter". Bachelorarbeit. Universität Würzburg, 6. Nov. 2014.

[9] David Koscheck. "Konzeption und Evaluierung einer Abstandssensorik für ein Versuchsfahrzeug". Bachelorarbeit. Hochschule Karlsruhe, 2014.

[10] WayCon Positionsmesstechnik GmbH. *Ultraschallsensoren - Messprinzip*. Hrsg. von WayCon Positionsmesstechnik GmbH. URL: https://www.waycon.de/produkte/ultraschallsensoren/messprinzip-ultraschallsensoren/ (besucht am 21. 05. 2019).

[11] Doug Newcomb. *How far infrared (FIR) technology enables 24/7 autonomous driving in any condition*. Whitepaper. AdaSky, 2018.

[12] Joseph Redmon und Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018).

[13] Michael Hilgers. "Fahrerassistenzsysteme". In: *Elektrik und Mechatronik*. Springer, 2016, S. 59–68.

[14] Florian Wetzel Mussie Beian Jan Schupp. *Infrarotspektroskopie*. Fortgeschrittenenpraktikum. TU Darmstadt, 2009.

[15] Wilbur de Souza. *Sensor Fusion Algorithms For Autonomous Driving: Part 1 - The Kalman filter and Extended Kalman Filter*. Hrsg. von Medium. 13. Juni 2017. URL: https://medium.com/@wilburdes/sensor-fusion-algorithms-for-autonomous-driving-part-1-the-kalman-filter-and-extended-kalman-a4eab8a833dd (besucht am 13. 06. 2019).

[16] Andreas Zell. *Simulation neuronaler Netze*. Bd. 1. 5.3. Addison-Wesley Bonn, 1994.

[17] Michael Ziegler. *Object Recognition with Convolutional Neural Networks*. Hrsg. von Universität Bamberg.

[18] Roland Becker. *CONVOLUTIONAL NEURAL NETWORKS – AUFBAU, FUNKTION UND ANWENDUNGSGEBIETE*. Hrsg. von JAAI. 6. Feb. 2019. URL: https://jaai.de/convolutional-neural-networks-cnn-aufbau-funktion-und-anwendungsgebiete-1691/ (besucht am 30. 10. 2019).

[19] Stefan Anders. *Yolo*. Hrsg. von Deep Data Ocean GmbH i. Gr. 2019. URL: https://deepdataocean.ai/de/deep-learning/yolo (besucht am 09. 10. 2019).

[20] Prof. Dr. Hans-Werner Six. *Objektorientiertes Software Engineering*. Zusammenfassung zum Skript. Fernuniversität in Hagen.

[21] RobotShop Inc. *SRF01 Ultrasonic range finder: Technical Documentation*. Hrsg. von RobotShop Inc. URL: http://www.robot-electronics.co.uk/htm/srf01tech.htm (besucht am 10. 10. 2019).

[22] Melanie Glomb. "Entwicklung eines Notfall-Bremsassistenten auf Basis von Ultraschall-Sensorsignalen". Bachelorarbeit. Hochschule Furtwangen, 2019.

[23] Creative Labs (IRL) Ltd. *BlasterX Senz3D*. Hrsg. von Creative Labs (IRL) Ltd. URL: https://de.creative.com/p/web-cameras/blasterx-senz3d (besucht am 14. 10. 2019).

[24] Basler AG. *Basler dart Serie*. Hrsg. von Basler AG. URL: https://www.baslerweb.com/de/produkte/kameras/flaechenkameras/dart/ (besucht am 14. 10. 2019).

[25] Basler AG. *Evetar Lens M13B03618W F1.8 f3.6mm 1/3 Objektive*. Hrsg. von Basler AG. URL: https://www.baslerweb.com/de/produkte/vision-komponenten/objektive/evetar-lens-m13b03618w-f1-8-f3-6mm-1-3/#tab=specs (besucht am 14. 10. 2019).

Abbildungsverzeichnis

Abkürzungsverzeichnis

CNN Convolutional Neural Network

GUI Graphical User Interface

Lidar Light Detection and Ranging

Radar Radio Detection and Ranging

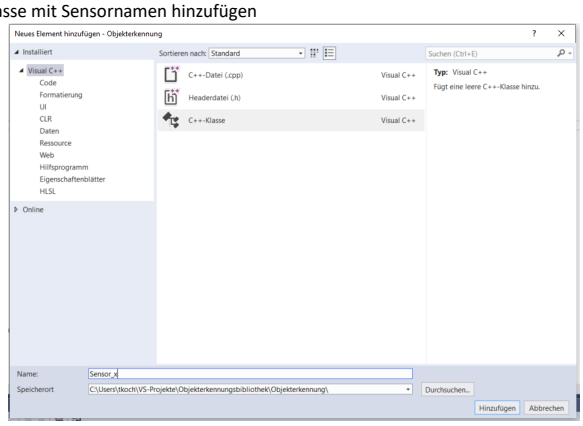
Yolo You Only Look Once

A Anleitung für die Sensor-Integration

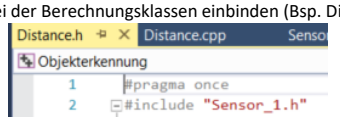
Anleitung - Integration eines Sensors

Bibliothek

1. Neue Klasse mit Sensornamen hinzufügen



2. Include Verzeichnisse und Definitionen in die Headerdatei des Sensors einbinden
3. Vordefinierte Funktionen in die cpp Datei der Klasse einfügen und Sensorspezifisch befüllen
4. Sensor in die Headerdatei der Berechnungsklassen einbinden (Bsp. Distanz)



