HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

MASTERARBEIT

Herr B.Sc.
Kevin Blümel

Konzeption und Entwicklung eines Warnsystems zur Detektion von ortsveränderlichen Objekten mit vibrotaktilen Feedback

Mittweida, 2021

Fakultät Ingenieurwissenschaften

MASTERARBEIT

Konzeption und Entwicklung eines Warnsystems zur Detektion von ortsveränderlichen Objekten mit vibrotaktilen Feedback

Autor:
Herr B.Sc.

Kevin Blümel

Studiengang:
Elektro- und Informationstechnik

Seminargruppe:
EI19w1-M

Erstprüfer:
Prof. Dr.-Ing. Michael Kuhl

Zweitprüfer:
M.Sc. Christian Georgi

Einreichung:
Mittweida, 17.08.2021

Verteidigung/Bewertung:
Mittweida, 2021

Faculty Engineering

MASTER‘S THESIS

Design and development of a warning system for the detection of moveable objects with vibrotactile feedback

author:
Mr. B.Sc.

Kevin Blümel

course of studies:
Electronic- and Informationtechnology

seminar group:
EI19w1-M

first examiner:
Prof. Dr.-Ing. Michael Kuhl

second examiner:
M.Sc. Christian Georgi

submission:
Mittweida, 17th of August 2021

defence/ evaluation:
Mittweida, 2021

Bibliografische Beschreibung:

Blümel, Kevin:
Konzeption und Entwicklung eines Aktorsystems,
88 Seiten, 50 Abbildungen, 18 Tabellen, 6 Anlagen,
Hochschule Mittweida (FH),
Fakultät Ingenieurwissenschaften

Masterarbeit, 2021

Referat:

Diese Arbeit befasst sich mit der Entwicklung eines mobilen Warnsystems für Werkarbeiter. Das System wird in eine Arbeitsweste integriert. Dabei messen Radar-Sensoren und eine ToF-Kamera den Bereich außerhalb des Gesichtsfeldes zur Detektion von Objekten. Nach einer Auswertung der Sensordaten und einer Gefahrenanalyse wird über ein taktiles Feedback mit Hilfe von integrierten Vibrationsmotoren in der Arbeitsweste der Werkarbeiter vor potenziell gefährlichen Objekten gewarnt.

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

µC	**M**ikro**c**ontroller
PCB	**P**rinted **C**ircuit **B**oard
LED	**L**ight-**E**mitting **D**iode
RGB	**R**ot **G**rün **B**lau
PWM	**P**ulse-**w**idth **m**odulation
PC	**P**ersonal **C**omputer
IDE	**I**ntegrated **D**evelopment **E**nvironment
Com-Port	**Com**munication-**Port**
UART	**U**niversal **A**synchronous **R**eceiver **T**ransmitter
USART	**U**niversal **S**ynchronous/**A**synchronous **R**eceiver **T**ransmitter
FMCW	**F**requency **M**odulated **C**ontinuous **W**ave
FFT	**F**ast **F**ourier **T**ransform
PMD	**P**hotonic **M**ixing **D**evice
USB	**U**niversal **S**erial **B**us
GUI	**G**raphical **U**ser **I**nterface
TCP/IP	**T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol
W-LAN	**W**ireless **- L**ocal **A**rea **N**etwork
LAN	**L**ocal **A**rea **N**etwork
WiFi	**Wi**reless **Fi**delity
FPS	**F**rames **P**er **S**econd
I2C	**I**nter-**I**ntegrated **C**ircuit
SDA	**S**erial **D**ata **L**ine
SCL	**S**erial **C**lock **L**ine

1 Einleitung

1.1 Motivation

Im Industrieumfeld findet eine fortlaufende Modernisierung bzw. Digitalisierung statt, um Arbeitskräfte zu entlasten und die Produktion effektiver gestalten zu können. Die entscheidenden Hilfsmittel sind z.B. bei dem Transport von Gütern von Station zu Station autonom fahrende Gabelstapler und andere digitalisierte, von einer Maschine gesteuerte Transportmittel.
Der Arbeiter in einer Produktionshalle bewegt sich dadurch in einer Sicherheitszone, in der seine Gesundheit an erster Stelle stehen muss. Um diese Sicherheit zu gewährleisten, müssen automatisierte Sicherheitsroutinen frühzeitig potenzielle Gefahren erkennen und effektiv reagieren. Auch der Arbeiter muss über diese Gefahren in Kenntnis gesetzt werden.

Anders als vernetzte Subsysteme in der Produktionshalle können Arbeiter nicht direkt per W-Lan oder einer anderen üblichen Schnittstelle gewarnt werden. Hier muss eine Schnittstelle entworfen werden, um den Arbeiter in das Sicherheitssystem integrieren zu können.

Durch die äußeren Einflüsse in einer Produktionshalle, z.B. Lärm bei der Metallbearbeitung, Hitze durch Hochöfen, oder grelles Licht durch Schweißarbeiten, werden mehrere Sinne des Arbeiters, hier der visuelle, auditive und Thermosinn, überflutet und können nicht zur eindeutigen Wahrnehmung von Gefahren oder Informationen verwendet werden.

Der Tastsinn des Menschen bleibt jedoch in der beschriebenen Umgebung vorerst unberührt und könnte für eine Mensch-Maschinen-Schnittstelle Verwendung finden. In dieser Arbeit wird daher ein Warnsystem beschrieben, welches Gefahren in der Nähe eines Arbeiters erkennt und mit Hilfe von vibrotaktilen Aktoren Richtungsinformationen der Gefahr an den Arbeiter übermittelt.

1.2 Aufgabenstellung

Ziel ist die Erarbeitung eines Warnsystems für den Werkarbeiter mit einem Interface zwischen Mensch und Maschine. Aufgrund der teilweisen Sinnesüberflutung in einer Produktionshalle sollen hier vibrotaktile Aktoren Verwendung finden.

Die Arbeit soll das System im Ganzen beschreiben und für die Verwendung durch Dritte verständlich sein. Dabei sollen folgende Punkte besonders ausgeführt werden:

- Erarbeitung eines Systemkonzepts
- Auswahl und Vergleich von Systemkomponenten
- Entwicklung einer funktionierenden Hardware
- Entwicklung einer geeigneten Software
- Erarbeitung und Implementierung eines Bedienkonzepts
- Aufbau und Inbetriebnahme eines Prototyps

Anforderungen

Die Anforderungen ergeben sich aus der Vorhabensbeschreibung des BMBF-geförderten Projektes „3D-Image to Haptic Device“ (3Dim-Hapt) in der Professur Systemelektronik der Hochschule Mittweida, zu finden in den Anlagen, Teil 1 (Seite 71). Dieses Projekt geht in ein EFRE- bzw. durch die Sächsische Aufbaubank gefördertes Kooperationsprojekt „Systemübergreifende Sicherheit autonomer und teilautonomer Systeme“ (3Dsys) mit der Technischen Universität in Chemnitz ein. Abbildung 1 zeigt beispielhaft die finale Systemstruktur von 3Dsys. Die Arbeit befasst sich mit dem taktilen Warnsystem.

3D-Umgebungsinformationen sollen in Form von taktilen Informationen in einem mobilen/tragbaren System im Werksverkehr Unfälle vermeiden. Dabei sollen durch haptisches Feedback Gefahrenquellen mit mehreren Gefährdungsintensitäten und Richtungsinformationen signalisiert werden.

Die Richtungsinformationen sollen mindestens links und rechts unterscheidbar machen. Es werden also mindestens zwei aktive Flächen benötigt. Nach entsprechender Systematisierung werden zehn aktive Flächen eingeplant, um eine bessere räumliche Wahrnehmung zu ermöglichen. Hierbei ist ein Abstand von mindestens 5 Zentimetern zwischen den einzelnen aktiven Flächenmittelpunkten einzuhalten, damit die Richtungsinformationen unterscheidbar bleiben. Die Reaktionszeit des Systems soll höchstens 25 Millisekunden betragen. Das System soll für einen Werkarbeiter tragbar sein und soll somit nicht mehr als 1 Kilogramm wiegen. Schutzkleinspannung (kleiner gleich 60 Volt) wird bei Hautkontakt gefordert und bei einem maximalen Strom von 5 Ampere ergibt sich eine maximale Leistung von 300 Watt.

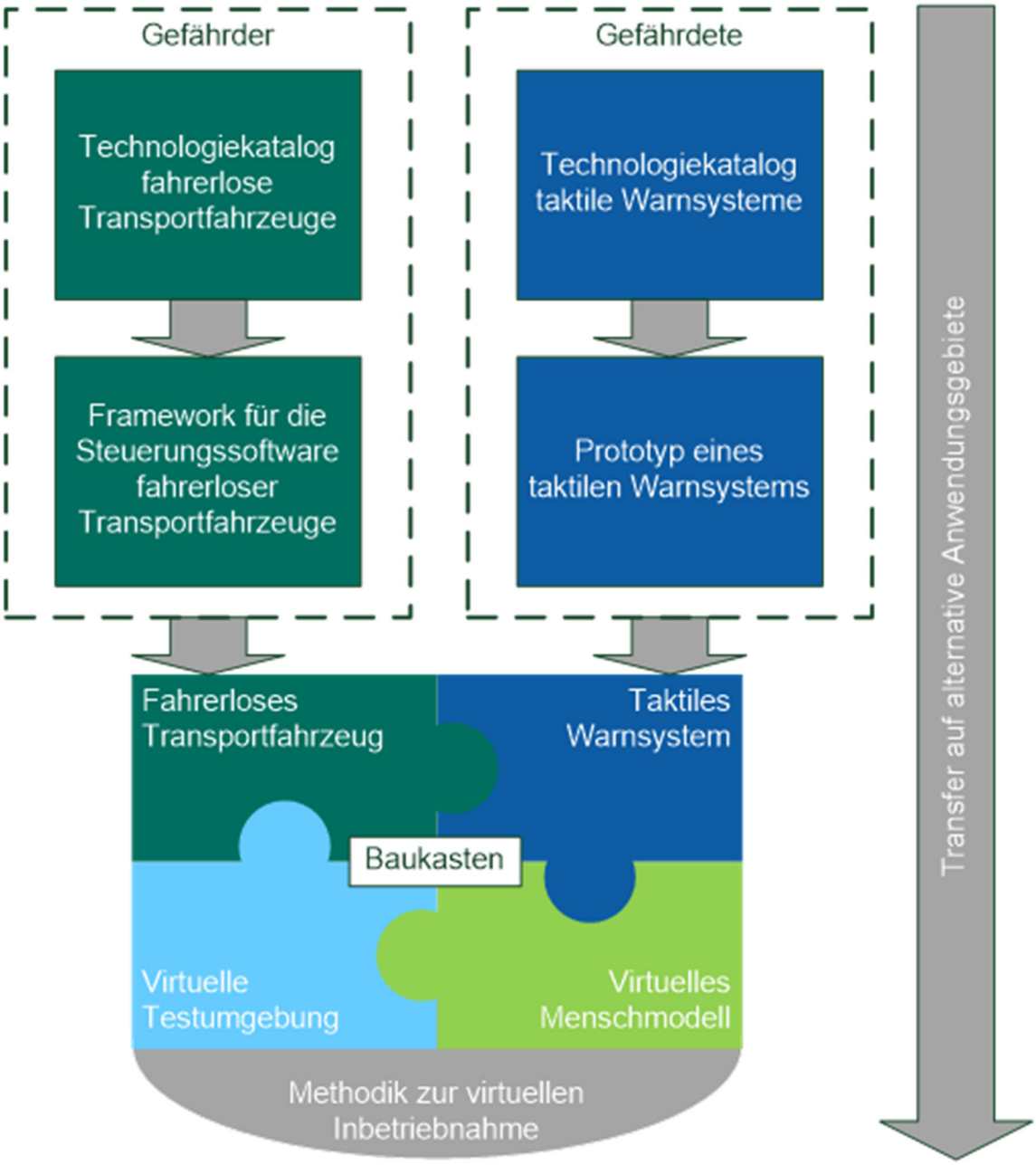


Abbildung 1: beispielhafte finale Systemstruktur,
(entnommen aus Projektantrag Professur Systemelektronik HSMW)

In der folgenden Tabelle sind einzelne technische Anforderungen aufgelistet:

Tabelle 1: Anforderungen an das System

Aktive Flächen / Taktoren	10 Stück
Abstand Taktor zu Taktor (Mittelpunkt)	>= 5 cm
Taktoranordnung	sinnvoll für räumliche Wahrnehmung
Maximale Leistung	300 W
Reaktionszeit	<= 25 ms
Gewicht	<= 1 kg

Aus der Aufgabenstellung ergeben sich zwei Kernsysteme:

- Das Detektionssystem, welches mit Hilfe von Sensorik den Raum aufnimmt, auswertet und Gefahren erkennt
- Das Aktorsystem, welches als Interface Informationen des Warnsystems an den Werkarbeiter übermittelt

2 Stand der Technik

Das Detektionssystem soll mit Hilfe von mehreren Sensorsystemen den Raum erfassen und auswerten. Hierbei spielt die Reaktionszeit eine entscheidende Rolle. Die Menge an Tiefeninformationen, welche in Echtzeit ausgewertet werden sollen, benötigt erhebliche Rechenleistung. Der Mini-PC und verschiedene Sensorsysteme werden in den folgenden Unterpunkten kurz beschrieben.

Ein mobiles vibrotaktiles Aktorsystem muss mit einer mobilen Stromquelle betrieben werden. Hier spielen die Miniaturisierung der Bauteile und die Effizienz in Bezug auf den Stromverbrauch eine entscheidende Rolle. Ein Mikrocontroller erfüllt diese Anforderungen und wird im folgenden Unterpunkt kurz beschrieben. Weiterhin werden die Aktorik sowie bestehende Aktorsysteme und Forschungen zur menschlichen Haut aufgeführt.

2.1 Rechnersystem

„Rechnersystem, die Gesamtheit der Bauelemente einer digitalen Datenverarbeitungsanlage und des dazugehörigen Betriebssystems. Die Art und Kombination der einzelnen Komponenten bestimmt die hardwareseitige Eignung des Rechnersystems für verschiedene Datenverarbeitungsbereiche (Serverrechner, graphische Workstation).“
(Martin, Spektrum.de - Rechnersysteme, 2000)

In dieser Arbeit sollen Mikrocontroller und Mini-PC Einsatz finden, welche folgend kurz erläutert werden.

2.1.1 Mikrocontroller

„Mikrocontroller sind leistungsfähige, kompakte, programmierbare Rechnersysteme. Diese enthalten einen Prozessor und alle benötigten Bausteine wie Speicher, Zeitgeber, digitale / analoge Ein- und Ausgabegeräte, usw. Alle Komponenten sind dabei auf einem Chip, deshalb bezeichnet man den Mikrocontroller auch als Ein-Chip-Mikrorechner.“ (Laser & Co. Solutions GmbH, 2005)

Diese Mikrorechner sind im Vergleich zu einem üblichen PC platzsparend und können aufgrund ihrer Bauform (Chip) in das eigene PCB integriert werden, sind energieeffizient und gut für einfache Anwendungen geeignet, hier: Ansteuerung von Aktoren. Die Anzahl an Berechnungen, die von einem Mikrocontroller in einer Zeiteinheit durchgeführt werden kann, ist von seiner Taktfrequenz abhängig. Dabei gibt es Mikrocontroller mit einigen wenigen Kilohertz für einfache Berechnungen, Einlesen von Sensoren und Ausgaben über LEDs o.Ä., bis in den Gigahertzbereich für Multitask-Anwendungen, Ausgaben auf RGB-Displays

bzw. High-Speed-Schnittstellen. Dabei variiert die Energieaufnahme des Mikrocontrollers zwischen Nanoampere und Milliampere und dessen Größe zwischen Millimeter und Zentimeter. Die Auswahl eines geeigneten Mikrocontrollers ist in Kapitel 5.1 zu finden.

2.1.2 Mini-PC

Im Vergleich zum üblichen Personal Computer besitzt der Mini-PC einen kleineren Formfaktor (Tabelle 2) und ist somit für mobile Einsatzbereiche geeignet. Mini-PCs werden in der Regel mit Gleichspannung kleiner 30 Volt betrieben und benötigen ein externes Netzteil für die Speisung über das lokale Stromnetz. Durch die Miniaturisierung und damit begrenzte Möglichkeit der Kühlung, ist die Rechenpower mit einem leistungsstarken Laptop vergleichbar.

Tabelle 2: Mini-PC Abmessungen

Höhe:	10 cm bis 35 cm
Breite:	10 cm bis 40 cm
Tiefe:	3 cm bis 4 cm

2.2 Sensorsysteme

„Ein Sensor dient zur quantitativen und qualitativen Messung von physikalischen, chemischen, klimatischen, biologischen und medizinischen Größen. Der Sensor besteht aus zwei Teilen: dem Sensor-Element und der Auswerte-Elektronik. Die zu messenden, nicht elektrische Eingangsgrößen werden im Sensor-Element durch naturwissenschaftliche Gesetze in ein elektrisches Ausgangs-Signal gewandelt. In einer Auswerte-Elektronik werden diese Ausgangssignale durch Schaltungselektronik oder auch Softwareprogramme so bearbeitet, dass ein Sensor-Ausgangssignal entsteht, das zu Steuerungs- oder Auswertezwecken zur Verfügung steht.“ (E. Hering, 2012)

Zur Detektion von Objekten müssen Tiefeninformationen aus der unmittelbaren Umgebung aufgenommen werden. Drei verschiedene Ansätze dafür sind Radar, Ultraschall und Time-of-Flight, welche in den folgenden Unterpunkten kurz beschrieben werden. Im Bereich des autonomen Fahrens in der Automobilbranche werden diese Konzepte bereits praktisch eingesetzt. Ein kurzer Überblick an bestehenden Systemen ist im Unterpunkt 2.2.4 Systeme zur Objektdetektion zu finden.

2.2.1 Radar

Über einen Sender werden elektromagnetische Wellen je nach Anwendung im Bereich von 30 Megahertz bis etwa 300 Gigahertz ausgesendet. (München, 2010) Durch Reflexion an Objekten werden die auftreffenden Wellen in Richtung des Einfallswinkels zurückgeworfen und ein Empfänger liest das rückgeworfene Signal ein. Zu unterscheiden sind:

- Pulsradargeräte, welche einen Impuls mit wenigen Mikrosekunden Dauer aussenden und auf das Echo warten und
- Dauerstrich-Radargeräte, welche kontinuierlich ein Signal aussenden und durch Differenzbildung mit dem zurückgeworfenen Signal Rückschlüsse auf den Raum zulassen (Wikipedia, Radar, 2021)

Bei einem Pulsradargerät wird ein Impuls mit einer Dauer im Mikrosekundenbereich ausgesendet. Es wird die Zeit zwischen Aussenden des Signals und dem Echo gemessen. Mit folgender Formel (Formel (1)) kann die Entfernung zum reflektierenden Objekt berechnet werden:

$$s = \frac{c_{Luft}}{2} \cdot \Delta t \qquad (1)$$

s – Entfernung zum Objekt, c_{Luft} – Lichtgeschwindigkeit in der Luft ($2{,}9971 \cdot 10^8 \frac{m}{s}$),
Δt – Laufzeit des Signals

Bei einem modulierten Dauerstrich-Radargerät (FMCW) wird kontinuierlich ein Signal mit periodisch steigender bzw. fallender Frequenz ausgesendet. Durch den Dopplereffekt und einer FFT kann hier die Geschwindigkeit, der Winkel und die Entfernung mehrerer Objekte bestimmt werden. (Wolff, 1977)

Die folgende Abbildung (Abbildung 2) zeigt ein ausgesendetes und empfangenes Radarsignal:

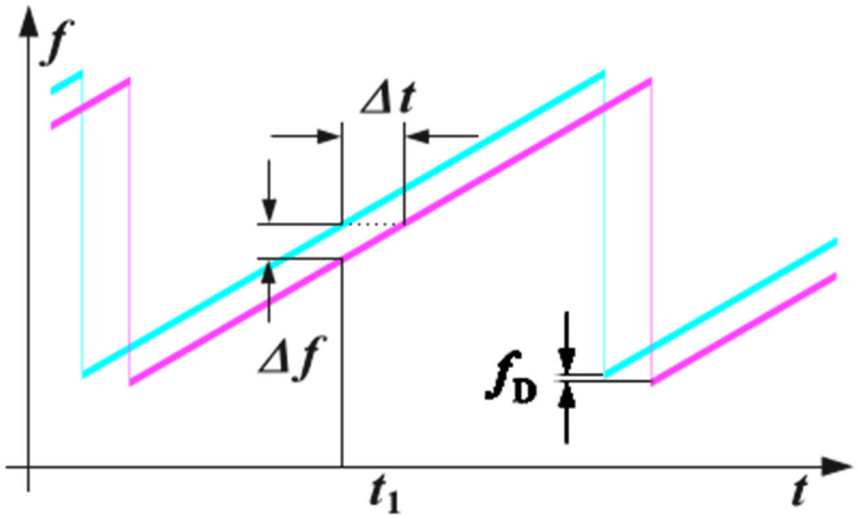


Abbildung 2: FMCW - Radar, Signalverlauf (Wolff, 1977)

Hier türkis dargestellt das ausgesendete Signal. Das empfangene Signal ist um Δt verschoben und entspricht der Laufzeit des Signals, mit obiger Formel (Formel (1)) kann somit die Entfernung berechnet werden. Weiterhin ist das Signal aufgrund der kontinuierlich steigenden Frequenz nach einer Zeit Δt auch um eine Frequenz Δf verschoben. Eine zusätzliche Frequenzverschiebung um Δf_D kommt durch ein bewegtes Objekt zustande und entspricht der Dopplerfrequenz.

2.2.2 Ultraschall

Bei der Ultraschallmessungen wird eine Schallwelle weniger Mikrosekunden mit einer Frequenz im Bereich von 20kHz bis 10 GHz (Wikipedia, Ultraschall, 2021) ausgesendet und an einem Objekt im Detektionsbereich reflektiert. Mit der Laufzeit des Signals und der Schallgeschwindigkeit in der Luft kann die resultierende Entfernung berechnet werden. Siehe Formel (2) zur Berechnung der Entfernung:

$$s = \frac{1}{2} \cdot \Delta t \cdot cs_{Luft} \tag{2}$$

s – Entfernung zum Objekt, cs_{Luft} – Schallgeschwindigkeit in der Luft ($343{,}2\ \frac{m}{s}$), Δt – Laufzeit des Signals

2.2.3 Time-of-Flight

Ein Photomischdetektor (PMD) sendet einen Lichtimpuls aus. Es wird anschließend die Signallaufzeit zum angeleuchteten Messobjekt und damit die Entfernung vom Sensor zum Objekt berechnet. Ein Array der Photomischdetektoren ergibt ein 2-dimensionales Bild mit

Tiefeninformationen. Die grundlegenden Komponenten einer ToF-Kamera sind folgend aufgeführt und kurz beschrieben (Wikipedia, TOF-Kamera, 2021):

- Beleuchtungseinheit: LEDs oder Laserdioden senden einen Lichtstrahl im nahen Infrarotbereich (780 nm bis 3 µm) mit einer Pulsdauer im Nanosekundenbereich aus
- Optik: Das reflektierte Lichtsignal wird gefiltert und auf dem Sensor abgebildet
- Sensor: Mit dem empfangenen Lichtsignal wird die Laufzeit gemessen
- Ansteuerelektronik: ist zur Steuerung der Beleuchtungseinheit und des Sensors notwendig, muss sehr präzise arbeiten, damit die Laufzeit genau berechnet werden kann (eine Abweichung von 10 ps ergibt eine Entfernungsänderung von 1,5 mm)
- Auswertung/Interface: die Distanz wird aus den Laufzeiten berechnet und an ein System über Ethernet oder USB übertragen

Mit folgender Formel (Formel (3)) wird die Entfernung berechnet:

$$s = \frac{c_{Luft} \cdot \Delta t}{2} \tag{3}$$

s – Entfernung zum Objekt, c_{Luft} – Lichtgeschwindigkeit in der Luft ($2{,}9971 \cdot 10^8 \frac{m}{s}$),

Δt – Laufzeit des Signals

Die maximale Entfernung ist von der Beleuchtungsdauer abhängig und in folgender Formel (Formel (4)) dargestellt:

$$t_0 = \frac{s_{max} \cdot 2}{c_{Luft}} \tag{4}$$

t_0 – Beleuchtungsdauer, c_{Luft} – Lichtgeschwindigkeit in der Luft ($2{,}9971 \cdot 10^8 \frac{m}{s}$),

s_{max} – maximale Entfernung zum Objekt

2.2.4 Systeme mit Objektdetektion

In den Bereichen des autonomen Fahrens und der Fahrsicherheit der Automobilbranche finden Sensoren zur Entfernungsbestimmung und zur Objektbestimmung schon seit über 10 Jahren teilweise in der Serienproduktion Einsatz. Die Sensoren arbeiten mit akustischen (Ultraschall) und elektromagnetischen Wellen.
(Radar, Lidar, ToF, Mono-Video, Stereo-Video)

Folgend werden einige Systeme vorgestellt (Speth, 2009):

Adaptive Cruise Control (ACC) ist ein Tempomat, bei dem eine Zielgeschwindigkeit eingestellt wird. Das Auto fährt nach Aktivierung maximal die Zielgeschwindigkeit. Befindet sich ein Objekt vor dem Fahrzeug, wird die Geschwindigkeit gedrosselt und entsprechend angepasst. Hier kommen Radar, Lidar und Video zum Einsatz, um Objekte und deren Geschwindigkeit zu erkennen. Auch der Bremsassistent greift auf das ACC zurück, um schnell näherkommende Objekt zu identifizieren, damit präemptiv schneller der maximale Bremsdruck für eine Gefahrenbremsung aufgebaut wird.
Der **Einparkassistent** nutzt Ultraschall, Radar und Video. Je nach Fahrzeug gibt der Assistent eine akustische oder optische Warnung aus, um das Einparken zu erleichtern. Dabei geben die Signale den Abstand zu erkannten Objekten an. Neuere Assistenten parken teilautonom bis autonom ein.
Der **Spurwechselassistent** warnt beim Setzen des Blinkers durch Vibration am Lenkrad oder optisch, wenn ein Spurwechsel aufgrund eines Hindernisses nicht möglich ist. Dabei ist Radar für die Abtastung seitlich und hinter dem Fahrzeug zuständig.
Der **Spurhalteassistent** funktioniert mit Videodaten und ist aktiv, solange kein Blinker gesetzt ist. Passt der Fahrtwinkel nicht zur Fahrspur, wird am Lenkrad eine Vibration ausgelöst, um den Fahrer zu warnen, damit dieser das Fahrzeug in der Spur hält.
Der **Fernlichtassistent** nutzt ebenfalls Video, um Objekte nach Modellen einzugliedern. Zum Beispiel wird beim Erkennen eines Autos das Fernlicht am unteren Rand des Kennzeichens ausgerichtet, damit der Fahrer nicht geblendet wird.

