|  |  |  |
| --- | --- | --- |
|  | |  |
| Konzeption und Realisierung eines Personenzählsystems mit Hilfe eines Radarsensors | | |
|  | | |
|  | | |
| des Studiengangs  an der Dualen Hochschule Baden-Württemberg Stuttgart | | |
| von | | |
| Abgabedatum: 10.06.2022 | | |
| Bearbeitungszeitraum | 17.01.2022 - 10.06.2022 | |
| Matrikelnummer, Kurs | 7081869, 3873765, TEL19GR5 | |
| Ausbildungsfirma | Infineon Technologies AG, Neubiberg | |
| Betreuer der Dualen Hochschule | Dr., Patrick Köberle | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Erklärung | | | | | |
| Ich versichere hiermit, dass ich meine mit dem Thema Konzeption und Realisierung eines Personenzählsystems mit Hilfe eines Radarsensors selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.  Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt. | | | | | |
|  |  |  |  |  |  |
| Ort |  | Datum |  | Unterschrift |  |

Abstract

Inhalt

[Abbildungsverzeichnis V](#_Toc105335797)

[Tabellen V](#_Toc105335798)

[Abkürzungen VI](#_Toc105335799)

[Formelgrößen und Einheiten VI](#_Toc105335800)

[4.1.1 dritte Ebene 4](#_Toc105335801)

[4.2.1 dritte Ebene 4](#_Toc105335802)

[Literaturverzeichnis 6](#_Toc105335803)

[Anhang 7](#_Toc105335804)

Abbildungsverzeichnis

**No table of figures entries found.**

Tabellen

**No table of figures entries found.**

Abkürzungen

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Formelgrößen und Einheiten

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

# Einleitung

# Aufgabenstellung

# Stand der Technik

# xxxx

## Unterüberschrift

Text

### dritte Ebene

Text

#### vierte Ebene

Text

## Unterüberschrift

Text

### dritte Ebene

Text

## Programmierung des Zählalgorithmus

Nachdem die Rohdaten verarbeitet, gefiltert und gemittelt worden sind, müssen nun die erkannten Targets aus der Klasse FTarget verfolgt und gezählt werden. Im Folgenden werden dazu das Konzept und die Realisierung des Trackings und der Zählung beschrieben.

### Konzept und Realisierung der Personenverfolgung

Die Personenverfolgung ist der wahrscheinlich komplizierteste Teil des Programms. Hier könnte es sich anbieten, klassische fertige Filter zur Personenverfolgung zu verwenden, die in der Radartechnik schon bewährt sind wie ein Alpha-Beta-Tracker oder ein Kalman Filter. Mit dem vorliegenden Radar-Sensor hat sich die Geschwindigkeits-Messung mithilfe des Doppler-Effekts als nicht sehr präzise und verlässlich in Versuchen ergeben. Aus diesem Grund wurden die zwei genannten Filter nicht verwendet, welche sonst eine gute Lösung gewesen wären. Ein anderer Vorteil einer eigenen Lösung ist, dass immer neue Personen in verschiedensten Positionen auftauchen können. Diese müssen anhand des eigenen Programms klassifiziert werden können.

In Figure 1 ist der Programmablaufplan des Personenverfolgung dargestellt, welcher im Folgenden weiter erläutert wird.

Die neuen aufbereiteten Daten werden mit jedem Durchlauf des Programms in den Vektor Targets der Klasse FTarget geschrieben, d.h. jedes Element des Vektors ist eine Person, welche duch die Klasse FTarget beschrieben wird. Dabei ist erst zur aktuellen Laufzeit bekannt, wie groß dieser Vektor ist, da nicht bekannt ist, wie viele Personen erkannt werden. Daher muss dieser Vektor bei jedem Durchlauf erst komplett gelöscht werden, bevor neue Daten hineingeschrieben werden. In der Variable position wird dazu dann die Anzahl an erkannten Personen in diesem Durchlauf gespeichert. Die For-Schleifen in diesem Abschnitt benutzen dann meist diese Variable als Endkriterium.

C:\Users\RostJosh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\AblaufplanTracking.emf

Figure 1: Programmablaufplan der Personenverfolgung

Im ersten Durchlauf wird Targets an den Vektor Targets\_aktuell, welcher auch aus Elementen der Klasse FTarget besteht, übergeben. Diese Variable wird dann gespeichert und nicht in jedem Durchlauf zurückgesetzt, um Personen zu verfolgen, die öfters erkannt werden.

Targets\_aktuell beschreibt im aktuellen Durchlauf den Abstand, Winkel und Geschwindigkeit einer Person und wird dann im Schritt der Zählung ein Herkunftsort gegeben, um schon geortete Personen zu identifizieren. In jedem weiteren Durchlauf nach dem ersten werden nun die Beträge der Differenz des Winkels und des Abstands zwischen jedem Objekt in dem Vektor Targets (nun neuere Werte) und jedem Objekt des Vektors Targets\_aktuell (hier sind theoretisch jetzt die Werte des vorherigen Durchlaufs) ermittelt. Diese Differenzen werden dann in den Differenzmatrizen Difference und Angle\_Difference gespeichert. Diese Matrizen werden dabei in jedem Durchlauf mit NaN in der Größe position initialisiert.