&

```
bool sensor1 = false;
```

5. Sensor in den cpp Dateien der Berechnungsklassen in den gewünschten Funktionen auf true abfragen und falls gewünscht in der Step-Funktion hochzählen lassen

```
if (sensor1 == true)
{
    numSen_disNear++;
    if (sensor1_dis.range_near() >= 0.0)
    {
        values_disNear.push_back(sensor1_dis.range_near());
    }
}
```

&

```
//reset the vectors and select the next line of recorded values
int Distance::steps()
{
    sensor1_dis.tmp_dis++;
```

Abbildung A.1: deutsche Anleitung für eine Sensor-Integration - Bibliothek

GUI

6. Den Array sensor auf die Anzahl der Sensoren anpassen

```
//number of sensors which can be selectet
std::array<bool, 12> sensor;
```

7. Sensor in der Liste Auswahl Sensoren hinzufügen (MyForm.h [Entwurf])

8. Sensor in MyForm.h hinzufügen und auf true abfragen

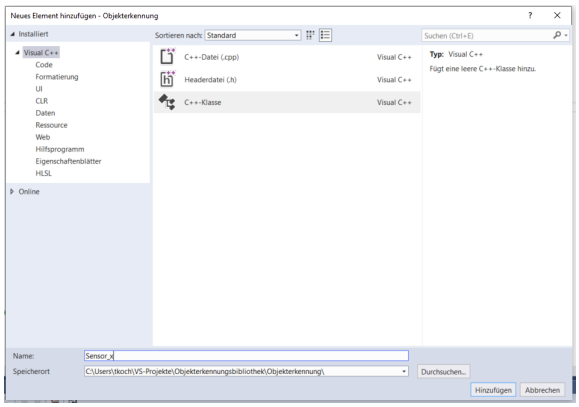
```
if (sensor[0] == true)
{
    distance_cl.sensor1 = true;
    speed_cl.sensor1 = true;
    position_cl.sensor1 = true;
    dimension_cl.sensor1 = true;
    class_cl.sensor1 = true;
    num_sen++;
}

else
{
    distance_cl.sensor1 = false;
    speed_cl.sensor1 = false;
    position_cl.sensor1 = false;
    dimension_cl.sensor1 = false;
    class_cl.sensor1 = false;
}
```

Abbildung A.2: deutsche Anleitung für eine Sensor-Integration - GUI

Instruction - sensor integration

Library

1. add a new class with sensor names



2. include directories and definitions into the header file of the sensor
3. insert predefined functions into the cpp file of the class and fill sensor-specific
4. include sensor in the header file of the calculation classes (e.g. distance)

```
Distance.h   Distance.cpp   Sensc
Objekterkennung
1    #pragma once
2    #include "Sensor_1.h"
```

&

```
bool sensor1 = false;
```

5. count up the sensor in the cpp files of the calculation classes in the desired functions on true queries and if desired in the step function

```
if (sensor1 == true)
{
    numSen_disNear++;
    if (sensor1_dis.range_near() >= 0.0)
    {
        values_disNear.push_back(sensor1_dis.range_near());
    }
}
```

&

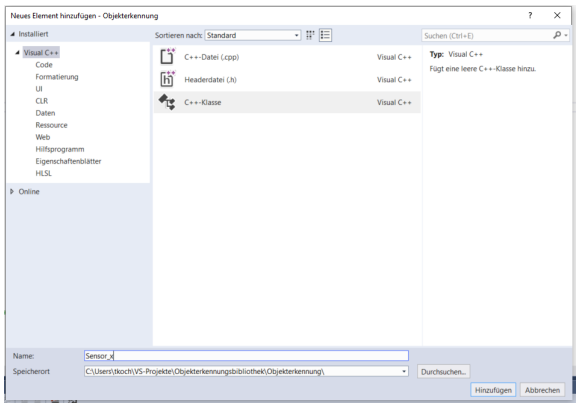
```
//reset the vectors and select the next line of recorded values
int Distance::steps()
{
    sensor1_dis.tmp_dis++;
```

Abbildung A.3: englische Anleitung für eine Sensor-Integration - Bibliothek

Instruction - sensor integration

Library

1. add a new class with sensor names



2. include directories and definitions into the header file of the sensor
3. insert predefined functions into the cpp file of the class and fill sensor-specific
4. include sensor in the header file of the calculation classes (e.g. distance)

```
Distance.h   Distance.cpp   Sensc
Objekterkennung
1    #pragma once
2    #include "Sensor_1.h"
```

&

```
bool sensor1 = false;
```

5. count up the sensor in the cpp files of the calculation classes in the desired functions on true queries and if desired in the step function

```
if (sensor1 == true)
{
    numSen_disNear++;
    if (sensor1_dis.range_near() >= 0.0)
    {
        values_disNear.push_back(sensor1_dis.range_near());
    }
}
```

&

```
//reset the vectors and select the next line of recorded values
int Distance::steps()
{
    sensor1_dis.tmp_dis++;
```

Abbildung A.4: englische Anleitung für eine Sensor-Integration - GUI