Entscheidend ist bei allen Systemen die Sensorik. Ultraschall liefert nur die Entfernung zum nächsten Objekt und wird daher nur im Einparkassistent eingesetzt. Radar liefert die Entfernung, den Winkel und die Geschwindigkeit mehrerer Objekte im Detektionsbereich, wobei elektrisch leitfähiges Material exakter erkannt wird als isolierende Materialien. Da bei dem ACC, dem Spurwechselassistent sowie bei dem Einparkassistenten andere Kraftfahrzeuge erkannt werden müssen, ist Radar gut für diesen Verwendungszweck geeignet. Durch Modellbildungen können aus Videodaten Objekte identifiziert werden und sind z.B. für den Fernlichtassistenten notwendig. Durch Modelle kann auch ein Mensch von einem Fahrzeug unterschieden werden, aber nur bei der Erfüllung aller Voraussetzungen und programmierten vorhersehbaren Szenarien. Da der Straßenverkehr nicht vorhersehbar ist, nehmen die Systeme dem Fahrer keine Verantwortung ab.

Lidar, ToF und Stereo-Video liefern Tiefeninformationen für die jeweiligen Abtastpunkte und unterstützen die Fahrassistenzsysteme. Aus diesen 2D- bzw. 3D-Informationen lassen sich Objekte ohne Modellbildung clustern und Informationen zur Lage und Geschwindigkeit bestimmen, welche für die Sicherheit und das autonome Fahren notwendig sind.

Auch in dieser Arbeit sollen annähernde Objekte ohne Modellbildung erkannt werden, weshalb hier Radar, ToF und Ultraschall Einsatz finden werden.

2.3 Aktorik

„Die Gesamtheit der haptischen Wahrnehmungen erlaubt es dem Gehirn, mechanische Reize, Temperaturreize und Schmerz zu lokalisieren und zu bewerten. Die Lehre von der haptischen Wahrnehmung wird als Haptik bezeichnet."
(Berlin-Brandenburgische Akademie der Wissenschaften, 2020)

Dieses haptische Wahrnehmen soll hier verwendet werden, um mit einer ansteuerbaren Aktorik Reize zu schaffen. Dabei lassen sich mechanische Reize in Form einer Vibration mit einer Spannung durch den inversen Piezoeffekt oder mit einer Unwucht an einem laufenden Motor, auch Vibrationsmotor genannt, erzeugen. In dieser Arbeit sollen die Vibrationsmotoren Verwendung finden.
DC-Motoren mit einer Unwucht finden heutzutage unter anderem im Smartphone Anwendung und erzeugen eine Vibration z.B. bei dem Empfangen einer Nachricht oder bei anderen Aktionen. Sie besitzen in diesem Anwendungsbereich eine Größe von mehreren Millimetern bis hin zu einigen Zentimetern. Die Intensität der Vibration ist dabei abhängig von der Größe der Unwucht und der Geschwindigkeit des Motors. Ein passender Vibrationsmotor wird im Kapitel 5.2.1.1 ausgewählt.

2.3.1 Sinnesorgan Haut

Drei Sinneseindrücke können der Haut zugeschrieben werden (Schmidt, 2007):

- Tastsinn – Mechanorezeption genannt
- Temperatur – Thermorezeption genannt
- Schmerz – Nozizeption genannt

Ein vibrotaktiles Feedback bzw. eine Vibration lässt sich der Mechanorezeption zuordnen und wird folgend näher beschrieben. Abbildung 3 zeigt den Aufbau der Haut mit den dazugehörigen Nervenzellen.

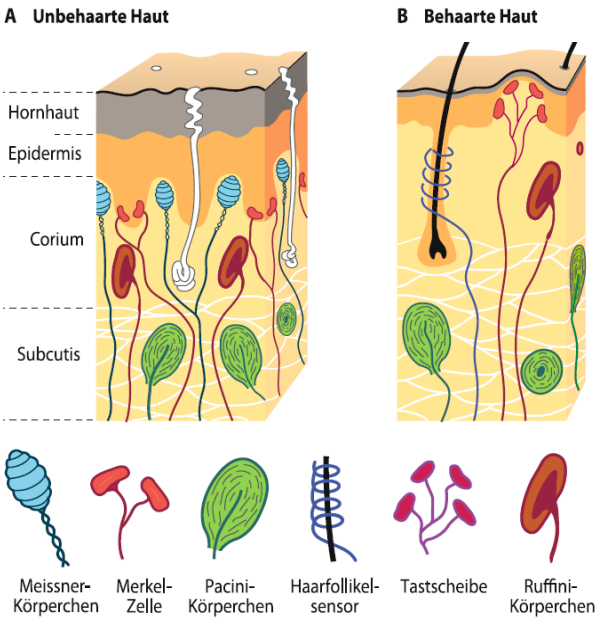


Abbildung 3: Histologie der Mechanorezeptoren der Haut (Schmidt, 2007)

Unbehaarte Haut ist an den Handflächen, Fußsohlen und Lippen zu finden, behaarte Haut an den übrigen Körperflächen.

Die Tastscheiben, Merkel-Zellen und Ruffini-Körperchen dienen der Detektion von Druckintensitäten sowie Druckänderungen und sind verantwortlich für die Tastempfindung. Es handelt sich um langsam anpassende Dehnungsrezeptoren. Zum Beispiel wird getragene Kleidung bei der Bewegung durch die differenzial-proportionalen Fühler direkter wahrgenommen.

Meissner-Körperchen und Haarfollikel-Sensoren nehmen die Geschwindigkeit einer Bewegung wahr. Je höher die Geschwindigkeit ist, desto höher ist der ausgesendete Nervenimpuls.

Pacini-Körperchen sind Beschleunigungsdetektoren und senden nur während einer Geschwindigkeitsänderung einen Impuls aus. Diese dienen zum Beispiel einer Vibrationsdetektion, wobei Vibrationen im Bereich zwischen 100 und 300 Hz besonders gut wahrgenommen werden.

Die simultane Raumschwelle der Rezeptoren wird geprüft, indem zwei Tastspitzen gleichzeitig auf die Haut gesetzt werden. Gesucht wird hier der minimale Abstand zwischen den zwei Spitzen, bei denen die beiden Reizpunkte gerade noch als getrennt wahrgenommen werden können. Weiterhin gibt es die Empfindungsschwelle, hiermit ist die minimale Eindrucktiefe für eine Reizauslösung gemeint. Diese liegt in der Handinnenfläche bei 5-10 µm.

Die Eindrucktiefe kann proportional zur simultanen Raumschwelle und proportional zur Dichte der Nervenzellen der Mechanorezeption angenommen werden. (Birbaumer, 1999) Abbildung 4 zeigt die simultane Raumschwelle (Zwei-Punkt-Schwelle) eines erwachsenen Körpers.

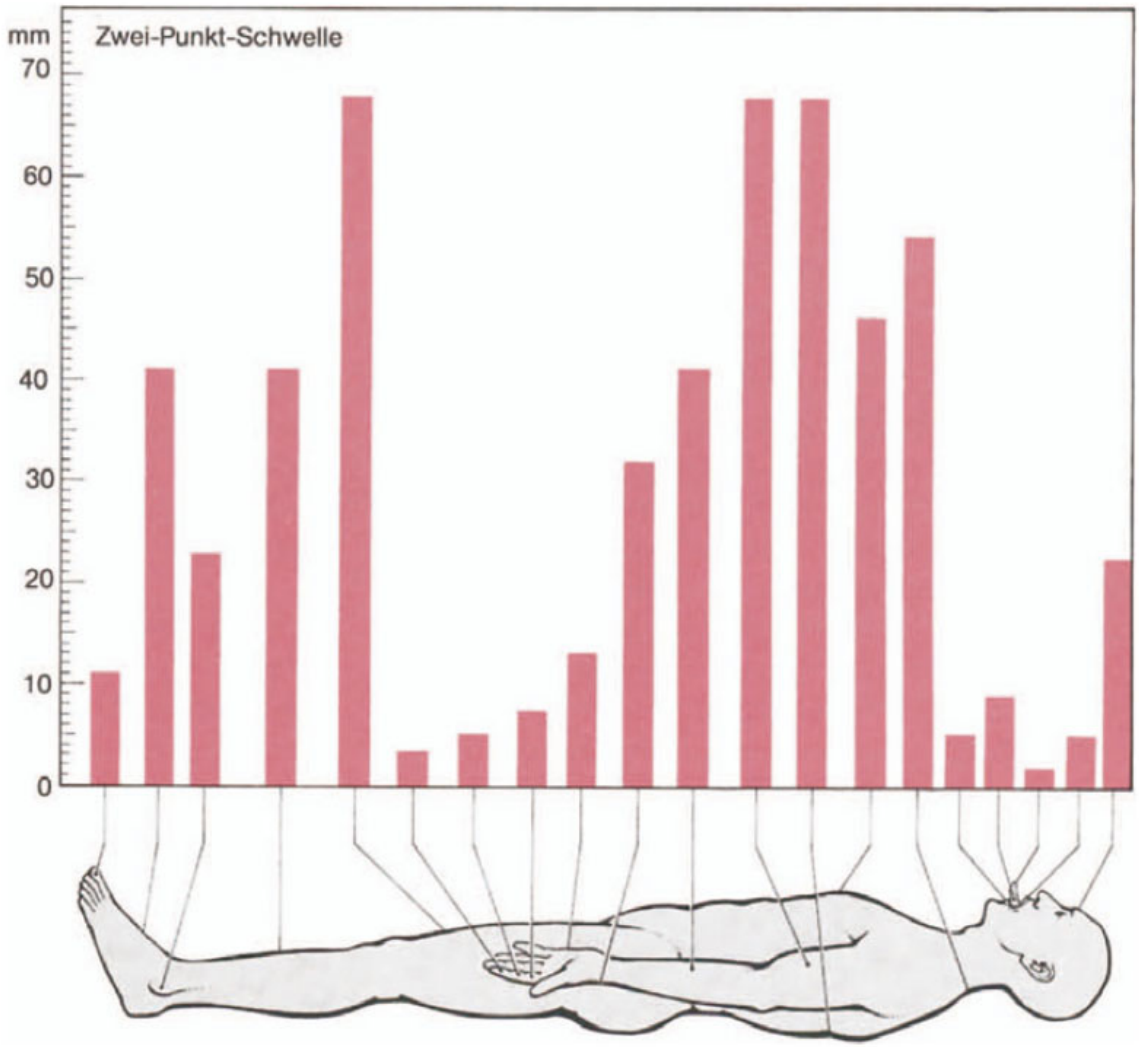


Abbildung 4: Simultane Raumschwelle des Erwachsenen (Birbaumer, 1999)

2.3.2 Systeme mit vibrotaktilen Aktoren

Vibrotaktile Systeme als Schnittstelle zum Menschen finden schon in den verschiedensten Formen Anwendung. Folgend werden einige dieser Systeme kurz vorgestellt.

„VibroTac“ wurde von dem deutschen Zentrum für Luft- und Raumfahrt entwickelt und ist ein vibrotaktiles Armband mit 6 Vibrationsmotoren zur Interaktion mit dem Menschen. Dabei lassen sich alle 6 Aktoren unabhängig voneinander in Frequenz, Amplitude und Signalverlauf ansteuern. Zusätzlich bietet dieses Armband 9 Anschlüsse für externe Aktoren, welche ebenfalls angesteuert werden können. Gesteuert wird das Armband über WiFi und ist für Anwendungen wie translatorische bzw. rotatorische Bewegungshinweise des Arms in Instandsetzungsszenarien entwickelt worden. (Simon Schätzle, 2010)

Lenkradvibrationen werden in der Promotionsarbeit von Frank Beruscha zuerst in 3 Studien unterteilt - Kollisionswarnsystem, Spurverlassenswarnsystem und örtliche Reiz-Reaktions-Kompatibilität - und anschließend näher betrachtet im Hinblick auf Ort, Lage, Richtung, Art, Intensität und Frequenz der Vibration am Lenkrad, um die 3 Systeme zum einen auseinander halten zu können und zum anderen Warn- und auch Richtungsinformationen der Vibration entnehmen zu können. (Beruscha, 2012)

Ein taktiles Warnsystem in Form von 4 Vibrationsmotoren, angebracht an einem Gürtel – in jede Himmelsrichtung einer – dient im Umfeld von autonomen Fabriken zum Anzeigen eines Verstoßes beim Übertreten eines Sicherheitsbereiches. Dabei vibriert einer bzw. vibrieren zwei der Taktoren, um die Lage der überschrittenen Sicherheitszone aus Sicht des Arbeiters, welcher den Gürtel trägt, bestimmen zu können. Die Position wird über ein Sensorsystem erfasst, ausgewertet und dann drahtlos an den Gürtel gesendet. Die Vibration endet, sobald der Arbeiter die „verbotene“ Zone wieder verlassen hat. (Mohr, 2019)

Vibrotaktile Displays finden im Bereich „Industrie 4.0“ immer häufiger Anwendung, um die Eingabe per Touch auch mit Handschuh zu ermöglichen bzw. zu vereinfachen. Dabei soll der Effekt eines Tastendrucks wie bei einer mechanischen Tastatur simuliert werden, um eine schnellere Eingabe von Texten sowie eine geringere Fehlerzahl bei Eingaben zu ermöglichen. Eine Studie mit 47 Versuchsteilnehmern und 8265 Einzelmessungen untersucht dabei die Frequenz, die Stimulusdauer und die Handschuhstärke in Bezug auf die Wahrnehmungsschwelle des Zeigefingers des Probanden. (Martin Seeger, 2017)

Melina Brell entwickelt in ihrer Dissertation eine „vibrotaktile Mensch-Maschine-Schnittstelle für chirurgische Applikationen“. Bislang wurden zum Navigieren bei Operationen in der Chirurgie optische Navigationssysteme in Form von Displays verwendet, welche mit Hilfe von voraufgenommenen Bildern und Positionssensoren bzw. Live-Sonden die Position der verwendeten Instrumente und das Ziel anzeigen. Das wiederum führt zum Blickkontaktverlust des Chirurgen zum Patienten und verkompliziert den Eingriff. Ein vibrotaktiler Aktorhandschuh soll hier als Ersatz-Navigationssystem dienen, indem durch gezielte Vibrationen auf dem Handrücken die Position, in welche das Instrument bei einer Operation bewegt werden soll, dem Chirurgen übermittelt wird. (Brell, 2009)

3 Präzisierung der Vorgehensweise

Für die Entwicklung des Warnsystems sind zunächst die beiden Teilsysteme – Detektionssystem und Aktorsystem – getrennt zu betrachten.

Für das Detektionssystem sind geeignete Sensorsysteme auszuwählen. Anschließend muss hier eine Analyse der Sensoreigenschaften stattfinden. Hierzu soll es Messungen zur Untersuchung des Sensorwinkels, des Objektwinkels, der Entfernung und der Materialabhängigkeit geben. Die Sensoren sind nach dem Auswahlverfahren und den Messungen so anzuordnen, dass ein großer Aufnahmebereich entsteht. Das Detektionssystem muss alle Sensordaten innerhalb der Reaktionszeit einlesen und auswerten, um auf näherkommende Objekte schließen zu können. Dabei muss die Geschwindigkeit und die Entfernung dieser Objekte Bestandteil einer Gefahrenanalyse sein, welche bei Überschreiten einer zu definierenden Schwelle Warnbefehle an das Aktorsystem sendet.

Für das Aktorsystem sind geeignete vibrotaktile Aktoren auszuwählen. Anschließend müssen zur Unterscheidbarkeit von Richtungen die Ansteuerung und die Anzahl bzw. Anordnung der Aktoren festgelegt werden. Hierbei spielt die Intensität und Frequenz der Vibration zur Wahrnehmbarkeit und zur Unterscheidung von Gefahrenstufen eine entscheidende Rolle.

Es soll ein Datenaustausch zwischen den Teilsystem möglich sein. Dazu ist ein Kommunikationsprotokoll sowie eine Schnittstelle für die beiden Teilsysteme zur Übertragung von Richtungsinformationen und Gefahrenstufen vom Detektionssystem zum Aktorsystem zu entwerfen.

Anschließend folgt die Umsetzung in Form eines Prototyps und eine Inbetriebnahme. Tests der Teilsysteme, sowie ein Systemtest sollen die Funktion des Warnsystems qualitativ überprüfen. Die Entwicklungsergebnisse werden dokumentiert, ausgewertet und zusammengefasst.

Mögliche Verbesserungsvorschläge sowie Ausbaumöglichkeiten sollen in einem Ausblick vorgestellt werden.

4 Konzeption

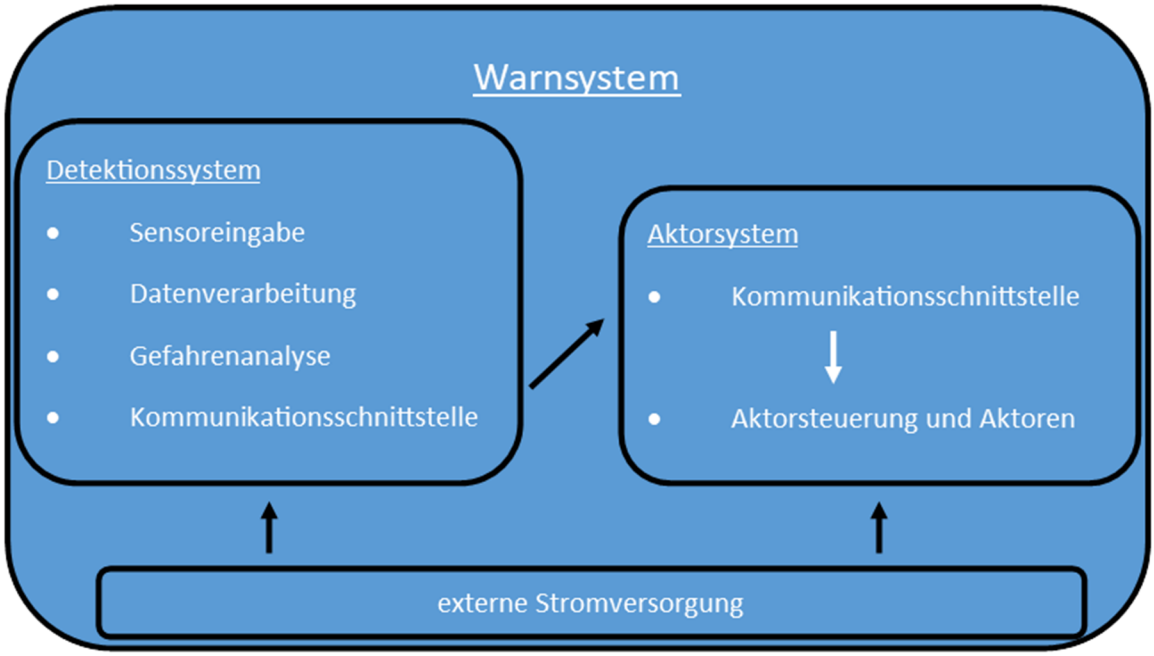


Abbildung 5: Systemkonzept

Abbildung 5 visualisiert das Systemkonzept des Warnsystems.
Das Warnsystem besteht aus den Bausteinen Detektionssystem und Aktorsystem, welche über eine externe Spannungsversorgung betrieben werden.

Das Detektionssystem umfasst alle Sensoren zur Aufnahme des Raumes. Durch eine Datenverarbeitung sollen Objekte im Raum ermittelt und in eine Gefahrenanalyse einbezogen werden. Wenn ein Objekt aufgrund seiner Lage und Geschwindigkeit als potenzielles Risiko eingestuft wird, werden Befehle über die Kommunikationsschnittstelle an das Aktorsystem gesendet.

Das Aktorsystem empfängt über die Kommunikationsschnittstelle die Befehle zur Ansteuerung der Aktoren. Die Befehle enthalten Richtungsinformationen sowie Intensitätswerte, Tempi und Muster für eine räumlichen Wahrnehmung des Risikos, welche die Aktoren entsprechend zum Vibrieren bringen.

Eine externe Stromversorgung muss gefunden und entsprechend dimensioniert werden, sodass alle Bausteine des Warnsystems betrieben werden können.

Es folgt die Konzeption des Detektionssystems und des Aktorsystems jeweils in Hardware und Software.

4.1 Detektionssystem

In diesem Kapitel werden das Hardware- und Softwarekonzept des Detektionssystems näher beschrieben.

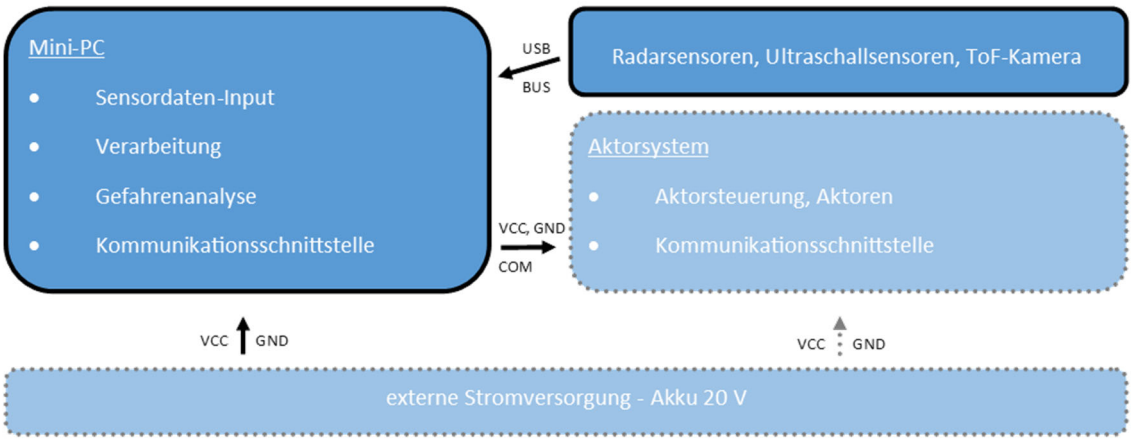


Abbildung 6: Hardwarekonzept des Detektionssystems

Abbildung 6 zeigt das Hardwarekonzept des Detektionssystems. Radar- und Ultraschallsensoren sowie eine ToF-Kamera sollen genutzt werden, um Tiefeninformationen zu gewinnen. Die Übertragung der Sensordaten sowie die Spannungsversorgung der Sensoren an die Verarbeitungseinheit soll über USB erfolgen. Für die Auswertung in Echtzeit wird aufgrund der enormen Datenmenge ein Mini-PC mit entsprechender Rechenleistung ausgewählt. Bei erkannter Gefahr wird über die serielle Kommunikationsschnittstelle ein Befehl zur Vibration an das Aktorsystem gesendet.
Der Mini-PC wird mit 20 Volt betrieben, eine Konzeption der Kommunikationsschnittstelle und der Spannungsversorgung ist im Kapitel 4.2 Aktorsystem zu finden.

Das Softwarekonzept des Detektionssystems ist in Abbildung 7 dargestellt.

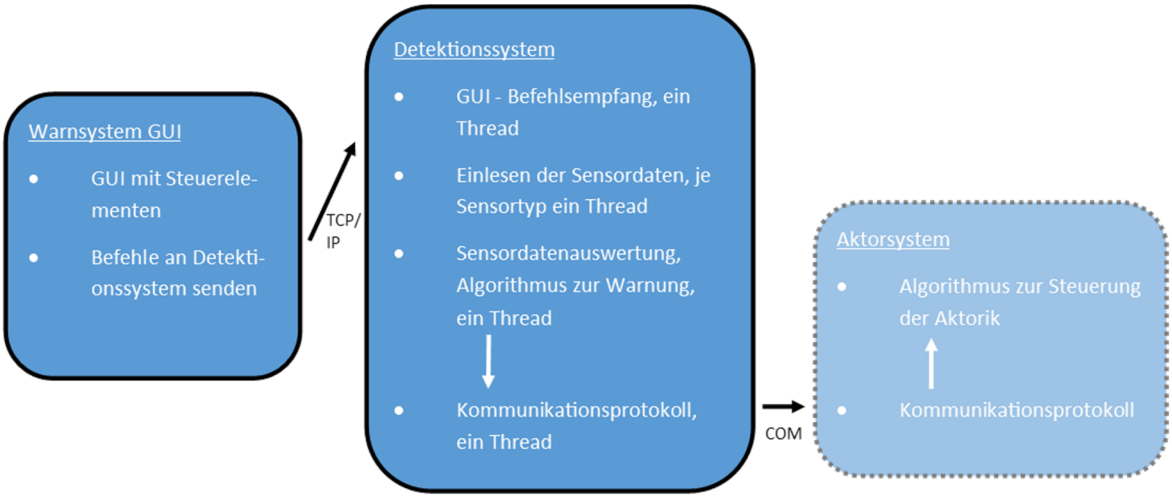


Abbildung 7: Softwarekonzept des Detektionssystems

Das Detektionssystem muss in Echtzeit die Sensordaten einlesen, diese verarbeiten, eine Gefahrenanalyse durchführen und gegebenenfalls eine Warnung in Form eines Befehls an das Aktorsystem weitersenden. Ein Polling-Loop zur Bewältigung dieser Schritte nacheinander würde zu viel Zeit in Anspruch nehmen, weshalb hier ein Betriebssystem Einsatz finden wird:

Je Sensortyp wird es einen Thread geben, welche unabhängig voneinander Daten einlesen. Synchron dazu läuft der Thread für die Datenverarbeitung, Auswertung und Gefahrenanalyse. Er greift dabei auf die Daten des Sensor-Threads zurück. Ein weiterer Thread ist für das Senden der Befehle über die serielle Schnittstelle vorgesehen und bezieht seine Daten vom Auswertungs-Thread.

Das System soll von einem externen Gerät aus steuerbar sein. Dabei müssen einzelne Parameter zu den Sensoren und der Gefahreneinstufung während des Betriebs abänderbar sein. Eine GUI auf dem externen Gerät soll über W-LAN durch das TCP/IP-Protokoll mit dem Detektionssystem verbunden werden und diese Funktionen ermöglichen.

4.2 Aktorsystem

In diesem Kapitel werden das Hardware- und Softwarekonzept des Aktorsystems erläutert.