Dann werden diese Differenzmatrizen gefiltert. Es werden dann nur Werte, die nicht NaN und unter dem jeweiligen max. Veränderungsschwellen sind, durchgelassen. Die Veränderungsschwelle für den Abstand ist in der Variable DetectionDistance gespeichert und hat den Wert 1. Für den Winkel ist die Schwelle in AngleDetectionDistance gespeichert und hat den Wert 10. Unter der Annahme, dass der Sensor verlässliche Werte in kurzen Abständen zurückgibt, sorgen diese Schwellen dafür, dass sich Personen nur mit kleinen Differenzen fortbewegen können. Die Werte bedeuten, dass ein erkanntes Target im nächsten Durchlauf nur max. 1m und 10° bewegen kann, sonst wird es als neues Target erfasst und das alte wird gelöscht. Die gefilterten Werte werden in den Differenzmatrizen min\_Difference\_Location und min\_Difference\_Angle derselben Größe gespeichert. Auch diese Matrizen werden dabei in jedem Durchlauf mit NaN initialisiert.

Für den Fall, dass nun mehrere Personen sich in unmittelbarer Nähe aufhalten und als unterschiedliche Personen erkannt werden, müssen diese auch durchgehend erkannt werden. Aus diesem Grund werden aus der gefilterten Differenzmatrix min\_Difference\_Location reihenweise die kleinsten Werte gesucht. Diese kleinsten Werte werden dann in der Variable minValue gespeichert, welche jeden Durchlauf mit -1 initialisiert wird. Da nur Beträge in den Matrizen stehen können, ist -1 ein nicht gültiger Wert und kann nicht in den Matrizen stehen. Dies ist somit eine zusätzliche Absicherung gegen Fehler. minValue wird allerdings nur aus min\_Difference\_Location gesucht, da die Winkelmessung nicht sehr zuverlässig gewesen ist. Als Verbesserungspunkt für diesen Teil könnte man noch die kleinsten Winkelunterschiede suchen und diese mit den kleinsten Werten aus minValue korrelieren.

Nachdem man die kleinsten Differenzen gefunden hat, werden die Spalten gesucht, in denen die minimalen Werte gespeichert sind. Die Spalten werden dann in minColumn gespeichert. In der Figure 2 und der Figure 3 wird anhand eines Beispiels gezeigt, wie die bisher genannten Variablen in diesem Abschnitt mit echten Werten aussehen. In Targets\_aktuell sind 2 alte Targets gespeichert und in Targets 3 neue mit jeweils unterschiedlichen Distanz, Winkel und Geschwindigkeitswerten. Dem ersten Target in Targets\_aktuell ist auch ein Herkunftsort zugeordnet, dazu später mehr.

In Figure 3 ist nun auch die Differenzmatrix dargestellt. Jede Zeile sind dabei die alten Distanzwerte aus Targets\_aktuell und in jeder Spalte werden davon die Distanzwerte aus Targets abgezogen. Nach der Filterung, die dann in min\_Difference\_Location zu sehen ist, werden die kleinsten Werte gesucht und dann die Spalten dieser in minColumn angezeigt. Somit kann man nun sehen, dass Targets(1) Targets\_aktuell(1) zugeordnet werden soll und Targets(3) Targets\_aktuell(3). Dabei muss natürlich beachtet werden, dass keine Ziffer doppelt in minColumn vorkommt. Falls doch muss diesem Element von Targets\_aktuell das zweitnächste Objekt der Vektors Targets zugeordnet werden. Falls dies nicht möglich ist, soll das alte Element von Targets\_aktuell gelöscht werden und das neue Element von Targets einfach zugewiesen werden. Falls in der Spalte keine Ziffer doppelt vorkommt, können existierende Elemente von Targets\_aktuell ihre neuen Werte aus Targets bekommen, indem sie die Ziffern aus minColumn zum dereferenzieren verwenden.

Im Falle, dass das Elemente von Targets\_aktuell nicht existiert, muss es neu angelegt werden und dann mit den neuen Daten aus Targets zugewiesen werden. Zum Schluss werden noch diejenigen Targets\_aktuell gelöscht, welchen keine neuen Daten zugewiesen bekommen haben. Dies kann der Fall sein, wenn z.B. im vorherigen Durchlauf 3 Personen erkannt wurden und im aktuellen Durchlauf nur 2. Wenn diese 2 Personen korrekt zugewiesen worden sind, muss die 3. Person gelöscht werden, da sie nicht mehr erkannt wird.