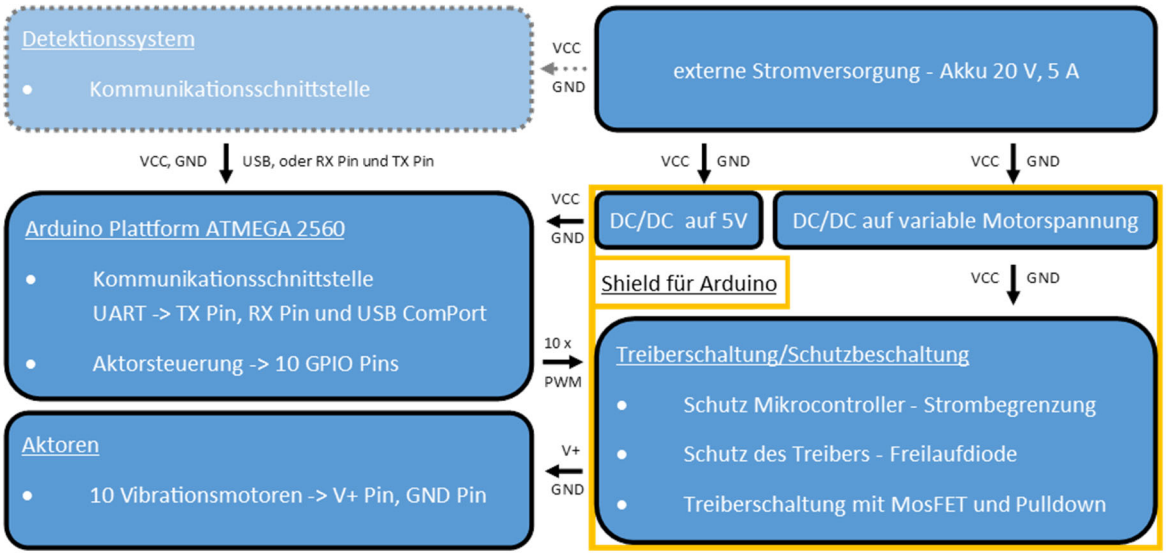


Abbildung 8: Hardwarekonzept des Aktorsystems

In der Abbildung 8 ist das Hardwarekonzept des Aktorsystems dargestellt. Die mobile externe Spannungsversorgung muss das Detektionssystem und das Aktorsystem mit Strom versorgen. Das Detektionssystem benötigt eine Spannung von 20 Volt und das Aktorsystem 5 Volt sowie zusätzlich eine einstellbare Spannung für die Vibrationsmotoren, welche sich im Bereich von 2 bis 8 Volt befindet. Es wird im Kapitel 5.2.1.1 ein Akku mit 20 Volt ausgewählt. Die niedrigeren Spannungen für das Aktorsystem werden mit Hilfe von DC/DC-

Wandlern erzeugt. Der Mikrocontroller kann seine Spannungsversorgung über eine USB-Schnittstelle des Detektionssystems beziehen oder nutzt einen separaten DC/DC-Wandler.

Als Kommunikationsschnittstelle wird die serielle Kommunikation (UART) genutzt. Es gibt die Anschlussmöglichkeiten entweder über die beiden Anschluss-Pins RX und TX oder den virtuellen Com-Port über die USB-Schnittstelle. Die vom Detektionssystem empfangenen Daten zur Aktor-Ansteuerung werden im Mikrocontroller verarbeitet und führen mit Hilfe von dessen internen Timern zur Erzeugung von PWM-Signalen an den zehn Steuerpins für die Aktorik.
Durch die geringe maximale Stromstärke, die durch die GPIO-Pins fließen darf, muss hier eine Treiberschaltung und Schutzbeschaltung entworfen werden. Je Vibrationsmotor ist eine Beschaltung mit MosFET, Freilaufdiode und Widerständen nötig.

Der Mikrocontroller soll für einen prototypischen Aufbau und Test auf einer Arduino-Plattform basieren. Die zusätzliche Beschaltung sowie die DC/DC-Wandler als auch die Anschlüsse für die Spannungsversorgung, Taktoren und Kommunikation über die serielle Schnittstelle sollen in Form eines PCB-Aufsteck-Shields für den Arduino konstruiert werden. Damit die Motorspannung stabil gehalten werden kann, müssen hier Pufferkondensatoren hinter dem DC/DC-Wandler parallel zu den Taktoren Einsatz finden.

Die Arduino-Plattform ist erst nach funktionaler Programmierung betriebsbereit. Folgend wird auf das Softwarekonzept eingegangen:

Zuerst muss ein Kommunikationsprotokoll erstellt werden. Anschließend muss dieses Protokoll als Programmcode für die Arduino-Plattform umgesetzt werden. Um die Funktionen des Aktorsystems auch ohne Detektionssystem überprüfen zu können, wird eine GUI unter Windows programmiert, welche alle implementierten Funktionen testen kann.

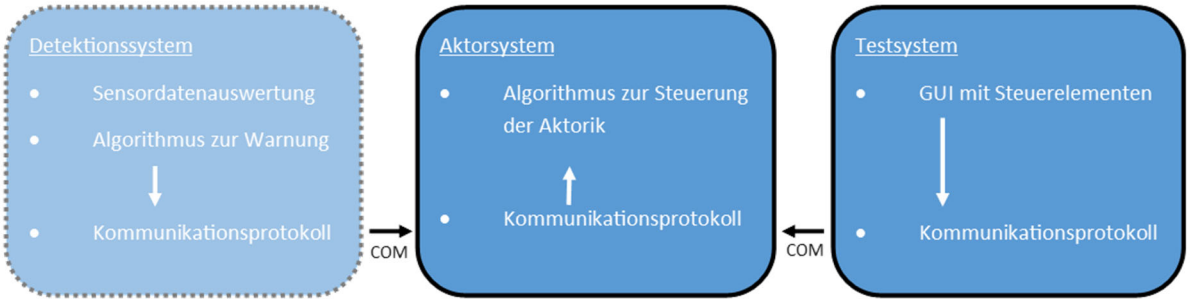


Abbildung 9: Softwarekonzept des Aktorsystems

In Abbildung 9 ist das Softwarekonzept zu sehen. Der gemeinsame Nenner zwischen dem Detektionssystem und dem Aktorsystem ist das Kommunikationsprotokoll, welches über die serielle Schnittstelle, auch Com-Port genannt, abgehandelt werden soll. Die Kommunikation findet einseitig vom Detektionssystem zum Aktorsystem statt. Um das Aktorsystem auch unabhängig testen zu können, wird ein Testsystem auf dem PC mit einer grafischen Oberfläche programmiert. Dort können durch grafische Steuerelemente alle Funktionen des Aktorsystems getestet werden. Es kann entweder das Detektionssystem oder das

Testsystem mit dem Aktorsystem verbunden werden, da die serielle Schnittstelle kein Bussystem ist.

Das Aktorsystem soll alle zehn Taktoren unabhängig voneinander mit einer PWM ansteuern können, damit die Intensität der Vibration in 255 linearen Schritten einstellbar ist. Jeder Taktor soll in sechs verschiedenen Modi/Mustern mit zehn verschiedenen Tempi vibrieren können. Es soll weiterhin die Möglichkeit geben, jeden Modus bzw. jedes Muster nach dem Startbefehl einmal oder dauerhaft bis Stoppbefehl abspielen zu lassen. Ein Notstopp soll alle Taktoren direkt ausschalten.

Folgend eine Auflistung der Funktionen pro Taktor:

- 6 Modi/Muster
- 10 Tempi
- 255 lineare Intensitätsstufen
- Einzelmodus (Muster 1x abspielen), Dauermodus (Muster bis Stopp abspielen)
- Notstopp (jeder Taktor wird ausgeschaltet bzw. der aktive Modus unterbrochen)

In der folgenden Tabelle (Tabelle 3) sind die Modi/Muster dargestellt. Jedes Muster besteht aus 16 einzelnen Schritten, entweder An (1) oder Aus (0). Jedes dieser Muster soll je nach eingestelltem Tempo zwischen 0,25 Sekunden (15,625 Millisekunden pro Schritt) und 2,5 Sekunden (156,25 Millisekunden pro Schritt) dauern.

Tabelle 3: Musterdarstellung

Takt / Muster	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1 Akzent	1	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
2 Herzschlag	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
3 Hinweis	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
4 Dauerhaft	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5 Staccato	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	0
6 Symphonie	1	0	1	0	1	0	1	1	1	1	1	1	0	0	0	0

Kommunikationsprotokoll:

Für einen Datentransfer sollen zwei Authentifizierungsbytes gesendet werden, zuerst 0xF5 und danach 0x53. Nur wenn diese zwei Bytes gesendet werden, soll der Empfänger auf Folgebytes reagieren. Nach erfolgreicher Authentifizierung soll jeweils ein Befehlsbyte und folgend die benötigte Anzahl an Informationsbytes gesendet werden. Die folgende Tabelle (Tabelle 4) zeigt die möglichen Byteketten, nachdem 0xF5 und 0x53 gesendet wurde.

Tabelle 4: Befehle des Kommunikationsprotokolls

Byte / Befehl	3	4	5
Muster	0x00	0x40 bis 0x45 für Muster 1 bis 6	-
Geschwindigkeit	0x01	0x30 bis 0x39 für 10 Geschwindigkeiten	-
Intensität	0x02	0x00 bis 0xFF für Intensität	-
Dauermodus	0x03	0x00 für deaktiviert, 0x01 für aktiviert	-
Start/Stopp	0x04	0x00 für Stopp, 0x01 für Start	-
Aktorauswahl	0x06	HIGH BYTE	LOW BYTE

Bei der Aktorauswahl müssen nach dem Befehl 0x06 zwei Bytes gesendet werden. Die Taktoren können in einer 10-Stellen-Bitmaske ausgewählt werden. Dabei bedeutet ein gesetztes Bit „ausgewählt“ und eine Null „nicht ausgewählt“. Da für 10 Taktoren 10 Bit benötigt werden und ein Byte nur 8 Bit besitzt, werden zwei Bytes gesendet. In der Tabelle 5 ist die Aufteilung der Taktoren auf die Bitmaske sowie die Unterteilung in HIGH BYTE und LOW BYTE zu sehen. Die Einstellungen werden für die jeweils ausgewählten Taktoren vorgenommen.

Tabelle 5: Aktorbelegung

Byte	**HIGH BYTE - Aktor 9 und 10**								**LOW BYTE - Aktor 1 bis 8**							
Bitmaske	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Aktor	-	-	-	-	-	-	10	9	8	7	6	5	4	3	2	1

Beispiele:
Für die Auswahl der Taktoren drei und sechs muss folgende Bytekette gesendet werden:
0xF5, 0x53, 0x06, 0x00, 0x24

Für die Geschwindigkeit drei muss folgende Bytekette gesendet werden:
0xF5, 0x53, 0x01, 0x32

5 Systementwurf des Warnsystems

Entscheidend für die Umsetzung das Warnsystem ist der mobile Einsatz an einem Werkarbeiter. Im Arbeitsalltag darf das Anlegen des Systems keine zusätzliche Zeit in Anspruch nehmen. Vielbewegte Körperbereiche wie Hände, Arme und Beine sowie der Kopf müssen frei beweglich bleiben und können somit nicht für das Anlegen des Systems genutzt werden. Im Industrieumfeld wird Arbeitsschutzkleidung getragen, also auch eine Schutzweste mit Reflektionsstreifen, um auch bei Dunkelheit und Dämmerung gut sichtbar zu bleiben. In dieser Arbeit wird der Ansatz einer Warnweste gewählt. Das bedeutet, alle Bestandteile des Warnsystems müssen in eine Sicherheitsweste integriert werden. Der Arbeiter kann zu Arbeitsbeginn die Weste mit dem integrierten mobilen Warnsystem anlegen und zum Arbeitsende wieder ablegen, ohne zusätzliche Schritte ausführen zu müssen.

In den folgenden Kapiteln wird die Umsetzung des Detektionssystems und des Aktorsystems sowie dessen Integration in eine Sicherheitsweste vorgestellt.

5.1 Detektionssystem

Für das Detektionssystem werden zuerst geeignete Sensoren ausgewählt und durch Messungen deren Verwendungsmöglichkeit geprüft. Anschließend wird die Anordnung der verwendeten Sensoren beschrieben. Es folgt die Auswahl der Verarbeitungseinheit in Bezug auf die benötigte Rechenleistung und Energieeffizienz. Im Softwareteil werden einzelne Abschnitte der Sensordatenauswertung, Objektdetektion sowie Gefahreneinstufung und die GUI des Warnsystems vorgestellt.

5.1.1 Hardware

5.1.1.1 Auswahl der Sensoren

Ultraschall

Tabelle 6: Auswahl an Ultraschall-Sensoren

Bezeichnung	maximale Entfernung	Öffnungswinkel	Interface	el. Anbindung
Devantech SRF04	3 m	60 °	TTL signal	5 V, 50 mA
Devantech SRF10	6 m	60 °	I2C	5 V, 15 mA
Maxbotix MaxSonar-EZ1	6,5 m	17 °	UART	5 V, 3 mA

Tabelle 6 zeigt eine Auswahl an Ultraschall-Sensoren. Das Warnsystem muss Objekte in einem Bereich von bis zu 4 Metern erkennen. Entsprechend muss die maximale Entfernung der Sensoren über diesem Wert liegen. Der Öffnungswinkel sollte möglichst groß gewählt werden, damit durch möglichst wenig Sensoren ein großer Detektionsbereich entsteht. Die Schnittstelle zum PC muss über einen Mikrocontroller erfolgen, welcher über das Interface der Sensoren die Daten empfängt, verarbeitet und weitersendet. Üblich ist hierbei die Verwendung von I2C, einer digitalen Bus-Schnittstelle. Somit können mehrere Sensoren über eine gemeinsame Daten- und Zeitgeberleitung agieren.

Aufgrund dieser Spezifikationen wird der Sensor mit der Bezeichnung „Devantech SRF10“ (Abbildung 10) näher untersucht:

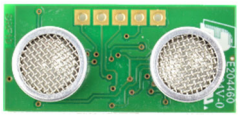


Abbildung 10: Ultraschallsensor

Der Sensor arbeitet mit einer Frequenz von 40 kHz und gibt die Entfernungsinformationen über die digitale Schnittstelle I2C aus. Durch Setzen von Registern im Sensor kann die Informationsausgabe als Laufzeit-Mikrosekunden oder als Entfernung in Millimeter oder Inches gewählt werden. Der Sensor besitzt ein Register zur Einstellung der Sendeleistung. Da bei dieser Anwendung ein möglichst großer Detektionsbereich entstehen soll, wird die Sendeleistung auf maximal gestellt. Ein weiteres Register dient der Änderung der I2C-Adresse für den Multi-Sensorbetrieb. Der Sensor benötigt 4 Leitungen: Spannungsversorgung 5V und GND sowie SCL und SDA der I2C Schnittstelle. (robot-electronics.co.uk, 2020)

Radar

Der Sensor „Position2Go development kit“ von Infineon arbeitet mit einer Frequenz von 24 GHz. Er arbeitet mit dem FMCW-Prinzip und liefert über eine USB-Schnittstelle seriell die aufgenommenen Rohdaten oder durch FFT und Doppler ausgewertete Objektdaten. Zur Kommunikation wird vom Hersteller eine Bibliothek bereitgestellt. Die maximale Entfernung liegt bei 15 Metern und der Öffnungswinkel des Detektionsbereichs bei 60°. Die Objektdaten bestehen aus bis zu fünf erkannten Objekten, welche jeweils durch Verstärkung, Entfernung, Winkel und Geschwindigkeit beschrieben werden.
Abbildung 11 zeigt die Vorder- und Rückseite des Sensors. Dabei befindet sich im oberen Bereich des Sensors ein Debugger zum Aufspielen einer neuen Firmware und zum Programmieren des integrierten Mikrocontrollers. Dafür müssen beide USB-Anschlüsse verbunden werden. Der Debugger kann nach erfolgreicher Konfiguration abgetrennt werden. Zum Betreiben des Sensors wird nur ein USB-Anschluss benötigt. (infineon.com, 2019)

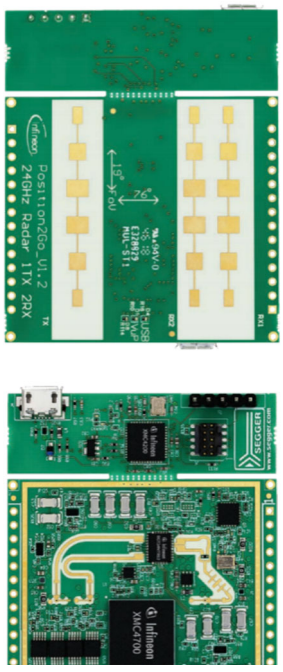


Abbildung 11: Radar-Sensor

Time-of-Flight-Kamera

Abbildung 12: ToF-Kamera

Abbildung 12 zeigt die ToF-Kamera des Herstellers Terabee mit der Bezeichnung „Terabee 3Dcam 80x60“. Sie nimmt den Raum mit einer Auflösung von 80 x 60 Pixeln und einer Geschwindigkeit von 30 FPS auf. Dabei entspricht der horizontale Öffnungswinkel 74° und der vertikale Öffnungswinkel 57°. Die Kamera besitzt einen USB-Anschluss und sendet die Bilddaten seriell weiter. Eine vom Hersteller gelieferte Bibliothek enthält ein Kommunikationsprotokoll, um auf die Daten der Kamera zugreifen zu können. (terabee.com, 2018)

5.1.1.2 Messbeschreibung, Messungen und Messauswertung

In diesem Kapitel wird die Verwendbarkeit der im Kapitel 5.1.1.1 ausgewählten Sensoren geprüft. Ein Versuchsstand soll die Entfernung, den Öffnungswinkel der Sensoren (Detektionsbereich) und den Objektwinkel vermessen können. Folgend werden diese Punkte näher beschrieben:

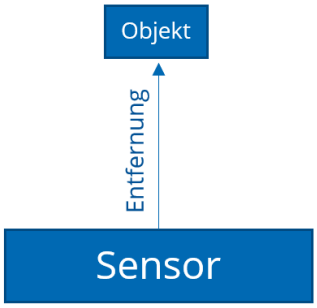


Abbildung 13: Skizze Entfernung

Die Entfernung

Die Entfernung ist der Abstand vom Sensor zum Objekt und muss in einem Bereich von 1 bis 4 Meter detektiert werden.

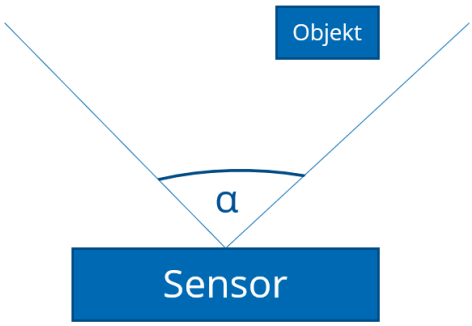
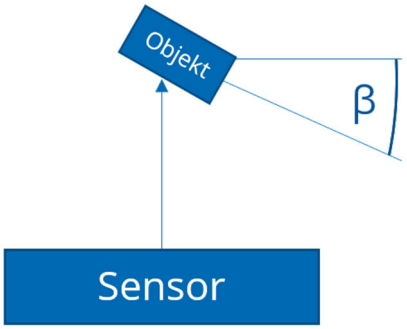


Abbildung 14: Skizze Detektionsbereich

Der Öffnungswinkel bzw. Detektionsbereich

Die Sensoren sollen einen großen Öffnungswinkel besitzen, damit das Warnsystem einen großen Bereich abdecken kann. Der Detektionsbereich ist hierbei der Bereich, indem ein Objekt mit einem Winkel alpha noch erkannt werden kann. Dabei sollen alle Winkel von -5° bis +10° in 1°-Schritten und von +10° bis 90° in 5°-Schritten gemessen werden. 0° entspricht der Mitte des Detektionsbereichs.



Der Objektwinkel

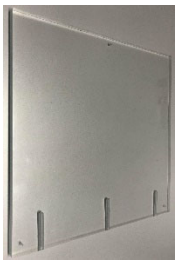
Der Objektwinkel Beta untersucht die Neigung eines Objekts in Bezug auf die Sensorposition. Dabei entspricht dem Winkel 0° ein senkrechtes Auftreffen des Sensorstrahls auf das Objekt. Es sollen alle Winkel von -5° bis +10° in 1°-Schritten und von +10° bis 90° in 5°-Schritten gemessen werden.

Abbildung 15:
Skizze Objektwinkel

Die Objekte

Zum Testen werden einheitlich Objekte mit einer Größe von 50 cm x 50 cm genutzt. Weiterhin soll der Einfluss des Materials auf die verschiedenen Sensortypen festgestellt werden. Tabelle 7 zeigt eine Auflistung der Materialien und deren Maße.

Tabelle 7: Auflistung der Messmaterialien

Material	Maße in cm (H x B x T)	Bild
Holz	50 x 50 x 1,8	
Pappe	50 x 50 x 2	
PVC	50 x 50 x 2	
Plexiglas	50 x 50 x 2	
Metall	50 x 50 x 0,2	
Metall-Lochraster	50 x 50 x 0,2	

Der Messaufbau

Auf einem geschienten Unterbau aus Traversenteilen werden die oben beschriebenen Objekte angebracht (Abbildung 16). Dadurch kann die Entfernung leicht zwischen dem Messobjekt und den messenden Sensoren variieren. Die Validierung der Entfernung erfolgt über einen Laserentfernungsmesser. Außerdem sind sowohl der Sockel für die Sensoren als auch der des Objektes drehbar gelagert, um den Sensorwinkel und den Objektwinkel einstellen zu können. Wie in Abbildung 16 gezeigt wird, lassen sich diese Drehungen über einen Winkelindikator einstellen.

Abbildung 16: Bilder des Messaufbaus

Die Messauswertung

Die aufgenommenen Messdaten müssen vor einer weiteren Betrachtung transformiert werden. Die Ultraschalldaten wurden als Laufzeit aufgenommen, zusätzlich dazu die Temperatur, von welcher die Schallgeschwindigkeit beeinflusst wird. Formel (5) zeigt die Umrechnung der Laufzeiten in Zentimeter:

$$s = \frac{1}{2} \cdot \Delta t \cdot cs_{Luft} \tag{5}$$

$$cs_{Luft} = 331{,}5 + 0{,}5 \cdot \vartheta \tag{6}$$

s – Entfernung zum Objekt, cs_{Luft} – Schallgeschwindigkeit in der Luft, Δt – Laufzeit des Signals, ϑ – Raumtemperatur

Die Radardaten bestehen wie in Kapitel 5.1.1.1 beschrieben aus Objektdaten und können ohne weitere Bearbeitung verwendet werden.
Die Kamera liefert pro Bild 80 x 60 Tiefenpunkte, welche mit Hilfe eines Algorithmus ausgewertet werden müssen, sodass am Ende Objektdaten vorliegen. Ein in dieser Arbeit verwendeter Algorithmus wird im Kapitel 5.1.2.2 erläutert.

Der Ultraschallsensor erkennt bei einem Meter Entfernung alle Materialien sehr gut. Der maximale Sensorwinkel beträgt hier 65° und der Objektwinkel 70°. Bei bereits zwei Metern Entfernung sinkt der maximale erkennbare Sensorwinkel auf 15° und der Objektwinkel auf 5°. Weiterhin liefert der Sensor nur die Laufzeit zum nächsten Objekt. Befindet sich ein nicht bewegtes Objekt in unmittelbarer Umgebung und ein risikobehaftetes, näherkommendes Objekt weiter weg, wird die Gefahr nicht immer wahrgenommen. Da der Ultraschallsensor bei größeren Entfernungen in Sensorwinkel sowie Objektwinkel eingeschränkt ist und nur die Laufzeit zum nächsten Objekt wahrnimmt, erfüllt er seinen Zweck nicht und wird bei der weiteren Betrachtung ausgeschlossen.

Der Radarsensor erkennt im Messbereich von 2 bis 4 Metern die Objekte bis zu einem maximalen Öffnungswinkel von 70° (von +35° bis -35°). Der Objektwinkel ist stark vom Material abhängig und wird in folgender Tabelle (Tabelle 8) dargestellt:

Tabelle 8: Daten der Messauswertung

Material	max. erkannter Objektwinkel 2 m	max. erkannter Objektwinkel 3 m	max. erkannter Objektwinkel 4 m
Holz	70°	50°	70°
Pappe	10°	10°	8°
PVC	20°	10°	9°
Plexi-Glas	8°	35°	15°
Metall	15°	20°	55°
Metall-Loch	30°	45°	10°

Aufgrund des konstanten Öffnungswinkels wird der Radarsensor Verwendung finden. Zusätzlich zum Radar muss die ToF-Kamera eingesetzt werden, da Objekte mit einem entsprechend größeren Objektwinkel vom Radar nicht erkannt werden können. Die Verwendung von 2 verschiedenen Sensoren ermöglicht zusätzlich das Erkennen von mehr Materialien.

Die ToF-Kamera besitzt einen konstanten Öffnungswinkel von 74°. Alle Objekte in diesem Bereich werden bei ausreichender Größe wahrgenommen, solange sie nicht durchsichtig oder stark-glänzend sind. Somit wurden die Materialen Holz, Metall-Loch, Pappe und PVC erkannt. Metall und Plexi-Glas wurden nicht erkannt.

Die ToF-Kamera wird hier Einsatz finden, da sie auch bei einer Entfernung von einem Meter Objekte erkennen kann. Weiterhin werden stark geneigte Objekte wahrgenommen, abhängig von der Tiefe eines Objekts.

5.1.1.3 Anordnung und Anzahl der Sensoren

Das binokulare Gesichtsfeld des Menschen beträgt ca. 220°.
(Martin, Spektrum.de - Gesichtsfeld, 2000)
Entsprechend sind 140° ohne Kopfbewegung nicht einsehbar. Der nicht sichtbare Bereich muss weitestgehend durch die Sensoren des Warnsystems abgedeckt werden. Der Radarsensor besitzt einen Öffnungswinkel von 70°. Entsprechend müssen hier zwei dieser Sensoren Einsatz finden. Die ToF-Kamera besitzt einen Öffnungswinkel von 74°. Da die Kamera viel Rechenleistung benötigt, mit den Abmessungen 5,4 x 5,3 x 2,4 Zentimetern für die Integration in eine Weste groß ist und ca. 100 Gramm wiegt, wird hier vorerst nur eine Kamera Einsatz finden. Abbildung 17 zeigt das Gesichtsfeld des Menschen, sowie die Anordnung der Sensoren mit Ihren Öffnungswinkeln. Die Radarsensoren werden mit einem Winkelversatz von 35° angebracht, sodass diese zu zweit den gesamten Bereich außerhalb des Gesichtsfeldes abdecken. Die ToF-Kamera deckt den mittleren Teil des Rückens ab. Folgend eine Tabelle (Tabelle 9) mit den Bereichen (0° entspricht dabei senkrecht zum Rücken der Person).

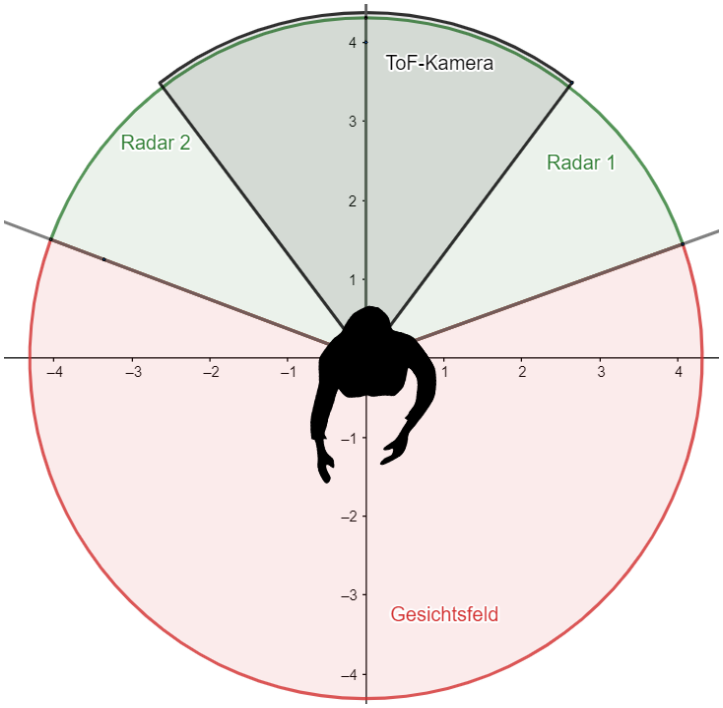


Abbildung 17: 360° Draufsicht auf die Arbeitsbereiche

Tabelle 9: Bereichseinteilung mit Winkelangabe

Bereich	Winkel
Gesichtsfeld	70° bis 290°
Radar 1	0° bis 70°
Radar 2	290° bis 0°
ToF-Kamera	323° bis 37°

5.1.1.4 Auswahl einer Recheneinheit

Das System soll in Echtzeit arbeiten, folgende Aufgaben müssen dabei erledigt werden:

- Befehlsempfang über die GUI
- Einlesen der Sensordaten,
  - ToF 30 FPS mit Objektdatengenerierung
  - 2 x Radar 100 FPS
- Algorithmus zur Warnung
- Kommunikation mit dem Aktorsystem

Die Kommunikation mit dem Aktorsystem und der Befehlsempfang über die GUI nehmen wenig Rechenleistung in Anspruch. Der Algorithmus zur Warnung wird immer dann ausgeführt, wenn neue Objektdaten generiert worden sind und benötigt mehr Rechenleistung. Der Empfang von den Objektdaten der Radarsensoren benötigt wenig Rechenleistung. Der Empfang der ToF-Kamera Rohdaten nimmt mehr Rechenleistung als der Empfang der Objektdaten der Radarsensoren in Anspruch, da pro Bild 4800 Bildpunkte gespeichert werden müssen. Anschließend muss hier eine Generierung der Objektdaten erfolgen, welche enorme Rechenleistung erfordert. Der prototypische Aufbau des Warnsystems soll eine große Kapazität an Rechenleistung besitzen, um die Auslastung nach dem Aufbau prüfen und ggf. in einem Folgesystem die Recheneinheit kleiner dimensionieren zu können.

Es wird der Mini-PC „NUC10i7FNK2“ (Abbildung 18) aufgrund seiner kleinen Bauform und dem leistungsstarken Prozessor ausgewählt. In Tabelle 10 sind wichtige Kennwerte zu finden:

Abbildung 18: Mini-PC

Tabelle 10: Kenndaten des Mini-PCs " NUC10i7FNK2"

Prozessor:	Intel Core i7-10710U – 4,7 GHz
Maße:	(117 x 36 x 112) mm
Spannungsversorgung:	19 VDC
Auswahl an Schnittstellen:	W-Lan, USB, Lan, SD-Karte

(intel.com, 2019)

5.1.2 Software

Dieses Kapitel befasst sich mit der Umsetzung des Softwarekonzepts des Detektionssystems.

5.1.2.1 Auswahl der Softwarekomponenten

Das Software wird in der IDE „Visual Studio“ in C++ programmiert. Die Kommunikation mit der Warnsystem GUI soll über TCP/IP erfolgen. Hier wird eine entsprechende Bibliothek eingebunden. Die Radarsensoren sowie die ToF-Kamera besitzen eigene Bibliotheken, welche das Auslesen der Daten über den virtuellen USB-Com-Port ermöglicht. Die Kommunikation mit dem Aktorsystem läuft ebenfalls über den virtuellen USB-Com-Port, hier muss jedoch ein Kommunikationsprotokoll entworfen werden, zu finden in Kapitel 5.2.2.2. Zur Visualisierung der ToF-Kamera-Tiefeninformationen wird die Bibliothek „opencv2“ verwendet. Die Bibliotheken können dem Programmcode des Detektionssystems von der mitgelieferten CD entnommen werden.

5.1.2.2 Implementierung der Software

Tabelle 11: Parallele Aufgaben der Detektionssystem-Software

Thread-Bezeichnung	Beschreibung
Main	Einstiegspunkt in das Programm, Objektbildung Kamera, Gefahrenanalyse, Ansteuerung Aktorsystem
Warnsystem-GUI	zur Kommunikation mit der GUI
Radar	zum Einlesen der Radar-Objektdaten
Kamera	zum Einlesen der ToF-Kamera-Daten,
depthViewer	zur Anzeige der unbearbeiteten/ bearbeiteten Bilddaten

Tabelle 11 zeigt die parallel ausgeführten Tasks des Detektionssystems, welche folgend näher beschrieben werden:

Main-Thread

Der Main-Thread stellt den Haupt-Eintrittspunkt in das Programm dar. Zuerst wird der Thread Warnsystem-GUI gestartet und die Kommunikation mit dem Aktorsystem initialisiert. Anschließend wird der Radar-Thread gestartet. Es wird gewartet, bis die Radarsensoren über den Radar-Thread initialisiert sind. Nun wird der Kamera-Thread sowie der depthViewer-Thread gestartet. Es folgt der Beginn der Main-Schleife, welche sich während des Betriebs des Warnsystems endlos wiederholt. In der Schleife wird bei vorhandenen Radarobjekt-Daten die Gefahrenanalyse durchgeführt. Sollte eine Warnung notwendig sein, werden Informationen an das Aktorsystem gesendet. Anschließend wird auf neue ToF-Kameradaten geprüft. Bei neuen Daten wird zuerst die Objektbildung ausgeführt mit anschließender Gefahrenanalyse. Zum Schluss werden Informationen an das Aktorsystem gesendet, falls Warnungen aufgrund der ToF-Kameradaten nötig sind. Abbildung 19 zeigt den Programmablaufplan des Main-Threads.

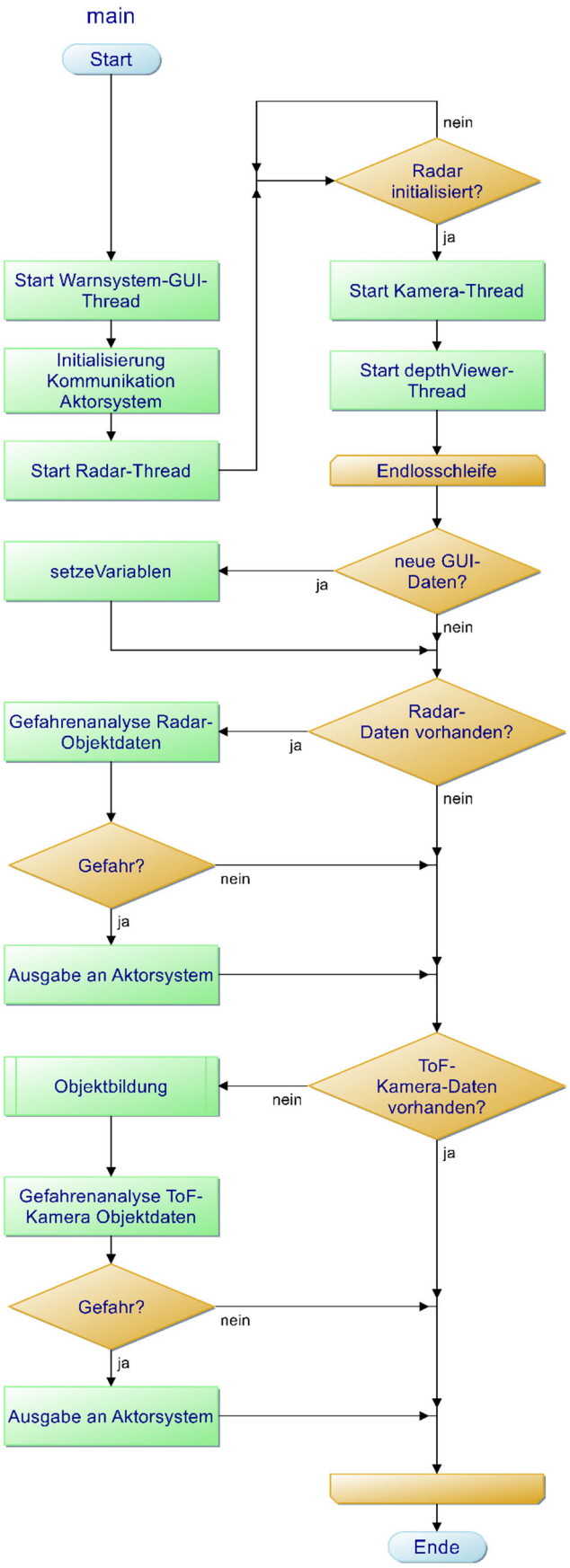


Abbildung 19: Programmablaufplan des Main-Threads

Die Objektdetektion der ToF-Kamera-Tiefenbilder wird hier näher beschrieben:

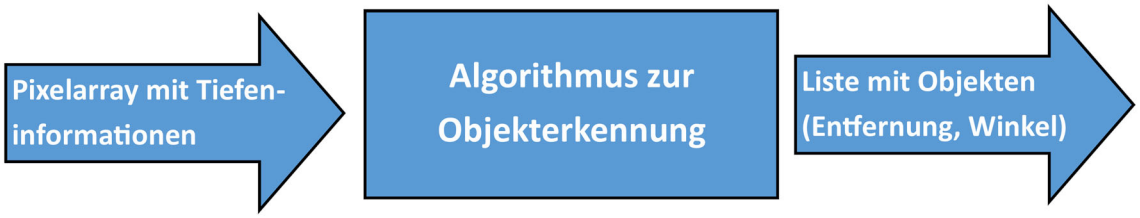


Abbildung 20: Schematische Darstellung des Objekterkennungs-Algorithmus

Die Eingabe besteht aus einem Pixelarray, wobei jeder Pixel einen Wert in Millimetern enthält und somit die Entfernung zum Reflektionspunkt angibt. Durch den Algorithmus entsteht aus diesen Entfernungsinformationen eine Liste von Objekten mit jeweiliger Angabe der Entfernung und des Winkels.

Algorithmus:

1. Zuordnung jedes Pixels zu einer Zehn-Zentimeter-Entfernungsklasse
2. Ausmusterung von Pixeln mit zu großer und zu kleiner Entfernung
3. Clusterung von benachbarten Pixeln, wenn sie zur gleichen Entfernungsklasse gehören, jede Clusterung ist ein Objekt
4. Auswahl der Objekte mit einer Pixelanzahl größer 26 Pixel (Default-Wert, über GUI einstellbar)
5. Entfernung eines Objekts: entspricht der Zehn-Zentimeter-Entfernungsklasse
6. Winkel eines Objekts: Bestimmung der Koordinaten des Mittelpunktpixels des Objekts, Winkelberechnung über die Entfernungsklasse, siehe Formel (7):

$$\varphi = \tan^{-1}\left|\frac{x}{\sqrt{e^2 - x^2}}\right| \cdot sign(x) \cdot \frac{180}{\pi} \qquad (7)$$

φ – Winkel zum Objekt, x – horizontale Koordinate des Pixels,
e – Tiefeninformation des Pixels

Der Algorithmus zur Gefahreneinstufung wird hier näher beschrieben.

Die Gefahrenbewertung erfolgt in 3 Stufen:

- Keine Gefahr
- Geringe Gefahr
- Große Gefahr

Zunächst findet die Gefahrenanalyse getrennt zwischen den Objektdaten der ToF-Kamera und den Radar-Sensoren statt, um die Funktionalität beider Sensorsysteme zu überprüfen.

Die Objektdaten der Kamera bestehen aus Entfernungs- und Winkelwerten. Deshalb wird die Kamera zur Warnung bei der Unterschreitung einer bestimmten Entfernung genutzt. Die Warnschwellenwerte können über die GUI eingestellt werden und besitzen folgende Default-Werte:

- Keine Gefahr, wenn alle Objekte weiter als 300 Zentimeter entfernt sind
- Geringe Gefahr, wenn ein Objekt weniger als 300 Zentimeter entfernt ist
- Große Gefahr, wenn ein Objekt weniger als 100 Zentimeter entfernt ist

Die Gefahr wird mit der Richtungsinformation an das Aktorsystem ausgegeben, wenn mindestens 2 Messungen in Folge eine gleiche Gefährdung ergeben.

Die Objektdaten des Radars bestehen aus Entfernungs-, Winkel- und Geschwindigkeitswerten. Deshalb wird das Radar zur Warnung bei der Überschreitung einer bestimmten Geschwindigkeit genutzt. Die Warnschwellenwerte können über die GUI eingestellt werden und besitzen folgende Default-Werte:

- Keine Gefahr, wenn alle Objekte langsamer als 2 km/h sind
- Geringe Gefahr, wenn ein Objekt schneller als 2 km/h ist
- Große Gefahr, wenn ein Objekt schneller als 6 km/h ist

Die Gefahr wird mit der Richtungsinformation an das Aktorsystem ausgegeben, wenn mindestens 2 Messungen in Folge eine gleiche Gefährdung ergeben.

Warnsystem-GUI-Thread

In diesem Thread wird der TCP-Client zur Kommunikation mit der Warnsystem-GUI initialisiert. Dabei ist es nötig, die IP-Adresse des TCP-Hosts der GUI in der Variablen „ipAdress_local“ einzutragen. Befindet sich die GUI Software auf demselben Rechner wie das Warnsystem, lautet die Adresse „127.0.0.1“.

Sobald Daten vom Host empfangen werden, wird das Flag „newdataGUI“ gesetzt und gesendete Werte werden in das Detektionssystem über den Main-Thread eingelesen.

Folgend wird die Warnsystem-GUI näher beschrieben.

Durch die GUI sollen folgende Variablen des Warnsystems einstellbar werden:

- Kamera
  - Anzahl der Pixel pro Objekt
  - Minimale Entfernung
  - Maximale Entfernung
- Radar
  - Minimale Entfernung
  - Maximale Entfernung
  - Minimale Geschwindigkeit
  - Maximale Geschwindigkeit
- Vibration
  - Intensität geringe Gefahrenstufe
  - Intensität große Gefahrenstufe

Abbildung 21 zeigt die GUI.

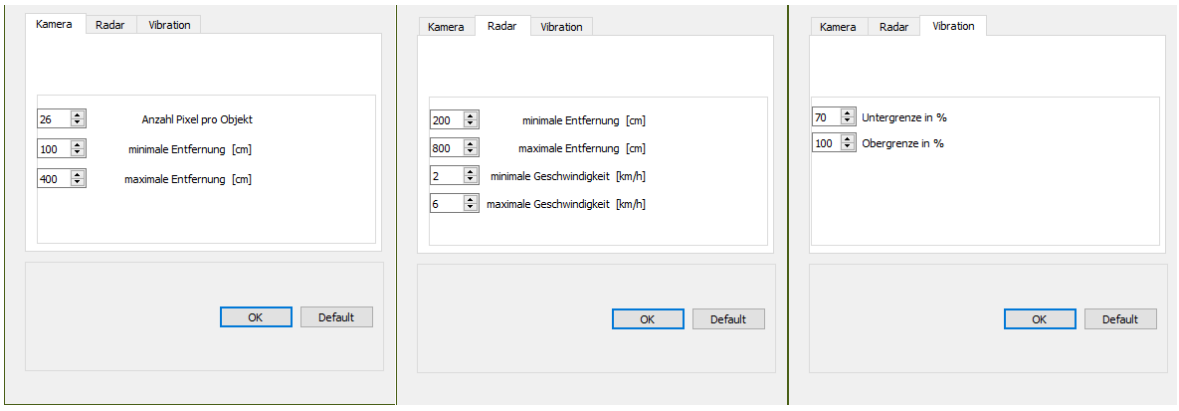


Abbildung 21: Grafische Darstellung der Warnsystem-GUI

Bei der Kamera sollen nur Objekt betrachtet werden, die eine bestimmte Anzahl an Pixeln besitzen. Die minimale und maximale Entfernung dienen zur Einstellung der Gefahrenschwellen, abhängig von der Entfernung der Objekte.

Die Grenzwerte „minimale Entfernung“ und „maximale Entfernung“ des Radars sorgen dafür, dass nur Objekte betrachtet werden, welche eine maximale und minimale Entfernung nicht über- bzw. unterschreiten. Objekte im Betrachtungsbereich werden in zwei Gefahrenstufen anhand ihrer Geschwindigkeit unterteilt, welche über „minimale Geschwindigkeit“ und „maximale Geschwindigkeit“ einstellbar sind.

Das Abändern der Intensität der Vibration der zwei Gefahrenstufen ist über das Setzen von „Untergrenze“ und „Obergrenze“ möglich.

Die GUI liefert nach Start die Default-Werte, mit welchen das Warnsystem betrieben wird. Durch Drücken des Buttons „OK“ werden alle Werte gesammelt in einem String an das Detektionssystem gesendet und dort anschließend in die entsprechenden Variablen eingelesen. Durch Drücken des Buttons „Default“ werden nach Änderung die Felder auf die Ursprungswerte zurückgesetzt.

Radar-Thread

Der Radar-Thread sucht die zwei über USB angeschlossenen Radarsensoren. Anschließend werden die USB-Endpoints in Variablen abgespeichert. Diese werden zur Kommunikation verwendet. Es folgt die Initialisierung und anschließend die Messroutine als Endlosschleife der Sensoren. Sensor 1 bekommt den Befehl zum Start der Messung und gibt nach einer endlichen Zeit über einen Interrupt in der Funktion „get_targets“ die detektierten Objekte zurück. Im Radar-Thread wird gewartet, bis über den Interrupt das Flag „WarteFlag“ wieder auf Null gesetzt wird. Anschließend wird das Flag im Radar-Thread wieder auf Eins gesetzt, sodass der zweite Sensor dem gleichen Ablauf folgen kann.

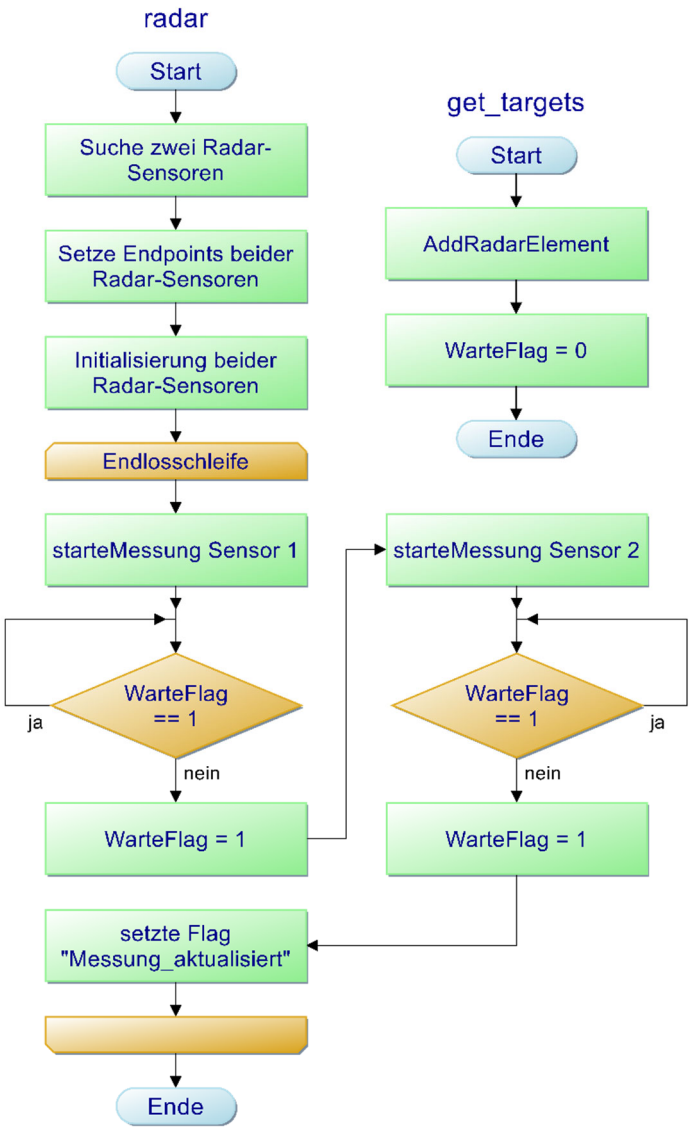


Abbildung 22: Programmablaufplan des Radar-Threads

Die Interrupt-Funktion „get_targets“ wird aufgerufen, sobald ein Radarsensor eine Messung abgeschlossen hat und Werte sendet. Die gesendeten Werte werden in der Funktion „AddRadarElement“ in einer Struktur aufgenommen und im Main-Thread weiterverarbeitet. Abbildung 22 zeigt den Programmablaufplan des Radar-Threads und der Interrupt-Funktion.

Kamera-Thread

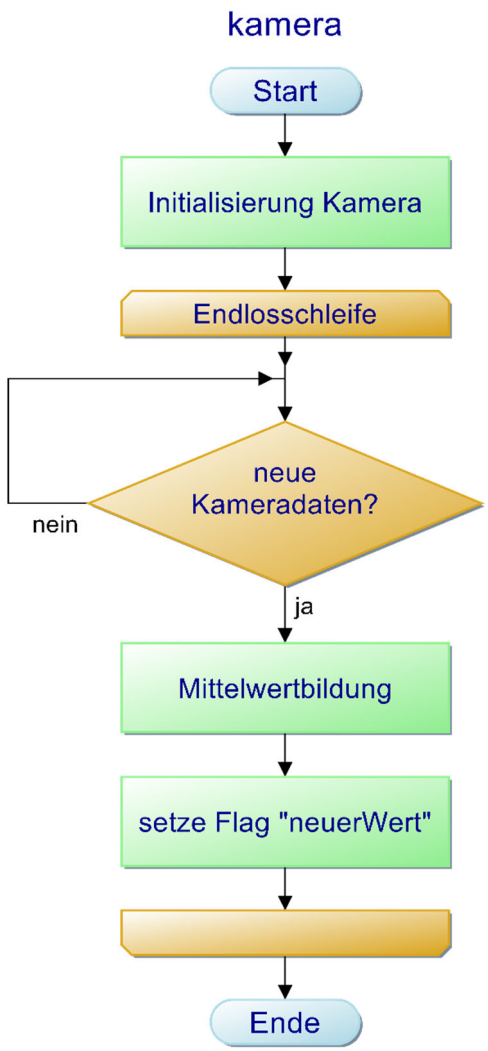
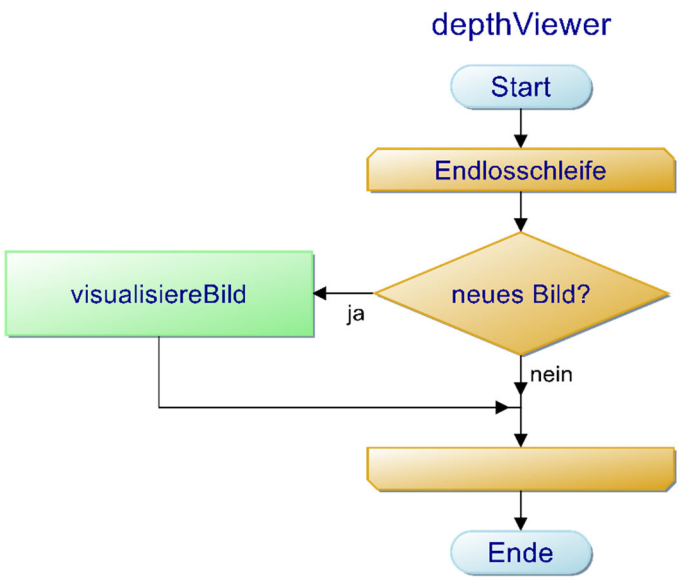


Abbildung 23: Programmablaufplan des Kamera-Threads

Der Kamera-Thread (Abbildung 23) dient zum Empfangen der Tiefenbildinformationen der ToF-Kamera. Zuerst wird die Kommunikation zur Kamera mit Hilfe der Bibliothek „openNi“ initialisiert. Anschließend wird auf neue Werte von der Kamera gewartet. Bei neuen Daten wird eine Mittelwertbildung ausgeführt und das Ergebnis in dem Vektor „exportMittelWert“ gespeichert, welcher vom Main-Thread aus aufgerufen werden kann. Das Flag „neuerWert“ zeigt eine Aktualisierung des Vektors an.

Die Mittelwertbildung verringert das Rauschen eines Bildes. Dazu wird jeder Pixel des aktuellen Bildes und zwei vorheriger Bilder addiert und durch einen Teiler dividiert. Der Teiler zählt pro Bild einmal, ausgenommen bei einem Pixel der Entfernung 0.
Sollten also alle drei Pixel (der drei Bilder) nicht 0 sein ergibt das einen Teiler von drei.

depthViewer-Thread



Der depthViewer-Thread (Abbildung 24) dient zur Visualisierung von ToF-Kamera-Bildern. Es wird in einer Endlosschleife auf neue Bilder geprüft; sobald ein neues Bild vorhanden ist, wird dieses dargestellt.

Abbildung 24: Programmablaufplan des depthViewer-Threads

Die Software des Detektionssystems, sowie der Warnsystem-GUI sind auf der beiliegenden CD zu finden.

5.2 Aktorsystem

In diesem Kapitel wird die Umsetzung des Aktorsystems erläutert. Zuerst werden die Hardwarekomponenten beschrieben. Es folgt anschließend die Anordnung der Vibrationsmotoren in der Weste. Das Kapitel schließt mit der Beschreibung der Software und dem GUI-Testsystem ab.

5.2.1 Hardware

5.2.1.1 Auswahl der Hardwarekomponenten

Der **Akkumulator** dient als mobile Spannungsversorgung des Gesamtsystems und muss entsprechend dimensioniert werden. Da dieser von einem Werkarbeiter getragen wird, dürfen das Gewicht und damit die Kapazität und auch die physikalische Größe nicht zu groß ausfallen. Dennoch muss das System stabil über längere Zeit laufen können.

Gesucht wird ein Akkumulator mit 20 Volt und mindestens 5 Ampere, welcher vom Nichtfachmann ohne großen Aufwand getauscht und geladen werden kann. Zumindest im Bauwesen bekannt sind sogenannte „Werkzeugakkus“ bzw. „Schiebeakkus“. Diese sind bei jedem mobilen Elektro-Baugerät zu finden und können mit einem entsprechenden Ladegerät nach einfacher Anbringung in kurzer Zeit geladen werden.

Bei einem Prototyp des Systems sollen zwei 20 Volt Akkus mit je 3 Amperestunden zum Einsatz kommen, zu sehen in Tabelle 12. Das ergibt allein für die Stromquelle des Systems 680 Gramm Gewicht für den Tragenden, für eine maximale Nutzungsdauer von 72 Minuten bei einem dauerhaften Strom von 5 Ampere. Es muss hier also ein Kompromiss zwischen Gewicht und Nutzungsdauer gefunden werden; eine entsprechende Optimierung des Strombedarfs sollte später noch betrachtet werden.

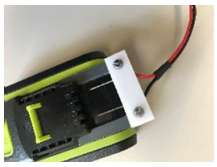
Berechnung der Nutzungsdauer bei dauerhaftem maximalem Strom (Formel (8)):

$$t = \frac{C}{I} \cdot 60\ min = \frac{2 \cdot 3\ Ah}{5\ A} \cdot 60\ \text{min} = \underline{72\ min} \qquad (8)$$

t – Laufzeit, C – Kapazität, I – Strom

Kenndaten:

Tabelle 12: Kenndaten und Bilder des Akkumulators „Jialitt WA3551"

Bezeichnung	„Jialitt WA3551 Akku 3,0 Ah 20 V“
Spannung	20 V
Kapazität	3 Ah
Gewicht	340 g
Maße (L x B x H)	10 cm x 8 cm x 4 cm
Bilder	 Akku  Ladegerät  Ladestecker

(amazon.de, 2020)

Ein passendes Ladegerät, zu sehen in Tabelle 12, wird vom gleichen Hersteller bezogen; mit diesem Ladegerät können beide Akkus synchron innerhalb von 90 Minuten vollständig geladen werden. Zum Benutzen des Akkus mit dem Warnsystem wird ein Stecker (Tabelle 12, Bild: Ladestecker) konstruiert.

DC/DC-Wandler

Der Akkumulator liefert 20 Volt und kann somit das Detektionssystem ohne weitere Wandlung betreiben. Das Aktorsystem benötigt jedoch einmal 5 Volt für den Arduino und weiterhin eine variable Spannung zwischen 2 Volt und 8 Volt zum Ansteuern der Taktoren. Für diesen Zweck werden zwei DC/DC-Wandler verwendet.

Der erste Wandler muss auf den maximalen Strom für zehn aktive Taktoren, addiert mit einem Puffer, ausgelegt werden. Da zum jetzigen Zeitpunkt die Wahl des Vibrationsmotors noch aussteht, muss der Wandler variabel von 2 bis 8 Volt einstellbar sein. Der maximale Strom der Motoren in der engeren Auswahl beträgt 85 Milliampere. Zehn Stück dieser Motoren würden 850 Milliampere unter Volllast benötigen. Es wird ein entsprechender DC/DC-Wandler mit mindestens 1 Ampere bei einstellbarer Spannung von 2 bis 8 Volt benötigt.

Der zweite Wandler für den Arduino muss mindestens 500 Milliampere bei 5 Volt liefern, da der Mikrocontroller ein Strom-Maximum-Rating von 200 Milliampere (Atmel, Datenblatt) besitzt. 300 Milliampere werden als Puffer für Spitzenströme und Einschaltströme eingeplant.

Um den Aufwand bei den Wandlern möglichst gering zu halten, werden 2 baugleiche Wandler mit fertigem PCB-Design verwendet: „DEBO DCDC DOWN 4“ (Tabelle 13).

Tabelle 13: Kenndaten des DC/DC-Wandlers „DEBO DCDC Down 4“

Bezeichnung	„DEBO DCDC Down 4“
Eingangsspannung	4,75 … 23 V
Ausgangsspannung	1 … 17 V
Maximaler dauerhafter Ausgangsstrom	1,8 A
Maximaler kurzzeitiger Ausgangsstrom	3 A
Schaltfrequenz	340 KHz
Bild	 DC/DC-Wandler

(reichelt.com, 2019)
Die variable Ausgangsspannung und der maximale Dauerstrom dieses Wandlers liegen weit über den Anforderungen und er kann somit Verwendung finden.
Der Wandler kann durch Verbinden mit Stiftleistenpins in das eigene PCB integriert werden und nimmt wenig Platz in Anspruch. Der zusätzliche Wandler für den Arduino ist optional und muss nur aufgelötet bzw. verwendet werden, wenn die USB-Schnittstelle nicht eingesetzt wird.

Mikrocontroller

Für das Aktorsystem wird eine Verarbeitungseinheit benötigt, welche zum einen die Befehle für die Vibration empfangen kann und zum anderen PWM-Signale für die Treiberschaltung erzeugt, so dass die Taktoren vibrieren können. Es werden zehn Taktoren unabhängig voneinander angesteuert, somit werden auch zehn PWM-Pins benötigt. Die unten aufgeführte Tabelle fasst die Anforderungen an den Mikrocontroller zusammen.

Tabelle 14: Anforderungen an den Mikrocontroller

Kriterium	Anforderung	ATMEGA 2560
Versorgungsspannung	<= 20 V, üblich 3V3 bzw. 5 V	5 V
Anzahl PWM-Pins	>= 10	15 Stück
PWM-Frequenz	490 Hz	490 Hz (Pins 4, 13: 980 Hz)
Flankenzeit	<= 100 µs	100 ns
Schnittstelle	UART	4 x USART
Timer-Anzahl	>= 3	6 Stück

Die Versorgungsspannung für den Mikrocontroller wird durch einen DC/DC-Wandler zur Verfügung gestellt und durch diesen begrenzt. Der ausgewählte Wandler ist im Bereich von 1 Volt bis 17 Volt funktionsfähig, die übliche Versorgungsspannung eines µC von 3,3 bzw. 5 Volt ist somit erreichbar.
Die PWM-Frequenz der PWM-Pins muss nicht hoch gewählt werden, da Vibrationsmotoren träge sind und nicht auf schnelle Spannungsänderungen reagieren können. 490 Hertz ist hier ein akzeptabler Wert.
Die Flankenzeit der PWM-Pins muss hier möglichst gering gewählt werden, da in der Treiberschaltung bei langen Schaltzeiten hohe Verlustleistungen entstehen. An dieser Stelle sollte die Flankenzeit kleiner 1/20 der Periode entsprechen und liegt somit bei 100 Mikrosekunden.

Um den Prototypaufbau zu beschleunigen, wird hier eine Arduino-Plattform gewählt. Dies hat den Vorteil, dass alle Pins des Mikrocontrollers über Stiftleisten erreichbar sind. Zu einem späteren Zeitpunkt kann der Mikrocontroller in das eigene Design integriert werden und hat dann den Vorteil, Platz und Energie zu sparen, da nur die nötigen Bauteile und Schaltungen von der Plattform übernommen werden.

Die Arduino-Plattform ATMEGA 2560 (Abbildung 25) besitzt alle erforderlichen technischen Spezifikationen und kann mit Hilfe einer USB-Schnittstelle über die Arduino-IDE am PC programmiert werden. (arduino.cc, 2018)

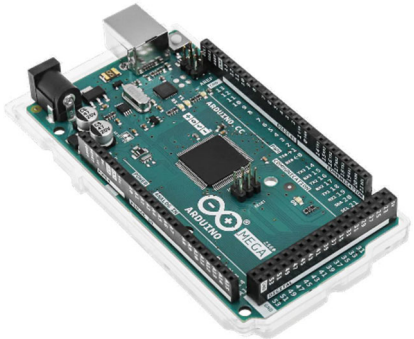


Abbildung 25: ATMega 2560

Die Werte für Versorgungsspannung, Anzahl der PWM-Pins, PWM-Frequenz, Kommunikationsschnittstelle und Timer-Anzahl wurde den Datenblättern entnommen, die Flankenzeit wurde mit einem Oszilloskop gemessen.

Zur Erzeugung der Flanken wurde ein einfaches Programm geschrieben, welches das Signal eines Pins zwischen HIGH und LOW umschaltet, zu finden in den Anlagen, Teil 2 (Seite 73). Die Flankenzeit für die fallende und steigende Flanke beträgt 100 Nanosekunden und ist somit ausreichend schnell (Abbildung 26).

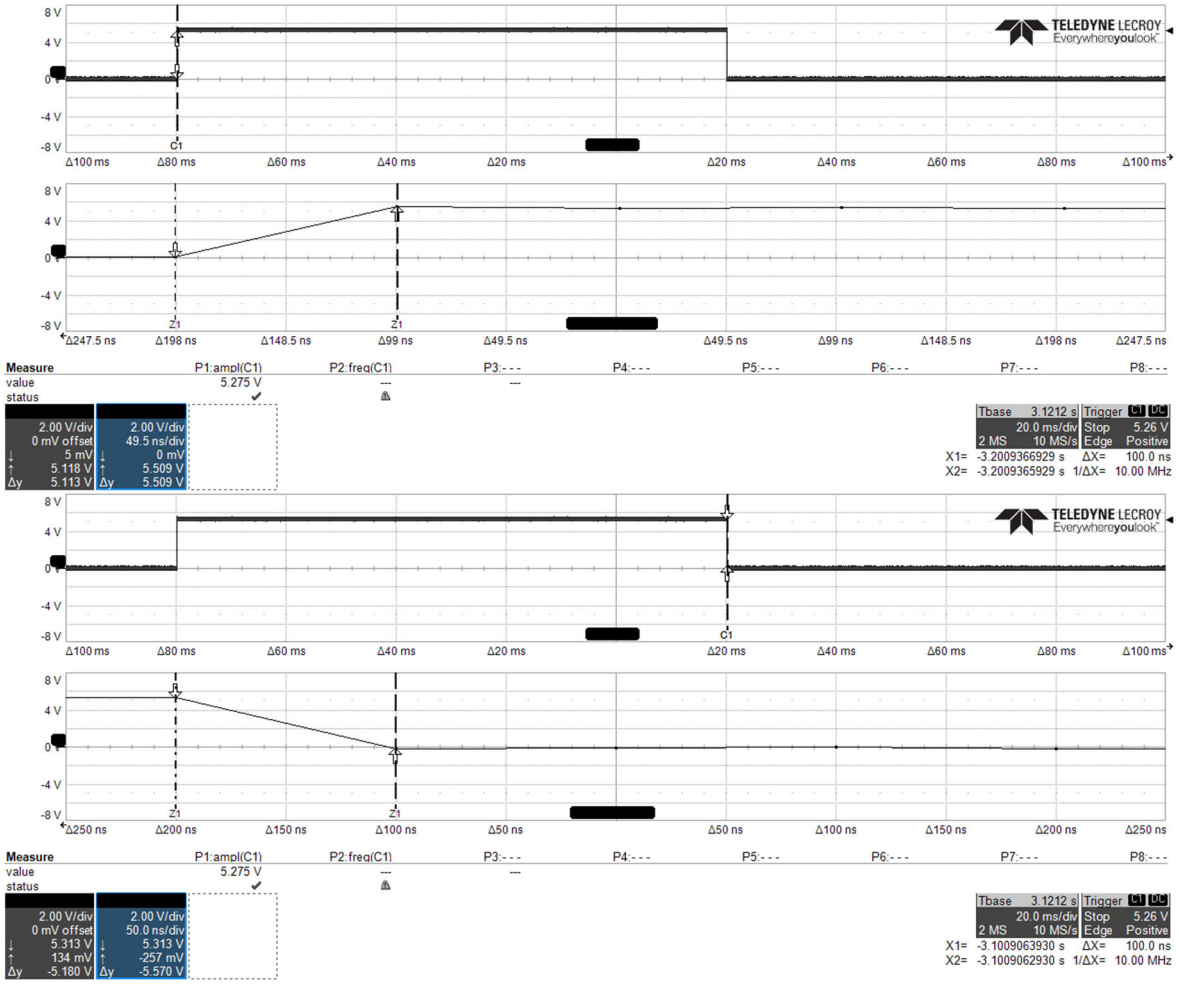


Abbildung 26: Graphische Darstellung der Flankenwechsel des ATMega 2560

Beschreibung: Zu sehen sind zwei Flankenwechsel (LOW auf HIGH und HIGH auf LOW) im Abstand von 100 Millisekunden. Die jeweils obere Grafik in der Abbildung 26 zeigt den aufgenommenen Zeitbereich, wobei die jeweils untere Grafik einen vergrößerten Bereich der Flankenwechsel abbildet. Die Cursor Z1 markieren den Bereich des Flankenwechsels und es wird die Zeitdifferenz berechnet (hier 100 Nanosekunden), welche jeweils rechts in der Abbildung unter den Grafiken und Trigger-Fenstern bei „ΔX“ abzulesen sind.

Die Pins des Mikrocontrollers können nicht mit hohen Strömen belastet werden; sie können deshalb für das Ansteuern der Aktorik, jedoch nicht für die Stromversorgung genutzt werden. An dieser Stelle findet eine **Treiberschaltung** Einsatz. Mit einem Widerstand in Reihe zum Mikrocontroller-Pin wird der fließende Strom begrenzt, er dient als Schutz. Ein Pulldown sorgt für ein Massepotential am Treiber, sobald der Mikrocontroller-Pin auf LOW schaltet oder ausgeschaltet wird, bzw. sich im Tristate befindet. Als Treiber dient ein Drei-Pin-MosFET, welcher mit einem Steuersignal als Schalter fungiert. Durch ihn werden die Taktoren im Ein-Zustand auf das Massepotential geschalten und ein Strom kann fließen bzw. im Aus-Zustand getrennt und die Taktoren befinden sich im Leerlauf (Abbildung 27).

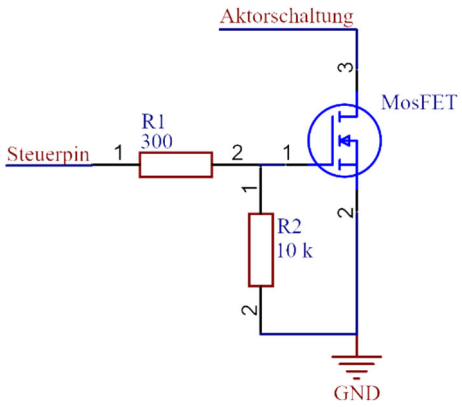


Abbildung 27: Treiberschaltung

Folgende Parameter werden für die Dimensionierung der Widerstände und des MosFETs benötigt:

Tabelle 15: wichtige Parameter für die Dimensionierung

Parameter	Wert
maximaler DC Strom pro I/O Pin	20 mA
Spannung I/O Pin im HIGH Zustand	5 V
maximaler Strom pro Taktor	80 mA
maximale Spannung pro Taktor	8 V
Schaltgeschwindigkeit PWM	490 Hz

Der Serienwiderstand am I/O-Pin muss mindestens 250 Ohm betragen, um einen zu hohen Strom zu vermeiden, zu sehen in der Rechnung (9). Um einen Sicherheitsfaktor mit einzubeziehen, wird der Widerstandswert 300 Ohm gewählt.

$$R = \frac{U}{I} = \frac{5\,V}{0{,}02\,A} = \underline{250\,\Omega} \qquad (9)$$

Für den Parallelwiderstand (Pulldown) zur Treiberschaltung nach dem Serienwiderstand müssen zwei Sachen beachtet werden:

- Je größer der Widerstand, desto höher ist die Steuerspannung am MosFET, sobald der Steuerpin auf HIGH geschalten wird
- Je kleiner der Widerstand, desto schneller wird das Massepotential beim Schalten des Steuerpins in den Tristate erreicht

Ein üblicher Wert für einen Pulldown ist 10 Kiloohm und wird in der Schaltung Einsatz finden. Durch den Spannungsteiler ergeben sich 4,85 Volt, welche als Steuerspannung am MosFET anliegen werden (Rechnung (10)).

$$U_{R_2} = \frac{R_2 \cdot U_{ges}}{R_{ges}} = \frac{10\ k\Omega \cdot 5\ V}{10\ k\Omega + 300\ \Omega} = \frac{10\ k\Omega \cdot 5\ V}{10{,}3\ k\Omega} = \underline{4{,}85437\ \mathrm{V}} \quad (10)$$

Als Schaltelement wird der MosFET „XP233N0501TR-G“ verwendet. Einige wichtige Datenblattwerte werden folgend aufgelistet und auf Verwendbarkeit geprüft:

- $V_{DSmax} = 30\ V$, die Drain-Source-Spannung wird maximal 8 Volt betragen
- $V_{GSmax} = \pm 20\ V$, die Gate-Source-Spannung wird maximal 4,85 Volt betragen
- $I_{Dmax} = 0{,}5\ A$, der Drain-Strom wird maximal 0,08 Ampere betragen

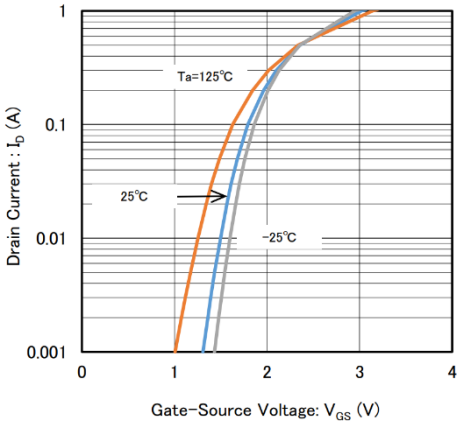


Abbildung 28: Id(Vgs) Diagramm

Alle oben betrachteten Werte liegen im gängigen Bereich. Wichtig ist noch die Betrachtung der Gate-Source-Spannung in Bezug auf den Drain-Strom. Dieser kann aus einem Diagramm im Datenblatt (Abbildung 28) abgelesen werden. Für den Taktor werden 80 Milliampere benötigt; bei den 4,85 Volt der Treiberschaltung könnte der MosFET bereits über 1 Ampere schalten und ist somit verwendbar.

Die vom MosFET umgesetzte Verlustleistung P_{tot} beträgt 40,316 Milliwatt und liegt somit unter dem Maximalwert (0,4 Watt), welcher im Datenblatt (Torex, Datenblatt) abgelesen werden kann.

$$P_{stat} = I_D \cdot r_{ds} = 80\ mA \cdot 0{,}5\ \Omega = \underline{0{,}04\ W} \quad (11)$$

$$P_{dyn} = P_{Last} \cdot T_{schalt} \cdot f = 0{,}08\ A \cdot 8\ V \cdot 0{,}0000001\ s \cdot 490\ Hz = \underline{0{,}314\ mW} \quad (12)$$

$$P_{steuer} = U_{GS} \cdot Q_G \cdot f = 4{,}85\ V \cdot 0{,}78 \cdot 490 = \underline{2\ \mu W} \quad (13)$$

$$P_{tot} = P_{stat} \cdot P_{dyn} \cdot P_{steuer} = \underline{\mathbf{40{,}316\ mW}} \quad (14)$$

Der Drain-Source-Widerstand r_{ds} und die Gate-Kapazität Q_G kann ebenfalls im Datenblatt (Torex, Datenblatt) abgelesen werden. Die Schaltzeit T_{schalt} wurde bereits bei der Auswahl des Mikrocontrollers bestimmt.

Je nach Einsatzort der **Aktorik** (in einer Armbanduhr, direkt auf der Haut oder in einer Stoffweste) muss die Vibration entsprechend stark sein, damit sie vom Menschen wahrgenommen werden kann. Aus diesem Grund werden Vibrationsmotoren mit zwei verschiedenen Bauformen und mehreren Größen betrachtet. Die Bauformen werden nach 1-Knopfzellenbauform und 2-Zylinderbauform unterschieden. Weitere wichtige Kenndaten der Motoren sind Spannung, Strom, Umdrehungsgeschwindigkeit und Maße - hier: Durchmesser und Tiefe (Abbildung 29).

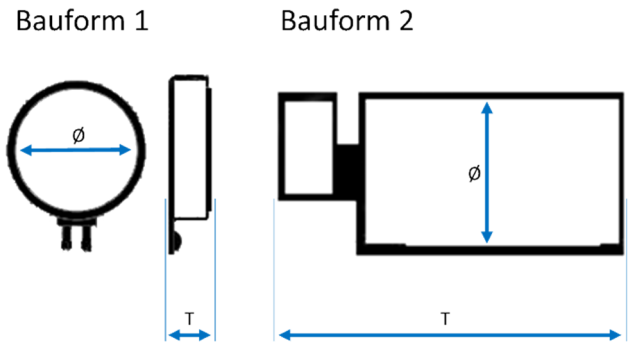


Abbildung 29: Skizze der Bauformen von Taktoren

Tabelle 16: Auswahl an Motoren

Parameter:	Spannung	Strom	Umdrehungen	Maße (Ø x T)	Bauform
Modell A	3,0 V	70 mA	12000 UpM	(8 x 2,2) mm	1
Modell B	3,0 V	70 mA	12000 UpM	(10 x 2,3) mm	1
Modell C	3,0 V	70 mA	12000 UpM	(5 x 11,3) mm	2
Modell D	2,7 V	75 mA	14000 UpM	(4,6 x 10,8) mm	2
Modell E	3,0 V	85 mA	10000 UpM	(6,8 x 15,2) mm	2
Modell F	3,0 V	85 mA	10000 UpM	(5,6 x 13,5) mm	2
Modell G	6,0 V	60 mA	9000 UpM	(12 x 20) mm	2

Das Aktorsystem funktioniert mit allen Taktoren, welche 8 Volt und 80 Milliampere nicht überschreiten. Somit können alle in Tabelle 16 aufgelisteten Taktoren eingesetzt werden. Nach dem Test der Taktoren auf Größe und Intensität der Vibration wird hier der Typ E bzw. F in einer Warnweste Einsatz finden.

Mechanik

Wie bereits im Kapitel 4.2 beschrieben, befinden sich die Treiber und die Schutzbeschaltung sowie die DC/DC-Wandler auf einem Arduino-Shield. Dieses Shield soll dieselbe Größe wie die Arduino-Plattform besitzen und durch Stiftleisten auf den Arduino aufgesetzt werden können.

Die Stiftleisten dienen als mechanischer Konnektor zwischen Arduino und Shield und weiterhin als elektrischer Konnektor für das Verbinden der Steuersignale und Spannungsversorgung.

Um Krafteinflüssen von außen entgegenzuwirken, wird eine Boden- und eine Deckplatte hergestellt, welche den Arduino und das Shield schützen. Die Deckplatte soll weiterhin als Zugentlastung für die angeschlossenen Kabel der Aktorik dienen.

Da die beiden ausgewählten Taktoren der Bauform 2 entsprechen, müssen diese in ein Gehäuse integriert werden. Die Schwungmasse befindet sich außerhalb des Motorengehäuses und würde durch das Futter der Weste ausgebremst werden.

5.2.1.2 Anordnung der Taktoren

Entscheidend für die Anordnung der Aktoren sind die Richtungsinformationen, welche ausgegeben werden sollen sowie die in Kapitel 2.3.1 beschriebene Zwei-Punkt-Schwelle der Haut. Das Warnsystem soll den Bereich außerhalb des Gesichtsfeldes einer Person aufnehmen und eine richtungsbedingte Vibration bei Gefahr auslösen. Die Taktoren müssen sich deshalb am Rücken einer Person befinden. Es werden 5 Taktoren in einer Reihe angeordnet, das ergibt einen Bereich von 28° pro Taktor. Die Taktoren müssen einen Abstand von mindestens 7 Zentimetern besitzen, um die Zwei-Punkt-Schwelle nicht zu unterschreiten.

Damit eine Untersuchung der Taktoren des Typs E und F möglich ist, wird jeweils eine Weste mit den zwei Typen bestückt.

Wie bereits im Kapitel 5.1.2.2 erwähnt soll die Ausgabe der Gefahrenstufen für die Objektdaten von Radar und ToF-Kamera zunächst getrennt erfolgen. Deshalb werden zwei Taktorreihen untereinander auf dem Rücken angebracht. Abbildung 30 zeigt die Anordnung der Taktoren in der Weste.

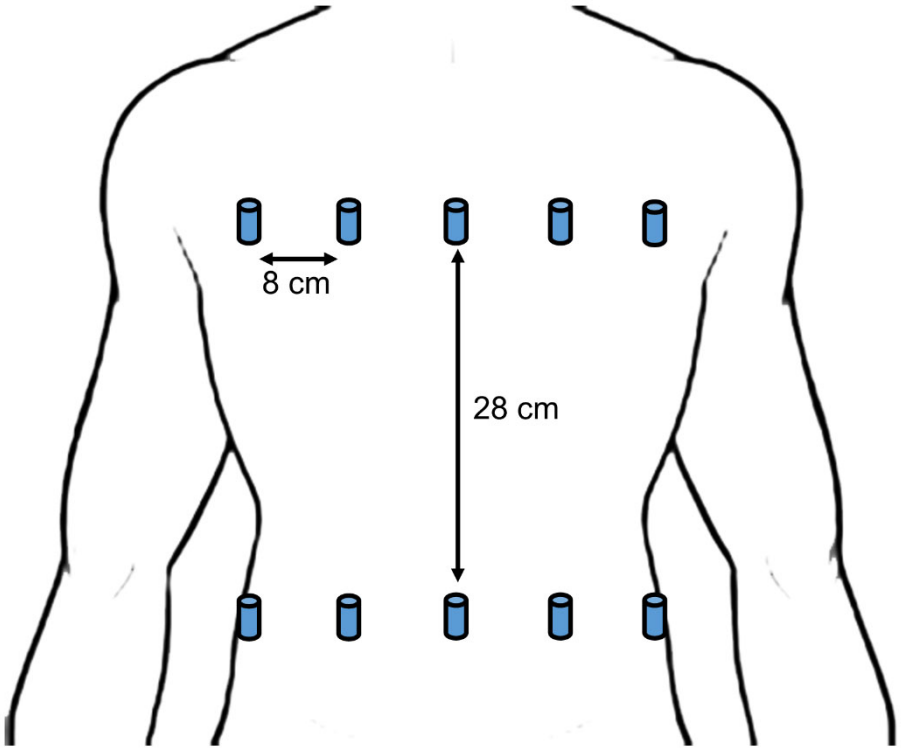


Abbildung 30: Anordnung der Taktoren

5.2.1.3 Schaltplan und Platinenlayout

Parallel zu jedem Vibrationsmotor wird eine Diode in Sperrrichtung gesetzt, um Induktionen beim Schalten entgegenzuwirken. Weiterhin wird die Spannungsversorgung der Aktorik durch das Einfügen mehrerer Kondensatoren als Puffer bei kurzzeitigen Stromspitzen abgesichert (Abbildung 31).

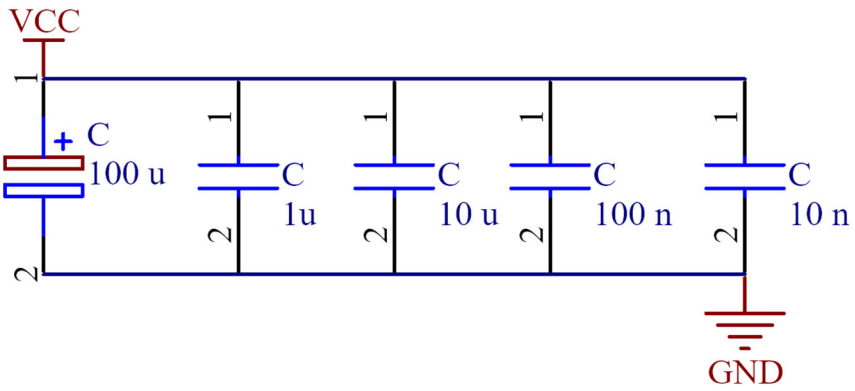


Abbildung 31: Schaltung der Pufferkondensatoren

Verschiedene Bauteile und deren Beschaltung wurden bereits im Kapitel 5.2.1.1 vorgestellt. Der vollständige Schaltplan ist in den Anlagen, Teil 3 (Seite 74) zu finden.
Das Platinenlayout wird dem Schaltplan entsprechend umgesetzt und ist in den Anlagen, Teil 4 (Seite 76) zu finden.

5.2.2 Software

5.2.2.1 Auswahl der Softwarekomponenten

Zum Übertragen der Daten über den virtuellen Com-Port wird eine „ComPort“-Bibliothek (ComPort.h) benötigt. Aus dieser können Funktionen zum Senden von einzelnen Bytes an die Arduino-Plattform ausgeführt werden. Diese Bibliothek findet im Testsystem und im Warnsystem Einsatz. Für das Programmieren der beiden Systeme wird die IDE VisualStudio von Microsoft verwendet. Im Testsystem wird weiterhin die Windows-Forms-Bibliothek (Windows.Forms) verwendet, um mit wenig Programmcode eine virtuelle Oberfläche mit den benötigten Steuerelementen erstellen zu können.

Das Aktorsystem wird mit der Arduino-plattform-eigenen Arduino-IDE programmiert. Hier wird die Serial-Bibliothek (serial.h) benötigt, um die Datenblöcke über die serielle Schnittstelle des Mikrocontrollers empfangen zu können.

Weitere Bibliotheken werden in den Systemen verwendet, um grundlegende Funktionen wie das Anlegen und Konvertieren von Dateitypen (stdint.h), das Berechnen mit mathematischen Operatoren (math.h), das Verwenden von Timern (timer.h) und das Benutzen von Multithreads (System.Threading) zu ermöglichen.

5.2.2.2 Implementierung der Software

Über die serielle Schnittstelle können Bytes gesendet werden. Diese Bytes müssen Informationen für die Aktorsteuerung enthalten. Das Kommunikationsprotokoll legt die Anzahl der Bytes, die Bytereihenfolge und die Information in jedem Byte fest, sodass der Sender die richtigen Bytes sendet und der Empfänger nach einer Auswertung die passende Aktion ausführen kann.

Das Aktorsystem

Nach dem Einbinden der Bibliotheken findet die Initialisierung der Com-Schnittstelle statt. Es wird die Symbolrate 115200 Bits pro Sekunde gewählt. Damit könnten 2880 Befehlsketten mit jeweils 5 Bytes in einer Sekunde gesendet werden. Die Schnittstelle ist für diesen Zweck verwendbar, da nicht mehr als eine Befehlskette pro Sekunde gesendet wird.

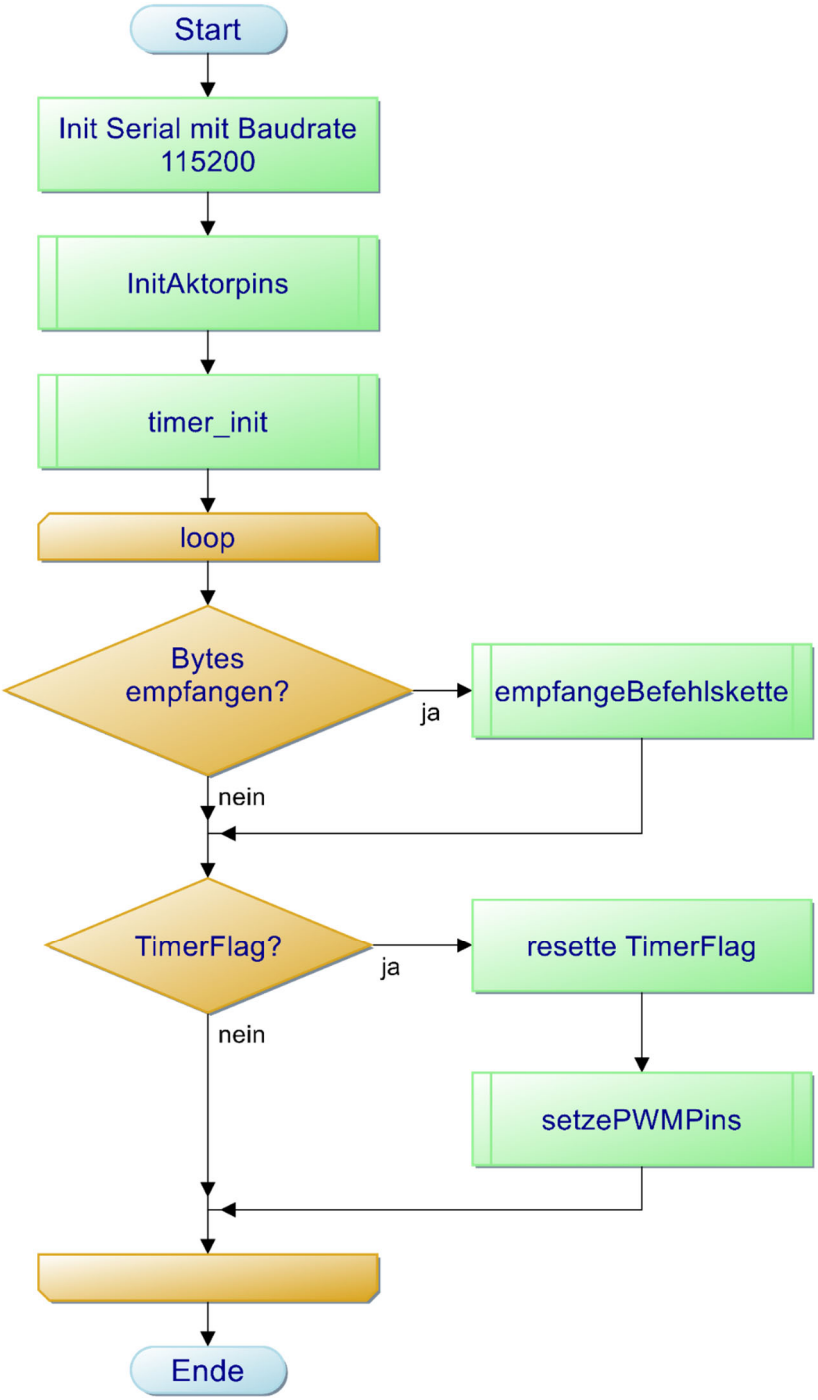
Für die Aktorsteuerung werden zwei Timer benötigt:

- Timer für die PWM-Pins, dieser wird durch die Arduino-Bibliothek automatisch initialisiert
- Timer für den Musterschritt bei höchstem Tempo (15,625 ms), dieser muss manuell initialisiert werden, siehe Abbildung 32 unten

```
void timer_init() {
  //PWM Timer5
  TCCR5A = 0;
  TCCR5B |= (1 << WGM52); //Setze Modus "Clear Timer on Compare Match"
  TCCR5B |= (1 << CS51); //Setze Prescaler 1/64 clock
  TCCR5B |= (1 << CS50);//Setze Prescaler 1/64 clock
  //0F41 Wert für das "Compare-Register" for 15,625 ms,
  //Formel im Manual ATMega2560 Seite 145 FF.
  uint16_t cmpmatchvalue = 0x0F41;
  //Setze den Wert in das Register HIGH BYTE und LOW BYTE
  OCR5AH = (cmpmatchvalue >> 8);
  OCR5AL = (cmpmatchvalue && 0xFF);
  //Aktiviere Interrupt bei erreichen des Vergleichswertes
  TIMSK5 |= (1 << OCIE5A);
  //Aktiviere die Interrupts
  sei();
  return;
}
```

Abbildung 32: Timer Initialisierung

Nach dieser Initialisierung wird alle 15,625 Millisekunden die Interrupt-Service-Routine „ISR(TIMER5_COMPA_vect)“ des Timers ausgeführt, um das Flag „timerFlag“ zu setzen, welches die Ausführung der Funktion „setzePWMPins“ für das Setzen des Musterzustands veranlasst.



Nach den Initialisierungen laufen die Abfrage nach neuen Bytes im Com-Puffer sowie die Abfrage nach dem gesetzten timerFlag als Loop. Eine Abbildung dazu ist unten zu finden (Abbildung 33). Dabei wird bei der empfangenen Befehlskette zuerst auf die zwei Authentifizierungsbytes geprüft; sollten diese vorhanden sein, wird die Funktion „empfangeBefehlskette“ ausgeführt und die Byteauswertung wird fortgesetzt.

Abbildung 33: Programmablaufplan

Die Funktion „empfangeBefehlskette“ prüft auf zwei weitere Bytes im Puffer mit der Funktion „checkForByte“. Anschließend findet das Prüfen der Bytes auf gängige Werte nach dem aufgestellten Protokoll statt. Zugelassene Einstellungen werden dann in der Funktion „setzeArraydaten“ in einem Array für die Aktoreinstellungen gespeichert. Diese Einstellungen werden in der Funktion „setzePWMPins“ abgerufen und je nach Zustand werden die Aktorpins mit Hilfe der Funktion „analogWrite“ gesetzt.

Der vollständige Programmcode ist in den Anlagen, Teil 5 (Seite 78) zu finden.

Das Testsystem

Über eine GUI sollen am PC alle Einstellungen des Aktorsystems änderbar sein, um die Funktionalität prüfen zu können. Dazu werden vorgefertigte Tools von WindowsForms verwendet, um Auswahllisten, Slider, Checkboxen und andere Steuerelemente grafisch zu implementieren. Nach einer Auswahl werden die entsprechenden Bytefolgen für eine Einstellung gemäß Kommunikationsprotokoll über die serielle Schnittstelle an das Aktorsystem gesendet. Abbildung 34 zeigt die GUI. Die Software des Testsystems ist in den Anlagen, Teil 6 (Seite 84) zu finden.

Abbildung 34: Testsystem - GUI

Im Bereich „Aktorauswahl“ können alle zehn Taktoren ausgewählt werden. Sobald eine Einstellung geändert wurde, wird die entsprechende Befehlskette an die ausgewählten Taktoren gesendet.
Eine ausführliche Beschreibung der einzelnen Funktionen des Aktorsystems ist im Kapitel 4.2 zu finden.

6 Aufbau und Inbetriebnahme

Dieses Kapitel befasst sich mit dem Aufbau und der Inbetriebnahme des Detektionssystems sowie des Aktorsystems. Dabei werden Komponenten systemgruppenweise in Betrieb genommen. Anschließend wird die Kommunikation zwischen den zwei Systemen geprüft und das komplette Warnsystem in Betrieb genommen.

6.1 Detektionssystem

Um das Detektionssystems in Betrieb nehmen zu können, wird zuerst die GUI getestet und anschließend die Anbindung der Sensoren.

Test der GUI

Die GUI soll auf einem externen Gerät laufen, um auch während des Betriebs des Detektionssystems (integriert in einer Weste) Änderung vornehmen zu können. Entscheidend ist hierbei, dass sich das externe Gerät und der Mini-PC des Detektionssystems im gleichen Netzwerk befinden müssen. Dabei sollte der Mini-PC drahtlos angebunden sein, das externe Geräte über Lan oder W-Lan.
Zuerst muss die IP-Adresse des externen Geräts ermittelt werden. Diese ist unter Windows im Command mit dem Befehl „ipconfig -all“ abrufbar und unter „IPv4-Adresse“ des benutzten Netzwerkadapters zu finden (Abbildung 35).

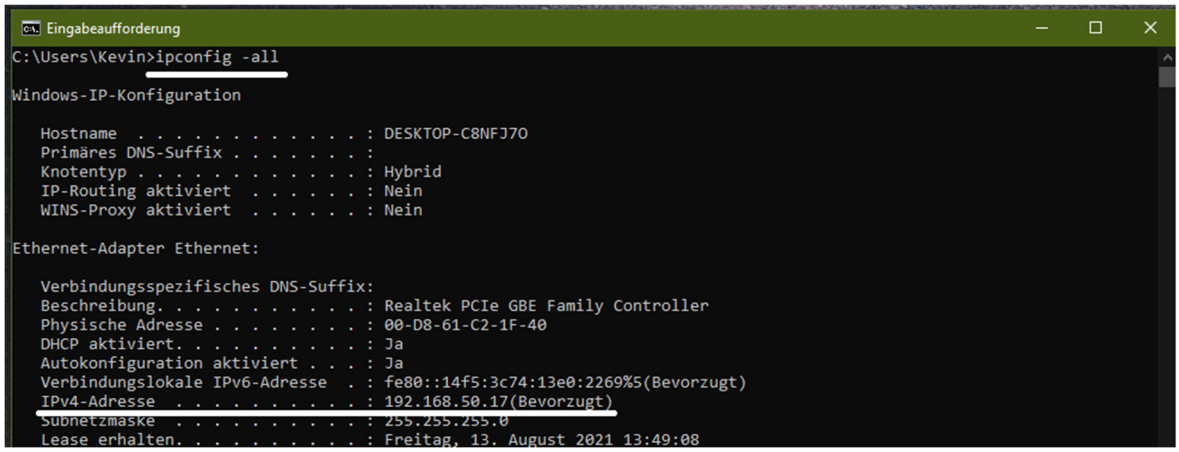


Abbildung 35: Grafische Darstellung des „ipconfig -all“-Befehls

An den Mini-PC des Detektionssystems ist kein Bildschirm und auch keine Maus/Tastatur angeschlossen. Die IP-Adresse des externen Geräts muss in der Software des Detektionssystems eingetragen werden. Der Zugriff auf den Mini-PC zum Einstellen der richtigen IP-

Adresse und anschließenden Starten der Software erfolgt über „Windows Remotedesktopverbindung“.

Hier muss zuerst die IP-Adresse des Mini-PCs eingegeben werden und folgend Benutzername sowie Passwort. Die IP-Adresse kann über den Router des Netzwerks oder über einen IP-Scanner herausgefunden werden.

Benutzername: 3dsys_TestPC
Passwort: 335123xY

Nach erfolgreicher Authentifikation erscheint der Windows-Desktop des Mini-PCs. Im Ordner „KombiniertesProtektorProgrammGUI“ kann nun die Datei „ConsoleApplication1.sln“ gestartet werden. Im File „ConsoleApplication1.cpp“ kann die Variable „ipAdress_local“ auf die IP-Adresse des externen Geräts gesetzt werden (Abbildung 36).

```
71    //weiterer Code zur Verbindung mit der GUI
72    string ipAddress_local = "192.168.50.17";          // IP Address of the server
73    #define portGUI 50000
74    char bufGUI[60];
75    bool* newdataGUI = new bool;
```

Abbildung 36: Grafische Darstellung des Programabschnittes zur IP-Adressen-Änderung

Die GUI kann über das externe Gerät gestartet werden, indem die Datei „main.exe“ ausgeführt wird. Dabei muss sich die Datei „test.ui“ im selben Ordner befinden.

Zum Überprüfen der Funktionalität der GUI wird das Detektionssystem und die GUI gestartet. In der Software des Detektionssystems wurden zuvor alle Threads der Sensoren und der depthViewer-Thread auskommentiert. Nach dem Drücken des Buttons „OK“ findet eine Kommunikation mit dem Detektionssystem statt. In der Software wurde durch Debugging überprüft, dass die neuen Variablenwerte empfangen wurden. Die GUI und deren Kommunikation ist einsatzbereit.

Einzeltest der Kamera und Radar

Die Kamera sowie die zwei Radarsensoren müssen über USB mit dem Mini-PC des Detektionssystems verbunden werden. Anschließend kann für das Debugging im Detektionssystem der GUI-Thread auskommentiert werden. Nach dem Start der Software findet automatisch die Initialisierung der Sensoren statt. Sollte die Software die Initialisierungen erfolgreich abschließen, beginnen die Endlosschleifen der Messroutinen. Die Funktionalität wurde durch Setzen und Erreichen von Debugpunkten in den Messroutinen bestätigt.

Zusammensetzen des Systems in der Weste

Abbildung 37: Sensor-Sockel

Nachdem die Funktionalität der Software des Dektektionssystems geprüft wurde, müssen nun der Mini-PC sowie die Sensoren in die Weste integriert werden. Dazu wird die Weste aufgeschnitten und die Sockel (Abbildung 37) werden in das Futter der Weste eingesetzt, damit die Radar-Sensoren geneigt werden können.

Abbildung 38: Radar-Sensoren auf Sockel

Anschließend werden die Radarsensoren angebracht und mit einem USB-Kabel verbunden (Abbildung 38).

Die Sensoren sind über einen Klettstreifen mit den Sockeln verbunden. Der Stoff der Weste kann entsprechend über die Sockel und hinter die Radar-Sensoren gelegt werden.

Es folgt nun das Anbringen und Verbinden der Kamera. Diese wird unterhalb der Radar-Sensoren platziert. Auf der Oberseite der Kamera wurden Klebestreifen angebracht, da nur die Linse außerhalb der Weste sichtbar sein muss.

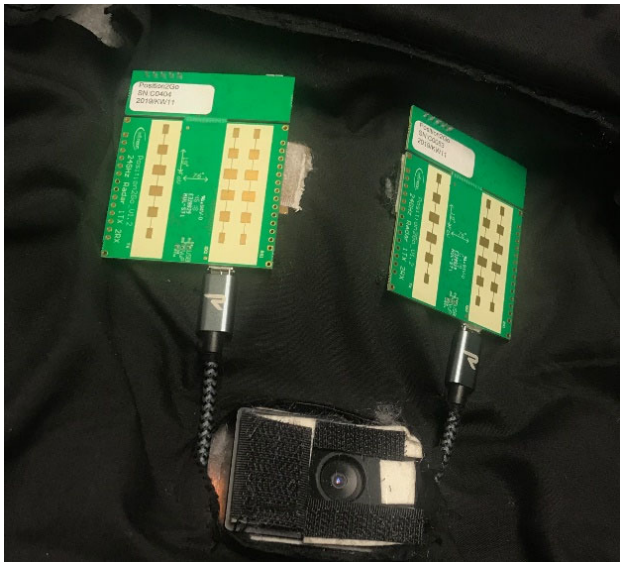


Abbildung 39: Westenansicht der Sensoren

Abbildung 39 zeigt die sichtbaren Bestandteile des Aufbaus nach dem Anbringen des Westenstoffs. Weitere Untersuchungen ergaben, dass die Radarsensoren durch den Stoff der Weste nicht beeinflusst werden. Somit kann der Westenstoff über die Radarsensoren gelegt werden.

Abbildung 40 zeigt den fertigen Aufbau an der Rückseite der Warnweste.

Abbildung 40: Westenansicht hinten

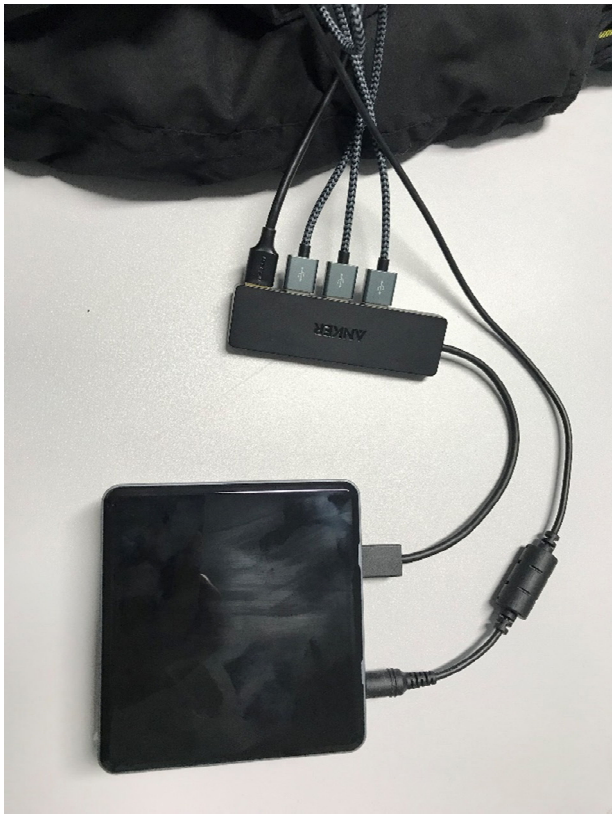


Abbildung 41: Anschluss des Mini-PCs

Der Mini-PC des Detektionssystems wird über einen USB-Hub mit den zwei Radar-Sensoren, der Kamera und dem Arduino des Aktorsystems verbunden (Abbildung 41).

Der Stromanschluss erfolgt über die Akkus, welche in den Kapiteln des Aktorsystems beschrieben wurden. Der Mini-PC wird mit allen Kabeln in der linken vorderen Westentasche verstaut. Dabei ist auf eine ausreichende Belüftung zu achten, da die Leistung der CPU bei thermischen Problemen gedrosselt wird. Die Lüftungslöscher des Mini-PCs müssen nach oben zeigen und die Tasche muss geöffnet bleiben (Abbildung 42).

Abbildung 42: Mini-PC in der Westentasche

6.2 Aktorsystem

Inbetriebnahme des Aktorsystems

Es folgt in diesem Kapitel die Inbetriebnahme des Aktorsystems. Zuerst wird das Arduino-Shield bestückt. Die SMD-Bauteile werden mit einer Pick&Place-Maschine nach dem Auftragen der Lötpaste aufgesetzt. Nach dem Abkühlen werden die Stift- und Klemmleisten angelötet; es folgt nun ein Kurzschlusstest, um Kurzschlussströme zu vermeiden. Abbildung 43 zeigt das fertig gelötete Board (links) mit der Arduino-Plattform (rechts).

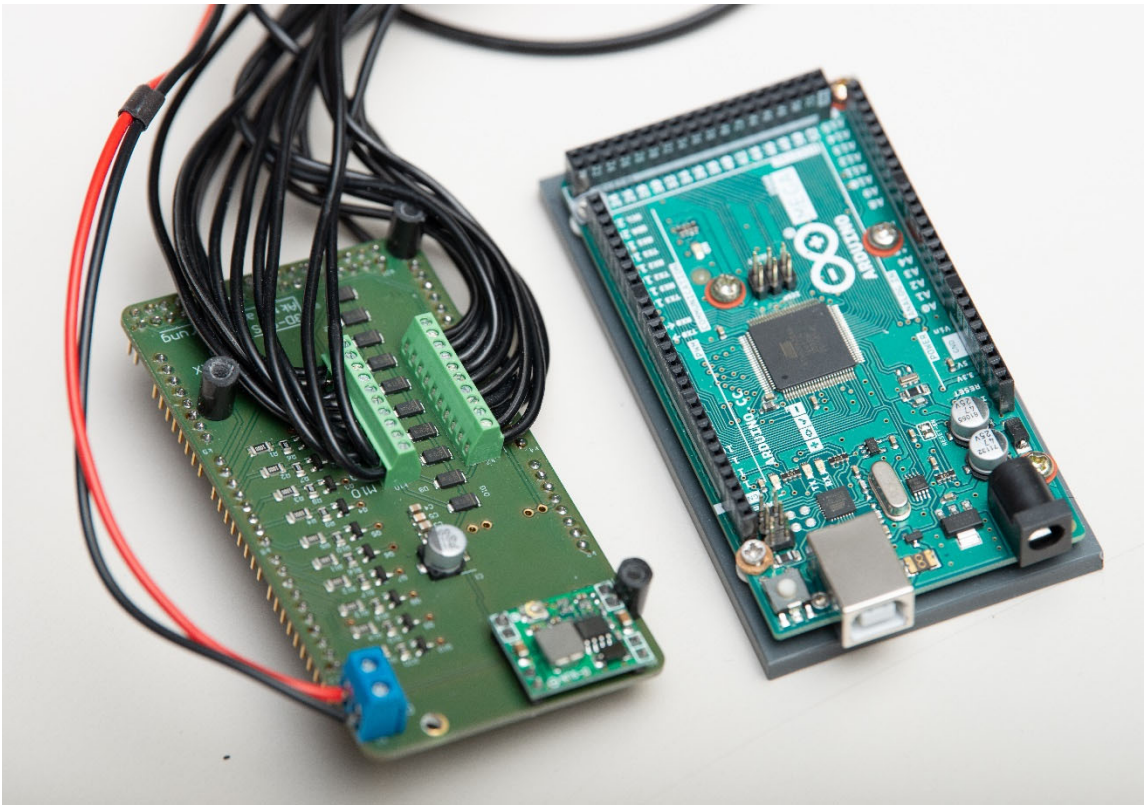


Abbildung 43: zu sehen - Links Arduino-Shield, Rechts Arduino ATMega2560

Im linken unteren Teil des Arduino-Shields ist ein roter und ein schwarzer Draht zu sehen. Hier ist die externe Spannungsversorgung angeschlossen. Diese speist den DC/DC-Wandler der Aktorik, welcher sich rechts neben dem externen Spannungsanschluss befindet. Der DC/DC-Wandler für die Arduino-Plattform wurde nicht mit aufgelötet, da diese über den USB-Anschluss gespeist werden soll. Die zehn Taktoren sind bereits an den Schraubklemmen befestigt. Nun wird das Shield mit Hilfe der Stiftleisten auf der Rückseite (Abbildung 44) auf die Arduino-Plattform aufgesetzt.

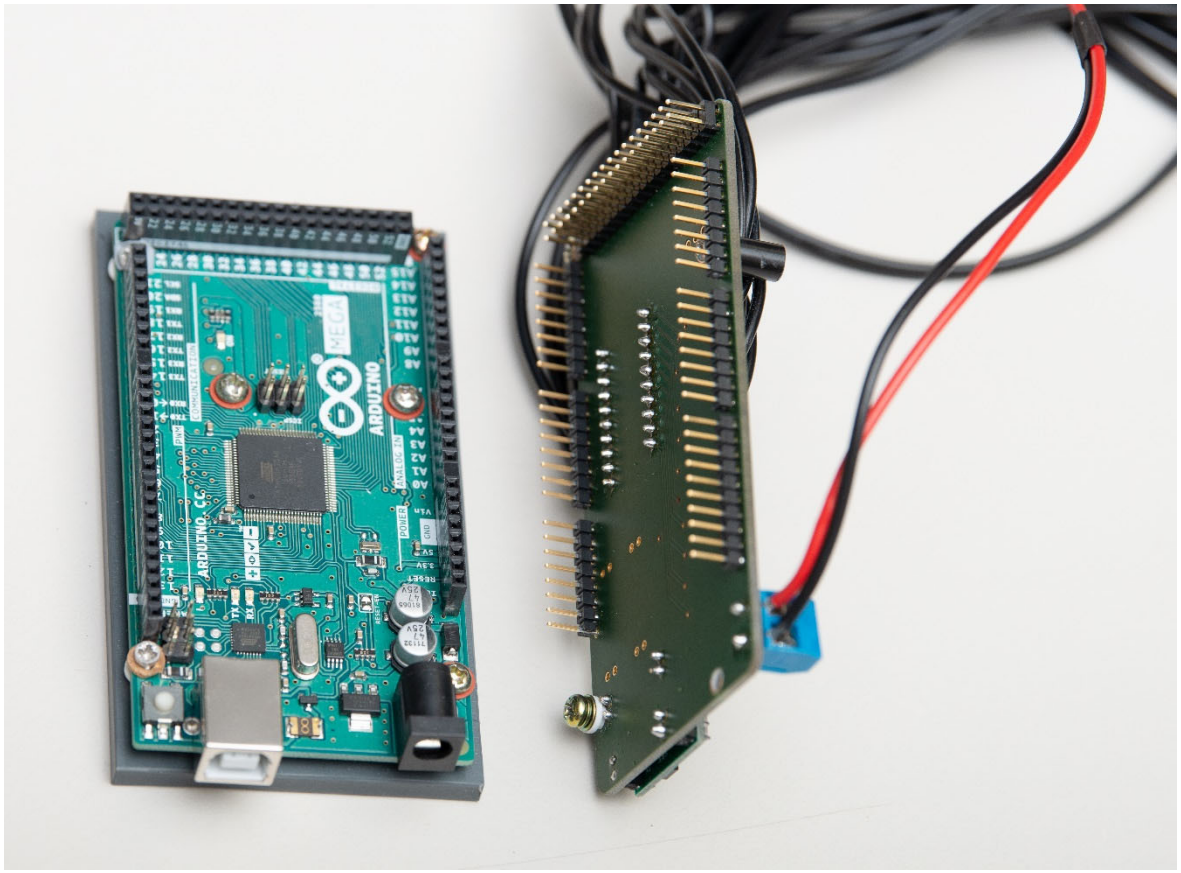


Abbildung 44: Links Arduino ATMega2560, Rechts Arduino-Shield Unterseite

Um das fertige System vor mechanischen Einflüssen zu schützen, wird eine Schutzplatte aufgesetzt; anschließend wird der Programmcode heruntergeladen.

Es werden für das Debugging einzelne Funktionen der Software zur Inbetriebnahme auskommentiert und mehrere „Serial.println(„Debugpunktxx“)“ eingefügt, um das erwünschte Durchlaufen bzw. den Ablauf der Programmschritte zu überprüfen. Mit der Software „hterm“ können die seriellen Debugpunkte empfangen und ausgewertet werden. Nach dieser Überprüfung wurden mit Hilfe des Testsystems alle Einstellungen des Aktorsystems auf Funktion getestet. Das Aktorsystem ist einsatzbereit. Abbildung 45 zeigt das fertige Aktorsystem mit einem Akkumulator als Spannungsversorgung. Alle Anforderungen an das System sowie das Hardware- als auch das Softwarekonzept wurden erfolgreich umgesetzt. Das Aktorsystem kann über die serielle Schnittstelle mit Hilfe des Kommunikationsprotokolls angesteuert werden, damit eine Vibration ausgelöst wird.

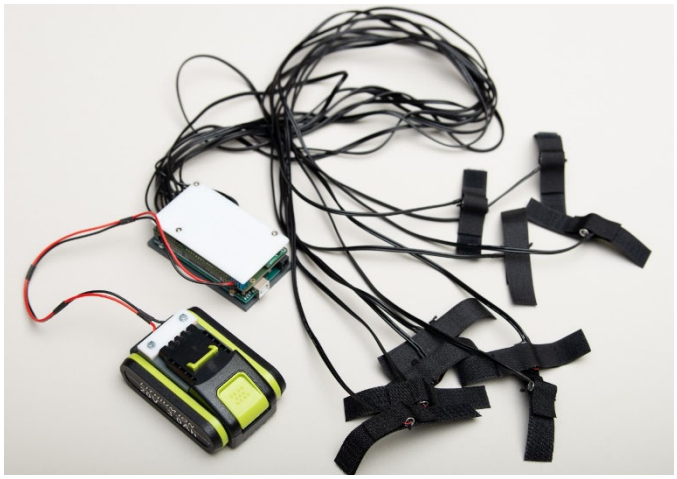


Abbildung 45: fertiges Aktorsystem

Zusammensetzen des Systems in der Weste

Abbildung 46: Taktoren integriert in die Weste

Die Taktoren werden, wie in Kapitel 5.2.1.2 beschrieben, in die Weste integriert. Dabei wird das Gehäuse der Taktoren in das Innenfutter der Weste eingenäht (Abbildung 46). Abbildung 47 zeigt die Weste von innen. Im Hüftbereich wurde ein Gürtel angebracht, welcher mit einem Klettverschluss geschlossen werden kann. An dem Gürtel wird die untere Reihe der Taktoren installiert. Für spätere Betrachtungen ist der Gürtel sinnvoll, da hier ein höherer Anpressdruck erreicht wird, wodurch die Vibrationsintensität besser wahrgenommen werden kann.

Abbildung 47: Westenansicht - innen

Die Arduino-Plattform des Aktorsystems wird in die rechte vordere Tasche der Weste verpackt. Die zwei Akkumulatoren werden jeweils in die vordere linke und rechte Brusttasche gepackt. Abbildung 48 zeigt eine Anordnung der Hardware auf der Vorderseite der Weste.

Abbildung 48: Westenansicht Front

6.3 Funktionstest des Warnsystems

Um die Anforderungen aus Kapitel 1.2 zu überprüfen, werden einige Messungen und Tests durchgeführt. Tabelle 17 zeigt die Anforderungen.

Tabelle 17: Anforderungen an das System

Aktive Flächen / Taktoren	10 Stück
Abstand Taktor zu Taktor (Mittelpunkt)	>= 5 cm
Taktoranordnung	sinnvoll für räumliche Wahrnehmung
Maximale Leistung	300 W
Reaktionszeit	<= 25 ms
Gewicht	<= 1 kg

Es wurden 10 Taktoren in die Weste integriert, der Abstand beträgt ca. 8 Zentimeter. Es wurden jeweils fünf Aktoren in einer Reihe angeordnet, um ein räumliches Wahrnehmen mit Winkelschritten von 28° zu ermöglichen – die Anforderung mindestens rechts und links unterscheidbar zu machen ist somit erfüllt.

Die Reaktionszeit des Warnsystems ist die Zeit vom Empfangen der Messdaten im Detektionssystems bis hin zur Vibration über die Taktoren des Aktorsystems. Folgend werden einige Zeiten tabellarisch dargestellt (Tabelle 18):

Tabelle 18: Darstellung der Timings des Warnsystems

Vorgang	Zeit
Senden Messbefehl, Messung, Antwort vom Radar-Sensor	56 ms
Verarbeitung der Daten im Main-Thread	2 ms
Sendezeit an das Aktorsystem	1 ms
Verarbeitung im Aktorsystem, Ausgabe auf Taktorik	1 ms

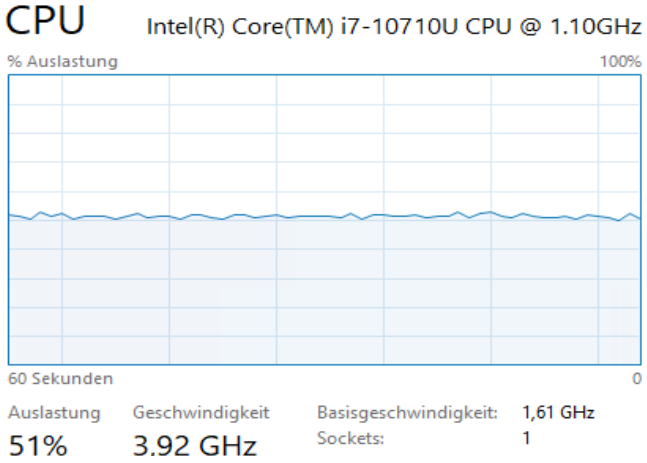
Die Zeiten des Detektionssystems wurden mit Hilfe der „clock()“-Funktion ausgelesen. Dabei wird jeweils am Anfang und zum Ende des Vorgangs die Zeit gelesen und anschließend voneinander subtrahiert. Abbildung 49 zeigt ein Beispiel im Radar-Thread. Die Variable „timeInSeconds“ enthält anschließend die gemessene Zeit.

```
378
379     for (modul = 0; modul < 2; modul++) {
380         clock_t startTime = clock();
381         ep_radar_base_get_frame_data(protocolHandle[modul], endpointRadarBase[modul], 1);
382         ep_targetdetect_get_targets(protocolHandle[modul], endpointRadarTarget[modul]);
383         while (warten);
384         clock_t endTime = clock();
385         clock_t clockTicksTaken = endTime - startTime;
386         double timeInSeconds = clockTicksTaken / (double)CLOCKS_PER_SEC;
387         warten = 1;
```

Abbildung 49: Darstellung des Programmcodes zur Zeitmessung

Das Warnsystem benötigt für das Bearbeiten der Daten, das Senden und die Ausgabe über das Aktorsystem 4 Millisekunden. Wir befinden uns somit innerhalb der Vorgabewerte.

Das Auslösen des Messbefehls, die anschließende Messung und die Antwort des Sensors nehmen jedoch 56 Millisekunden in Anspruch. Die Zeit ist vom Sensor abhängig und kann nur durch Tausch des Sensors bzw. durch anpassen der Messparameter, z.B. Verringerung der Messqualität, erreicht werden. Für einen Funktionstest ist das Timing des Prototypen ausreichend.



Die Auslastung der CPU des Mini-PCs liegt nach einem Dauertest von 10 Minuten bei ca. 51% (Abbildung 50). Es ist davon auszugehen, dass alle Threads des Detektionssystems ohne Leistungseinbußen betrieben werden.

Abbildung 50: Auslastung der CPU des Mini-PCs

Das Warnsystem benötigt im Durchschnitt 2,5 Ampere bei 20 Volt. Die Anforderung von maximal 300 Watt ist somit erfüllt. Mit den 2 verwendeten Akkumulatoren besitzt das System aktuell eine maximale Laufzeit von 144 Minuten.

Das Gewicht der Weste beträgt 2,85 Kilogramm und übersteigt das gewünschte Gewicht von einem Kilogramm aus den Anforderungen. Dabei nehmen der Mini-PC mit 496 Gramm und die beiden Akkumulatoren mit 691 Gramm nehmen vom Warnsystem das meiste Gewicht ein.

Es folgt ein Kurztest der Warnweste. Dazu trägt ein Proband die Weste und bei näherkommenden bzw. nahen Objekten wird in der entsprechenden Richtung eine Vibration ausgelöst. Die Funktion der Weste wurde nachgewiesen.

7 Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde ein Warnsystem entwickelt, welches durch vibrotaktile Aktorik auf näherkommende Objekte außerhalb des Gesichtsfeldes reagieren kann. Das System wurde für das Werkarbeiterumfeld entwickelt und entsprechend in eine Warnweste integriert. Wichtige Systemvariablen können mit Hilfe einer GUI auf einem externen Gerät während des Betriebs abgeändert werden. Das Warnsystem wurde in ein Detektionssystem, welches für die Sensoren, Sensordaten und dessen Verarbeitung sowie eine Gefährdungsanalyse zuständig ist und in ein Aktorsystem, welches für die Ansteuerung der Taktoren zuständig ist, untergliedert.

Für das Detektionssystem wurden Tests mit den Sensoren durchgeführt, um deren Verwendbarkeit zu prüfen. Anschließend wurden Algorithmen zur Sensordatenverarbeitung und für die Gefahrenanalyse aufgestellt und in Programmcode umgesetzt. Um eine Bearbeitung der Daten in Echtzeit zu ermöglichen, wurden Teilaufgaben des Programms in Threads unterteilt, welche nebenläufig ablaufen.

Damit die Informationen der Gefahrenanalyse an den Werkarbeiter über die Taktoren weitergeleitet werden können, wurde das Aktorsystem entwickelt. Dieses empfängt Richtungsinformationen mit Intensitätsmerkmalen vom Detektionssystem und gibt diese auf die 10 Aktoren aus. Dabei wurden zwei Reihen untereinander mit jeweils 5 Aktoren horizontal angeordnet, die einen Abstand von 8 Zentimetern zueinander aufweisen.

Es wird zwischen einer Annäherungswarnung auf der oberen Reihe der Aktoren und einer Entfernungswarnung auf der unteren Reihe der Aktoren unterschieden. Die Aktoren der unteren Reihe werden zusätzlich durch einen Gürtel an die Hüfte gedrückt.

Es wurden im Außenbereich Tests mit der Warnweste durchgeführt, dabei haben näherkommende Objekte mit einer Geschwindigkeitsüberschreitung eine Vibration ausgelöst. Die Entfernungswarnung wurde durch Annäherung an eine Wand ebenfalls geprüft.

Eine Untersuchung in der Zukunft könnte wichtige Rückschlüsse auf den Einsatz bzw. Notwendigkeit einer Entfernungswarnung zulassen. Weiterhin könnte eine Analyse der Wahrnehmung der Vibrationsintensitäten Rückschlüsse auf die bessere Position der Aktor-Reihe zulassen (Hüfte oder oberer Rücken). Es sollte auch die Notwenigkeit beider Aktor-Reihen überdacht werden, eventuell ist eine Kombination der Reihen, also Kombination von Entfernungswarnung und Geschwindigkeitswarnung, durch verschiedene Vibrationsmuster möglich.

Es wurden zwei Westen entwickelt mit jeweils verschiedenen Taktor-Typen mit verschiedenen Vibrations-Intensitäten. Tests in der Zukunft mit verschiedenen Kleidungen unterhalb der Weste (Sommerkleidung – dünn, Winterkleidung – dick) können Rückschlüsse auf die Verwendung der beiden verbauten Taktor-Typen zulassen.

Die Radar-Sensoren wurden in dieser Arbeit für die Annäherungswarnung verwendet. Die ToF-Kamera wurde für die Entfernungswarnung verwendet. Weitere Algorithmen sind denkbar, um aus den erkannten Objekten eines Tiefenbildes durch Betrachten mehrerer Aufnahmen und dessen Zeitversatz die Geschwindigkeit zu erhalten. Eine Fusionierung der Sensordaten könnte anschließend eine stabilere Detektion von Objekten und auch deren Nachverfolgbarkeit (Trajektorien) ermöglichen.

Im Vergleich zum PKW sind die Sensoren einer Warnweste ständig in 3-dimensionaler Bewegung, da sich die Neigung beim Verdrehen des Körpers (Arbeiter streckt sich, oder Arbeiter bückt sich) ständig ändert. Durch das Verwenden eines Welt-Koordinatensystems und Integration eines Westenkoordinatensystems mit Koordinatensystemen der einzelnen Sensoren und mit dem Einfügen eines 3-Achs-Gyrosensors in das System könnten solche Szenarien abgefangen und gesondert betrachtet werden.

Im Anschluss kann über die Miniaturisierung und die Steigerung der Effizienz des Systems nachgedacht werden, um das Gewicht zu verringern. Weiterhin könnte damit der Stromverbrauch reduziert werden, was wiederrum die Verwendung von nur einem Akkumulator oder eine längere Laufzeit des Systems ermöglicht.

Literaturverzeichnis

amazon.de. (2020). *Jialitt WA3551 Battery*. Abgerufen am 15. 08 2021 von https://www.amazon.de/-/en/Jialitt-Battery-WA3551-1-WA3553-1-WA3553-2/dp/B07RTB97X2

arduino.cc. (2018). *ARDUINO MEGA 2560 REV3*. Abgerufen am 15. 08 2021 von https://store.arduino.cc/arduino-mega-2560-rev3

Atmel. (Datenblatt). *microchip.com.* Abgerufen am 18. 05 2021 von https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf

Berlin-Brandenburgische Akademie der Wissenschaften. (21. 08 2020). *"Haptik - Schreibung, Definition, Bedeutung"*. Abgerufen am 18. 05 2021 von https://www.dwds.de/wb/Haptik

Beruscha, F. (2012). "Nutzerorientierte Gestaltung haptischer Signale in der Lenkung". In P. D. Augsburg, *"Kraftfahrzeugtechnische Berichte Band 5".* Technische Universität Ilmenau/Universitätsbibliothek: Universitätsverlag Ilmenau.

Birbaumer, N. (1999). Biologische Psychologie. In N. Birbaumer, *Biologische Psychologie* (S. 326 - 337). Springer.

Brell, D. M. (2009). *"Eine vibrotaktile Mensch-Maschine-Schnittstelle für chirurgische Applikationen".* Oldenburg: Carl von Ossietzky Universität Oldenburg.

E. Hering, G. S. (2012). *Sensoren in der Wissenschaft und Technik.* Springer Fachmedien Wiesbaden GmbH.

infineon.com. (2019). *DEMO POSITION2GO*. Abgerufen am 15. 08 2021 von https://www.infineon.com/cms/en/product/evaluation-boards/demo-position2go/

intel.com. (2019). *Intel® NUC 10 Leistungs-Kit - NUC10i7FNK*. Abgerufen am 15. 08 2021 von https://ark.intel.com/content/www/de/de/ark/products/188808/intel-nuc-10-performance-kit-nuc10i7fnk.html

Laser & Co. Solutions GmbH. (2005). *"Mikrocontrollerprogrammierung lernen mit myAVR"*. Abgerufen am 18. 05 2021 von http://einsteiger.myavr.de/index.php?id=5

Martin Seeger, T. S. (13. 09 2017). "Vibrotaktile Wahrnehmung bei der Verwendung von Handschuhen". *"Mensch und Computer 2017 - Tagungsband"*, S. 160-170.

Martin, C. (2000). *Spektrum.de - Gesichtsfeld*. Abgerufen am 15. 08 2021 von https://www.spektrum.de/lexikon/biologie/gesichtsfeld/27780

Martin, C. (2000). *Spektrum.de - Rechnersysteme*. Abgerufen am 15. 08 2021 von https://www.spektrum.de/lexikon/geowissenschaften/rechnersystem/13278

Mohr, S. (2019). *"Diplomarbeit über das Thema Prototypische Entwicklung eines taktilen Warnsystems für autonome Fabriken".* Dresden: Hochschule für Technik und Wirtschaft Dresden.

München, T. U. (Hrsg.). (2010). Abgerufen am 15. 08 2021 von Lehrstuhl I05: https://www5.in.tum.de/lehre/seminare/math_nszeit/SS03/vortraege/radar/technik.html

reichelt.com. (2019). *DEBO DCDC DOWN 4 Entwicklerboards - Spannungsregler, DC/DC-Wandler*. Abgerufen am 15. 08 2021 von https://www.reichelt.com/de/en/developer-boards-voltage-regulator-dc-dc-converter-debo-dcdc-down-4-p291460.html?r=1

robot-electronics.co.uk. (2020). *SRF10 Ultrasonic range finder*. Abgerufen am 15. 08 2021 von http://www.robot-electronics.co.uk/htm/srf10tech.htm

Schmidt, D. D. (2007). *Physiologie des Menschen 30. Auflage.* (S. Verlag, Hrsg.) Würzburg.

Simon Schätzle, T. E. (15. 09 2010). "Vibrotac: An Ergonomic And Versatile Usable Vibrotactile Feedback Device". *"19th IEEE International Symposium on Robot and Human Interactive Communication Principe di Piemonte"*, S. 670-675.

Speth, D.-I. U. (2009). *Promotionsarbeit zur Videobasierte modellgestützte Objekterkennung.* München: TU München.

terabee.com. (2018). *Terabee 3Dcam 80x60 - the compact, robust and affordable 3D Time-of-Flight camera*. Abgerufen am 15. 08 2021 von https://www.terabee.com/shop/3d-tof-cameras/terabee-3dcam/

Torex. (Datenblatt). *torexsemi.com.* Abgerufen am 18. 05 2021 von https://www.torexsemi.com/file/xp233n0501tr/XP233N0501TR.pdf

Wikipedia. (15. 08 2021). *Radar*. Von https://de.wikipedia.org/wiki/Radar abgerufen

Wikipedia. (2021). *TOF-Kamera*. Abgerufen am 15. 08 2021 von https://de.wikipedia.org/wiki/TOF-Kamera

Wikipedia (Hrsg.). (2021). *Ultraschall*. Abgerufen am 15. 08 2021 von https://de.wikipedia.org/wiki/Ultraschall

Wolff, C. (1977). *radartutorial.eu*. Abgerufen am 15. 08 2021 von https://www.radartutorial.eu/02.basics/Frequenzmodulierte%20Dauerstrichradarger%C3%A4te.de.html

Anlagen

Anlagen, Teil 1

Kurz Vorhabensbeschreibung – Projekt „3Dim-Hapt“

Prof. Dr.-Ing Michael Kuhl

Die Arbeitsziele fokussieren auf die Entwicklung eines Systems zur Aufnahme von 3D-Umgebungsinformationen und deren Umsetzung in taktile Informationen in einem mobilen/tragbaren System, z.B. zur Vermeidung schwerer Arbeitsunfälle mit Werksverkehr.

Das System muss dabei

- die Umgebung aufnehmen und Objekte separieren
- mögliche Trajektorien der Objekte analysieren
- sie in gefährlich/ungefährlich kategorisieren
- haptisches Feedback an die gefährdete Person/den Verursacher geben
- zusätzlich Gefährdungsintensitäten oder Richtungsinformationen signalisieren

Dabei sind folgende Teilbereiche zu untersetzen:
Objekterkennung, Trajektorienberechnung sowie die Gefährdungskategorisierung müssen genau und abgesichert arbeiten. Hierzu werden verschiedene Verfahren und Algorithmen getestet und kombiniert. Eine Datenfusion mehrerer physikalisch unterschiedlich arbeitender Sensoren erscheint am aussichtsreichsten, muss jedoch auch ein leichtes und energiesparsames System ermöglichen.

sichere Informationsverarbeitung und -weitergabe unter Beachtung wesentlicher Randbedingungen wie Energieaufnahme, hohe Verfügbarkeit, Verarbeitungsgeschwindigkeit etc.

Die durch das System zu erreichenden technischen Zielparameter sind in folgender Tabelle gegeben. Die Parameter dienen als Forschungsvorgaben für eine sinnvolle Performance und basieren auf entsprechenden Voruntersuchungen und Abschätzungen.

Tabelle 1: Technische Zielparameter (ca. Werte)

Technischer Parameter	**Zielparameter**
Arbeitsraum	
Arbeitsraumtiefe t Arbeitsraumhöhe h	t_{min} = 3000 mm h_{min} = 2000 mm
Arbeitsraumwinkel (bei Sensorgröße opt. 1/3“ 4,8 mm) α = horizontal β = vertikal	 α_{min} = 37° β_{min} = 37°
Allgemeine Systemauslegung	
Informationsaufnahme	An einer Person (Gefährdeter) An einem Fahrzeug (Gefährder)
Lokalisierung der Informationsausgabe	Direkt am Gefährdetem (Person) Direkt am Gefährder (Person)
Art der Informationsausgabe	Vibration | Frequenz / Druck-Intervalle - Signalisierung von Gefahr von hinten - hinten / rechts-links-Orientierung (nach Ergbn.)
Zeitliche Reaktion gesamt t_{ges}	$t_{ges} \leq$ 800 ms
Stromaufnahme I	I_{max} = 5000 mA
Gewicht gesamt (für Personen) m_{gesP} Gewicht gesamt (für Transportsysteme) m_{gesT}	m_{gesP} < 1000 g (incl. Energieversorg.), 1. Interation m_{gesT} < 5000 g
Aufnahmesystem	
Auflösung A (bei (5*5) mm^2 Objektauflösung je Pixel)	$A_{min} \geq$ 400 x 600 Pixel
Bildrate (optisch oder adäquat) Zeit Datenakquisition	$f \geq 20\ s^{-1}$ (FPS) $t \leq$ 50 ms
Warnsystem	
Anzahl aktive Flächen a (bei mind. Orientierung der Gefahr rechts/links)	$a_{min} \geq 2$
Fläche der aktiven Einzelflächen A A = (Länge * Breite) (zur Erreichung einer adäquaten Wahrnehmung bei Anbringung der Aktoren im Rückenbereich)	$A \geq (50 * 50)\ mm^2$
Zeitliche Reaktion	$t_{min} \leq$ 25 ms

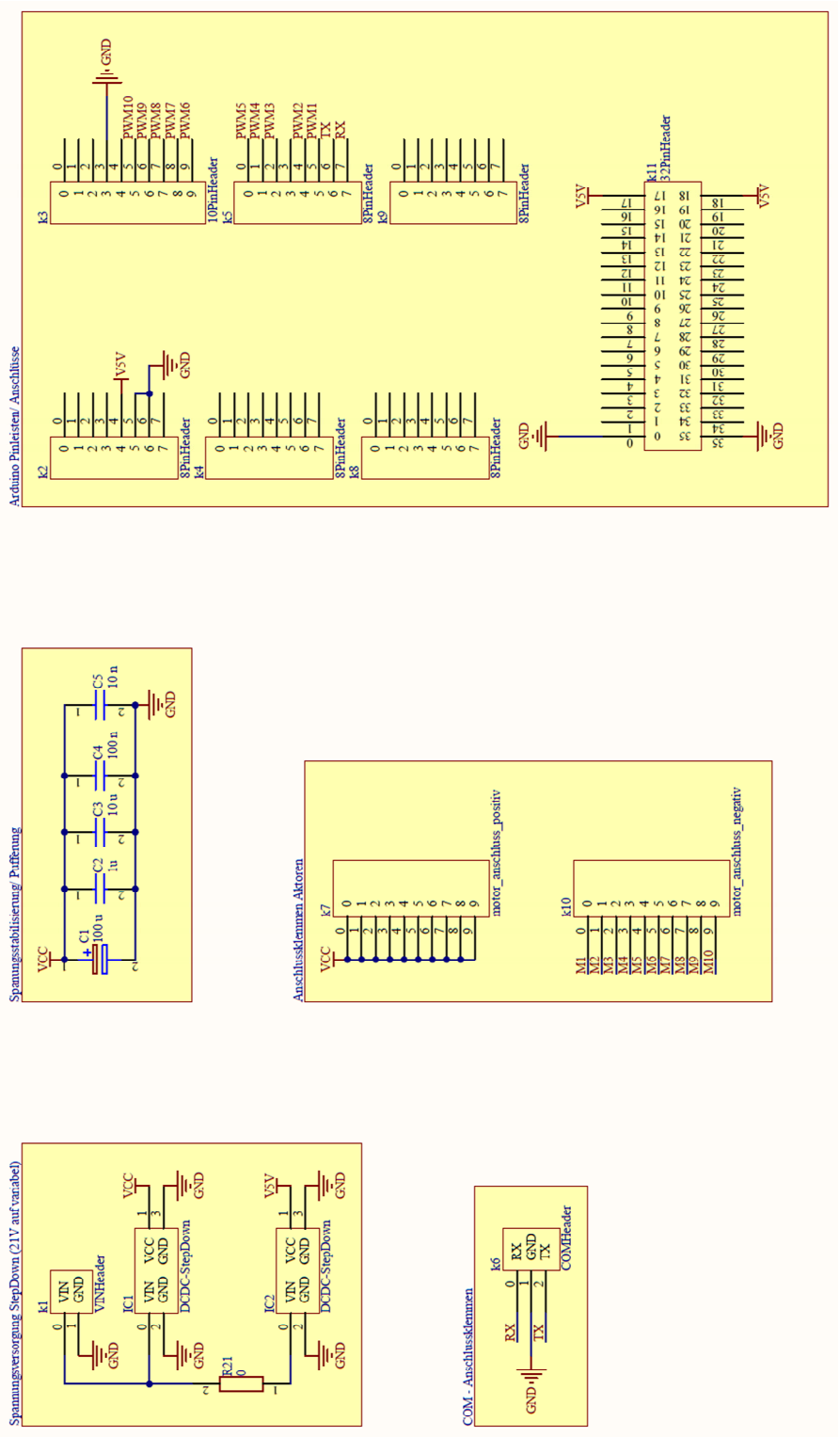
Anlagen, Teil 2

Software zur Zustandsumschaltung HIGH auf LOW und LOW HIGH mit dem Arduino ATMega 2560

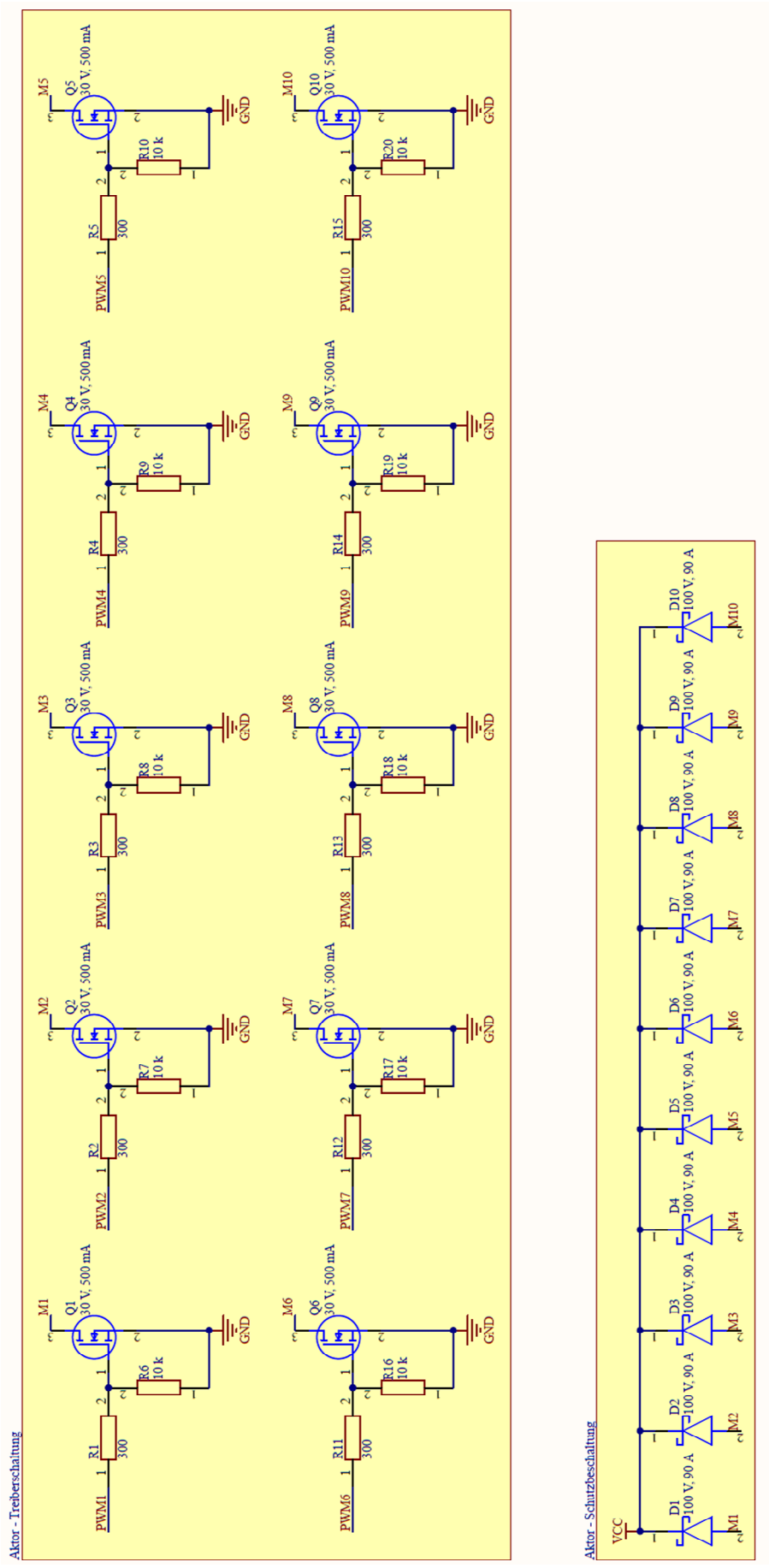
```
int Pin5 = 5;

void setup() {
  // declare pin 5 to be an output:
  pinMode(Pin5, OUTPUT);
}

void loop() {
  // set the pin 5 LOW:
  analogWrite(Pin5, 0);
  //wait 100 ms
  delay(100);
   // set the pin 5 HIGH:
  analogWrite(Pin5, 255);
   //wait 100 ms
  delay(100);
}
```

Anlagen, Teil 3

Schaltplan

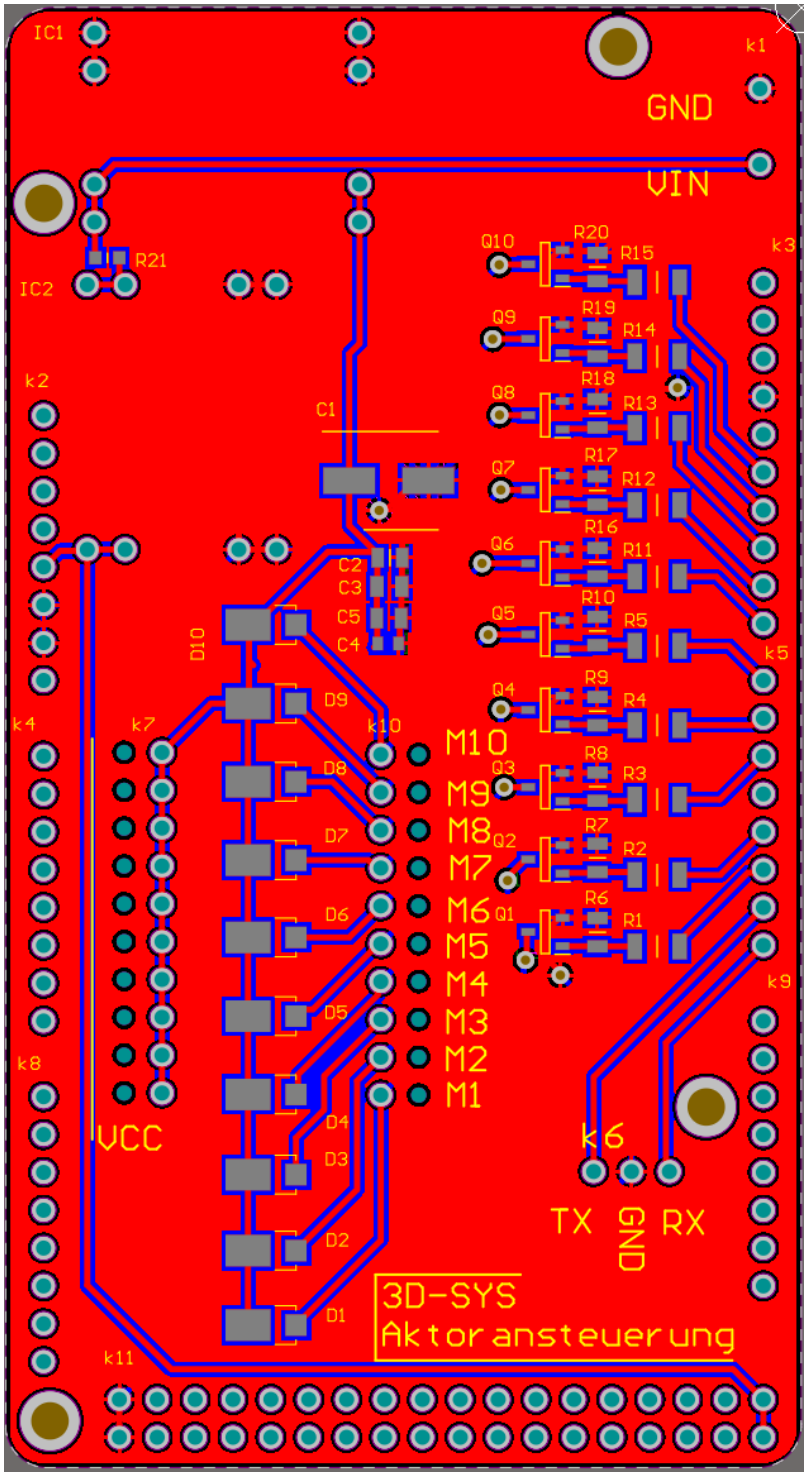


Anlagen, Teil 3



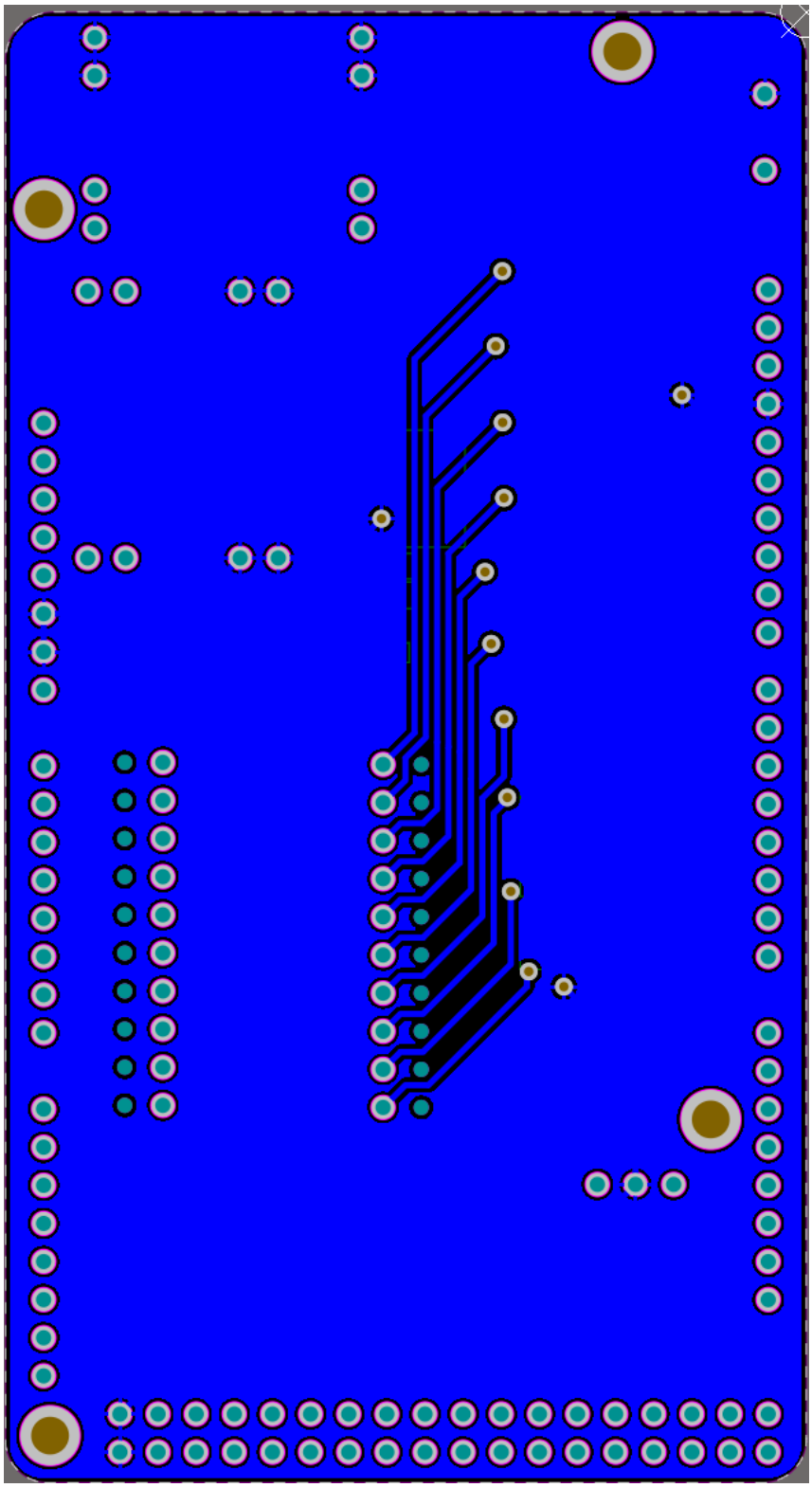
Anlagen, Teil 4

Platinenlayout TOP Layer



Anlagen, Teil 4

Platinenlayout BOTTOM Layer



Anlagen, Teil 5

Software für Arduino ATMega 2560

```
//defines
#define Anzahl_Aktoren  10
#define Anzahl_Einstellungen 6

#define Einstellung_Muster 0
#define Einstellung_Geschwindigkeit 1
#define Einstellung_PWM 2
#define Einstellung_Dauermodus 3
#define Einstellung_Aktiviert 4
#define Einstellung_PIN 5
#define Aktorauswahl 6

#define Muster_0 0x40 //Akzent
#define Muster_1 0x41 //Herzschlag
#define Muster_2 0x42 //Hinweis
#define Muster_3 0x43 //Dauerhaft
#define Muster_4 0x44 //Staccato
#define Muster_5 0x45 //Symphonie

#define Geschwindikeit_0 0x30
#define Geschwindikeit_1 0x31
#define Geschwindikeit_2 0x32
#define Geschwindikeit_3 0x33
#define Geschwindikeit_4 0x34
#define Geschwindikeit_5 0x35
#define Geschwindikeit_6 0x36
#define Geschwindikeit_7 0x37
#define Geschwindikeit_8 0x38
#define Geschwindikeit_9 0x39

//deklarationen
volatile uint8_t aktorarray[Anzahl_Aktoren][Anzahl_Einstellungen];

volatile long myTimer = 0;
volatile long myTimeout = 50;

volatile uint16_t aktor = 0x00;
volatile uint8_t timerFlag = 0;
volatile uint8_t comFlag = 0;

//MUSTER: 16 Einzelschritte, Geschwindigkeit_9 entspricht 0,25 Sekunden,
//Geschwindigkeit_0 entspricht 2,5 Sekunden pro Durchlauf

//Musterangabe für Geschwindigkeit_9, kann durch multiplizieren
//auf andere Geschwindigkeiten hochgerechnet werden
```

Anlagen, Teil 5

```
volatile uint64_t musterzaehler = 0;
volatile static uint8_t musterarray[6][16] =
 {{1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0},
  {1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
  {1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0},
  {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
  {1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0},
  {1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0}
};

//Funktionsruempfe
void GPIO_init();
void timer_init();
void ressetArray();
void setzeAktorpins();
void setzeArraydaten(uint8_t wert, uint8_t einstellung);
uint8_t checkFor(uint8_t einstellung, uint16_t data);
bool checkForByte();
bool Bytesempfangen();
void empfangeBefehlskette();

void setup() {
  // start serial port at 9600 bps:
  Serial.begin(115200);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
  setzeAktorpins();
  timer_init();
}

void loop() {
  if (Bytesempfangen()) {
    empfangeBefehlskette();
  }
  if (timerFlag) {
    timerFlag = 0;
    setzePWMPins();
  }
}
```

```
void empfangeBefehlskette() {
  if (checkForByte()) {
    uint8_t einstellung = Serial.read();
    if (checkForByte()) {
      uint8_t serialBuffer = Serial.read();
      switch (einstellung) {
        case Einstellung_Muster:              //setze Muster
          uint8_t muster;
          if (muster = checkFor(Einstellung_Muster, serialBuffer)) {
            setzeArraydaten(muster , Einstellung_Muster);
          }
          break;
        case Einstellung_Geschwindigkeit:     //setze Geschwindigkeit
          uint8_t geschwindigkeit;
          if (geschwindigkeit = checkFor(Einstellung_Geschwindigkeit, serialBuffer)) {
            setzeArraydaten(geschwindigkeit, Einstellung_Geschwindigkeit);
          }
          break;
        case Einstellung_PWM:                 //setze PWM
          if (serialBuffer <= 255) {
            setzeArraydaten(serialBuffer, Einstellung_PWM);
          }
          break;
        case Einstellung_Dauermodus:          //setze Dauermodus
          if (serialBuffer == 1 || !serialBuffer) {
            setzeArraydaten(serialBuffer, Einstellung_Dauermodus);
          }
          break;
        case Einstellung_Aktiviert:           //setze Aktiviert
          if (serialBuffer == 1 || !serialBuffer) {
            setzeArraydaten(serialBuffer, Einstellung_Aktiviert);
          }
          break;
        case Aktorauswahl:                   //Aktorauswahl
          if (checkForByte()) {
            uint16_t aktorbuffer = (serialBuffer << 8) + Serial.read();
            if (checkFor(Aktorauswahl, aktorbuffer)) {
              aktor = aktorbuffer;
            }
          }
          break;
        default:
          break;
      }
    }
  }
  comFlag = 1; //setze Timerzaehler zurück
  return;
}
```

Anlagen, Teil 5

```
ISR(TIMER5_COMPA_vect) {   //This is the interrupt request
  //digitalWrite(7, !digitalRead(7));

  if (comFlag) {
    musterzaehler = 0;
    comFlag = 0;
  } else {
    musterzaehler ++;
  }
  timerFlag = 1;
}

void GPIO_init() {
  //PWM Pins auf Output setzen und low
  for (int i = 0; i < Anzahl_Aktoren; i++) {
    pinMode(aktorarray[i][Einstellung_PIN], OUTPUT);
    analogWrite(aktorarray[i][Einstellung_PIN], 0);
  }
  return;
}

void timer_init() {
  //PWM Timer5
  TCCR5A = 0;
  TCCR5B |= (1 << WGM52); //Setze Modus "Clear Timer on Compare Match"
  TCCR5B |= (1 << CS51); //Setze Prescaler 1/64 clock
  TCCR5B |= (1 << CS50);//Setze Prescaler 1/64 clock
  //0F41 Wert für das "Compare-Register" for 15,625 ms,
  //Formel im Manual ATMega2560 Seite 145 FF.
  uint16_t cmpmatchvalue = 0x0F41;
  //Setze den Wert in das Register HIGH BYTE und LOW BYTE
  OCR5AH = (cmpmatchvalue >> 8);
  OCR5AL = (cmpmatchvalue && 0xFF);
  //Aktiviere Interrupt bei erreichen des Vergleichswertes
  TIMSK5 |= (1 << OCIE5A);
  //Aktiviere die Interrupts
  sei();
  return;
}
```

```
void setzeAktorpins() {
  ressetArray();

  aktorarray[0][Einstellung_PIN] = 2;
  aktorarray[1][Einstellung_PIN] = 3;
  aktorarray[2][Einstellung_PIN] = 5;
  aktorarray[3][Einstellung_PIN] = 6;
  aktorarray[4][Einstellung_PIN] = 7;
  aktorarray[5][Einstellung_PIN] = 8;
  aktorarray[6][Einstellung_PIN] = 9;
  aktorarray[7][Einstellung_PIN] = 10;
  aktorarray[8][Einstellung_PIN] = 11;
  aktorarray[9][Einstellung_PIN] = 12;

  GPIO_init();
  return;
}

void setzePWMPins() {
  for (int i = 0; i < Anzahl_Aktoren; i++) {
    if (aktorarray[i][Einstellung_Aktiviert]) {

      uint8_t musterstelle = musterzaehler / aktorarray[i][Einstellung_Geschwindigkeit]
      % 16;//berechne Musterstelle
      analogWrite(aktorarray[i][Einstellung_PIN], aktorarray[i][Einstellung_PWM] *
      musterarray[aktorarray[i][Einstellung_Muster]][musterstelle]);
      //setzte deaktiviert nach einem Durchlauf,falls Dauermodus deaktiviert
      if (!aktorarray[i][Einstellung_Dauermodus] && musterstelle == 15 &&
      (musterzaehler + 1) / aktorarray[i][Einstellung_Geschwindigkeit] % 16 == 0) {
        aktorarray[i][Einstellung_Aktiviert] = 0;
      }
    } else {
      analogWrite(aktorarray[i][Einstellung_PIN], 0);
    }
  }
  return;
}

bool Bytesempfangen() {
  if (Serial.available() >= 2)
  {
    uint8_t byte1 = Serial.read();
    uint8_t byte2 = Serial.read();
    if (byte1 == 0xF5 && byte2 == 0x53) {
      return true;
    }
  }
  return false;
}
```

Anlagen, Teil 5

```
bool checkForByte() {
  myTimer = millis();
  while (millis() <= myTimeout + myTimer) {
    if (Serial.available()) {
      return true;
    }
  }
  return false;
}

uint8_t checkFor(uint8_t einstellung, uint16_t data) {
  switch (einstellung) {
    case Einstellung_Muster:
      if (data == Muster_0 || data == Muster_1 || data == Muster_2 || data == Muster_3 ||
      data == Muster_4 || data == Muster_5) {
        return data - 0x40; //Muster 0 bis 5 zurückgeben
      }
      break;
    case Einstellung_Geschwindigkeit:
      if (data == Geschwindikeit_0 || data == Geschwindikeit_1 || data == Geschwindikeit_2 ||
          data == Geschwindikeit_3 || data == Geschwindikeit_4 || data == Geschwindikeit_5 ||
          data == Geschwindikeit_6 || data == Geschwindikeit_7 || data == Geschwindikeit_8 ||
          data == Geschwindikeit_9) {
        return 10 - (data - 0x30);
        //Geschwindigkeit 1 bis 10 zurückgeben und invertieren,
        //damit Geschwindigkeit 1 am langsamsten ist
      }
      break;
    case Aktorauswahl:
      if (data <= (1 << Anzahl_Aktoren) - 1) {
        return 1;
      }
    default:
      break;
  }
  return 0;
}

void setzeArraydaten(uint8_t wert, uint8_t einstellung) {
  for (int i = 0; i < Anzahl_Aktoren; i++) {
    if ((1 << i)& aktor) {
      aktorarray[i][einstellung] = wert;
    }
  }
}
```

```
void ressetArray() {
  for (int i = 0; i < Anzahl_Aktoren; i++) {
    for (int j = 0; j < Anzahl_Einstellungen; j++) {
      if (j == Einstellung_Geschwindigkeit) {
```

Anlagen, Teil 6

Software des GUI-Testsystems

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using System.Windows.Forms;
using System.IO.Ports;


namespace _3dimHapt_Aktorsystem
{
    15 Verweise
    static class Program
    {

        public static System.IO.Ports.SerialPort ArduinoCOM = new System.IO.Ports.SerialPort();
        public static ushort sensorauswahl = 0;


        /// <summary>
        /// Der Haupteinstiegspunkt für die Anwendung.
        /// </summary>
        [STAThread]
        0 Verweise
        static void Main()
        {
            ArduinoCOM.PortName = "COM7";
            ArduinoCOM.BaudRate = 115200;
            ArduinoCOM.Open();
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
            ArduinoCOM.Close();
        }

        5 Verweise
        static void setzeSensor(ushort sensor)
        {
            byte[] senddata = {0xF5, 0x53, 0x06, Convert.ToByte(sensor>>8), Convert.ToByte(sensor & 0xFF) };
            ArduinoCOM.Write(senddata, 0, senddata.Length);
        }

        1-Verweis
        public static void setzeMuster(byte muster, ushort sensor)
        {
            setzeSensor(sensor);
            byte[] senddata = { 0xF5, 0x53, 0x00, Convert.ToByte(muster +0x40) };
            ArduinoCOM.Write(senddata, 0, senddata.Length);
        }

        1-Verweis
        public static void setzeGeschwindigkeit(byte speed, ushort sensor)
        {
            setzeSensor(sensor);
            byte[] senddata = { 0xF5, 0x53, 0x01, Convert.ToByte(speed + 0x30) };
            ArduinoCOM.Write(senddata, 0, senddata.Length);
        }
```

Anlagen, Teil 6

```
        1-Verweis
        public static void setzePWM(byte intensity, ushort sensor)
        {
            setzeSensor(sensor);
            byte[] senddata = { 0xF5, 0x53, 0x02, intensity };
            ArduinoCOM.Write(senddata, 0, senddata.Length);
        }

        1-Verweis
        public static void setzeDauermodus(byte modus, ushort sensor)
        {
            setzeSensor(sensor);
            byte[] senddata = { 0xF5, 0x53, 0x03, modus};
            ArduinoCOM.Write(senddata, 0, senddata.Length);
        }

        3 Verweise
        public static void setzeAktiv(byte aktiv, ushort sensor)
        {
            setzeSensor(sensor);
            byte[] senddata = { 0xF5, 0x53, 0x04, aktiv };
            ArduinoCOM.Write(senddata, 0, senddata.Length);
        }
    }

}
namespace _3dimHapt_Aktorsystem
{
    3 Verweise
    partial class Form1
    {
        /// <summary>
        /// Erforderliche Designervariable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Verwendete Ressourcen bereinigen.
        /// </summary>
        /// <param name="disposing">True, wenn verwaltete Ressourcen
        /// gelöscht werden sollen; andernfalls False.</param>
        0 Verweise
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        Vom Windows Form-Designer generierter Code

        private System.Windows.Forms.CheckedListBox checkedListBox1;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.ListBox listBox1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.TrackBar trackBar1;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.TrackBar trackBar2;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.CheckBox checkBox1;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button button3;
    }
}
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace _3dimHapt_Aktorsystem
{
    3 Verweise
    public partial class Form1 : Form
    {
        1-Verweis
        public Form1()
        {
            InitializeComponent();
        }

        1-Verweis
        private void checkedListBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            Program.sensorauswahl = 0;
            for(int i = 0; i<checkedListBox1.CheckedIndices.Count; i++)
            {
                Program.sensorauswahl |= Convert.ToUInt16(1 << checkedListBox1.CheckedIndices[i]);
            }
        }

        1-Verweis
        private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            Program.setzeMuster(Convert.ToByte(listBox1.SelectedIndex), Program.sensorauswahl);
        }

        1-Verweis
        private void checkBox1_CheckedChanged(object sender, EventArgs e)
        {
            Program.setzeDauermodus(Convert.ToByte(checkBox1.Checked), Program.sensorauswahl);
        }

        1-Verweis
        private void trackBar1_Scroll(object sender, EventArgs e)
        {
            Program.setzeGeschwindigkeit(Convert.ToByte(trackBar1.Value), Program.sensorauswahl);
        }

        1-Verweis
        private void trackBar2_Scroll(object sender, EventArgs e)
        {
            Program.setzePWM(Convert.ToByte(trackBar2.Value), Program.sensorauswahl);
        }

        1-Verweis
        private void button1_Click(object sender, EventArgs e)
        {
            Program.setzeAktiv(1, Program.sensorauswahl);
        }

        1-Verweis
        private void button2_Click(object sender, EventArgs e)
        {
            Program.setzeAktiv(0, Program.sensorauswahl);
        }

        1-Verweis
        private void button3_Click(object sender, EventArgs e)
        {
            Program.setzeAktiv(0, 1023);
        }
    }
}
```

Anlagen, Teil 6

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Alle Abbildungen ohne Quellenangabe im selben Kapitel sind selbst erstellt worden.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Mittweida, den 16.08.2021

Kevin Blümel