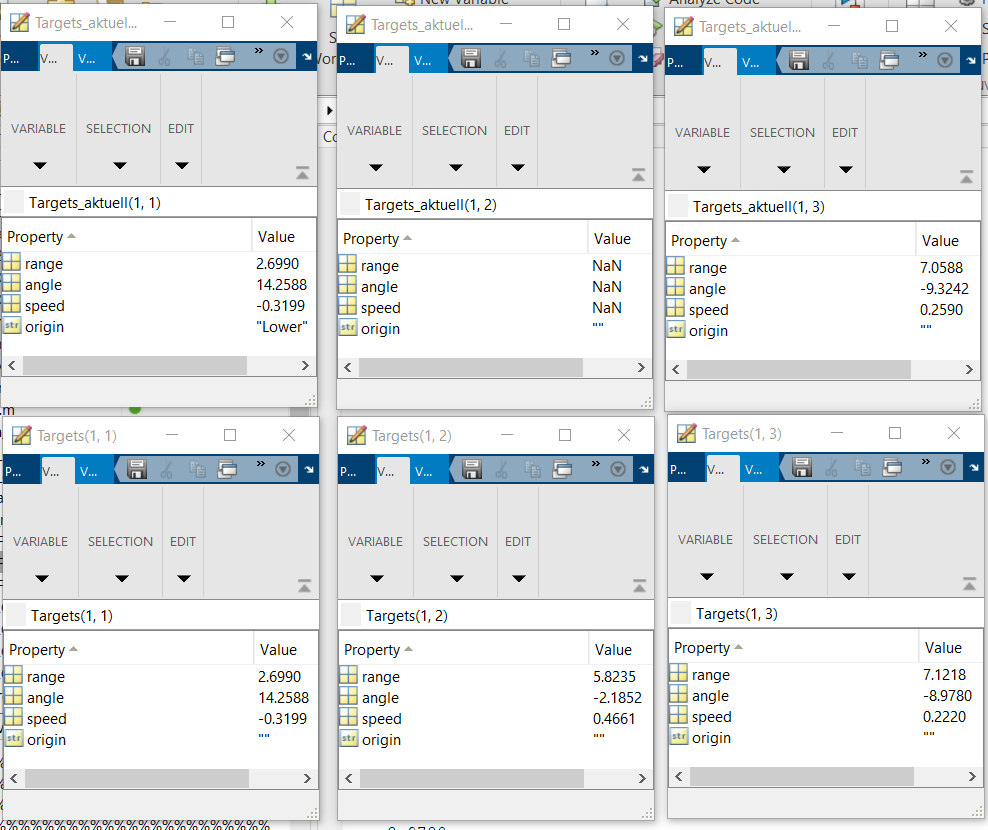


Figure 2: Darstellung eines Zeitpunkts mit 2 alten Targets und 3 neuen Targets

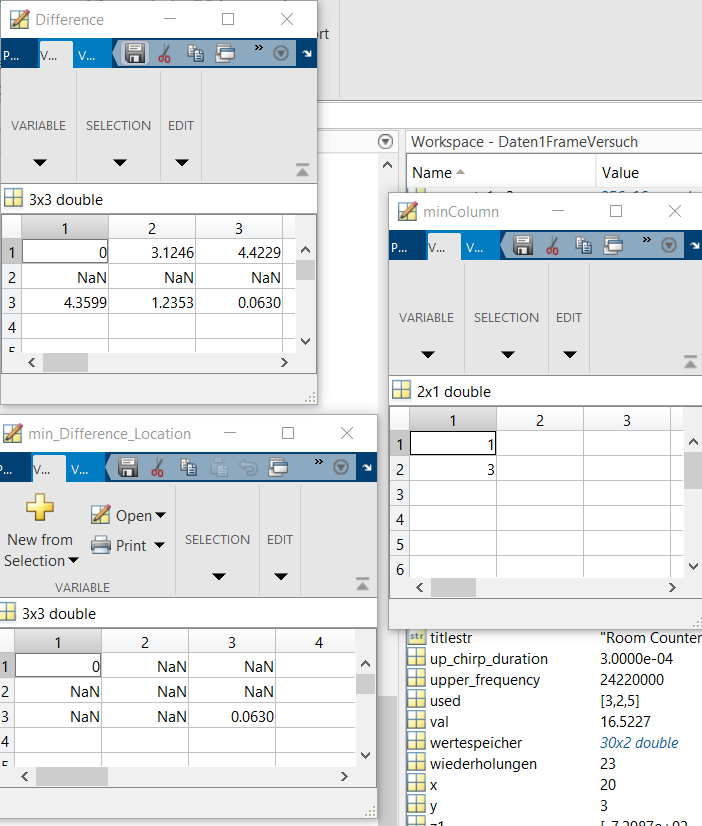


Figure 3: Zugehörige Darstellung der Differenzmatrix, gefilterten Differenzmatrix und Gefundenen Spalten

### Konzept und Realisierung der Personenzählung

Nachdem die Daten nun immer in jeden Durchlauf in Targets\_aktuell gespeichert werden, soll anhand den Daten dieser Variable die Personenzählung funktionieren. In Figure 1 ist das Konzept des Versuchsaufbau dargestellt, welches das Konzept der Personenzählung direkt beeinflusst. Da sich die Präzision und Zuverlässigkeit des Abstandes in Versuchen als hoch erwiesen hat und die Zuverlässigkeit der Winkelmessung geringer, wurde dieser ungewöhnliche Versuchsaufbau gewählt, um Personen über einen längeren Zeitpunkt zu verfolgen und zu zählen.

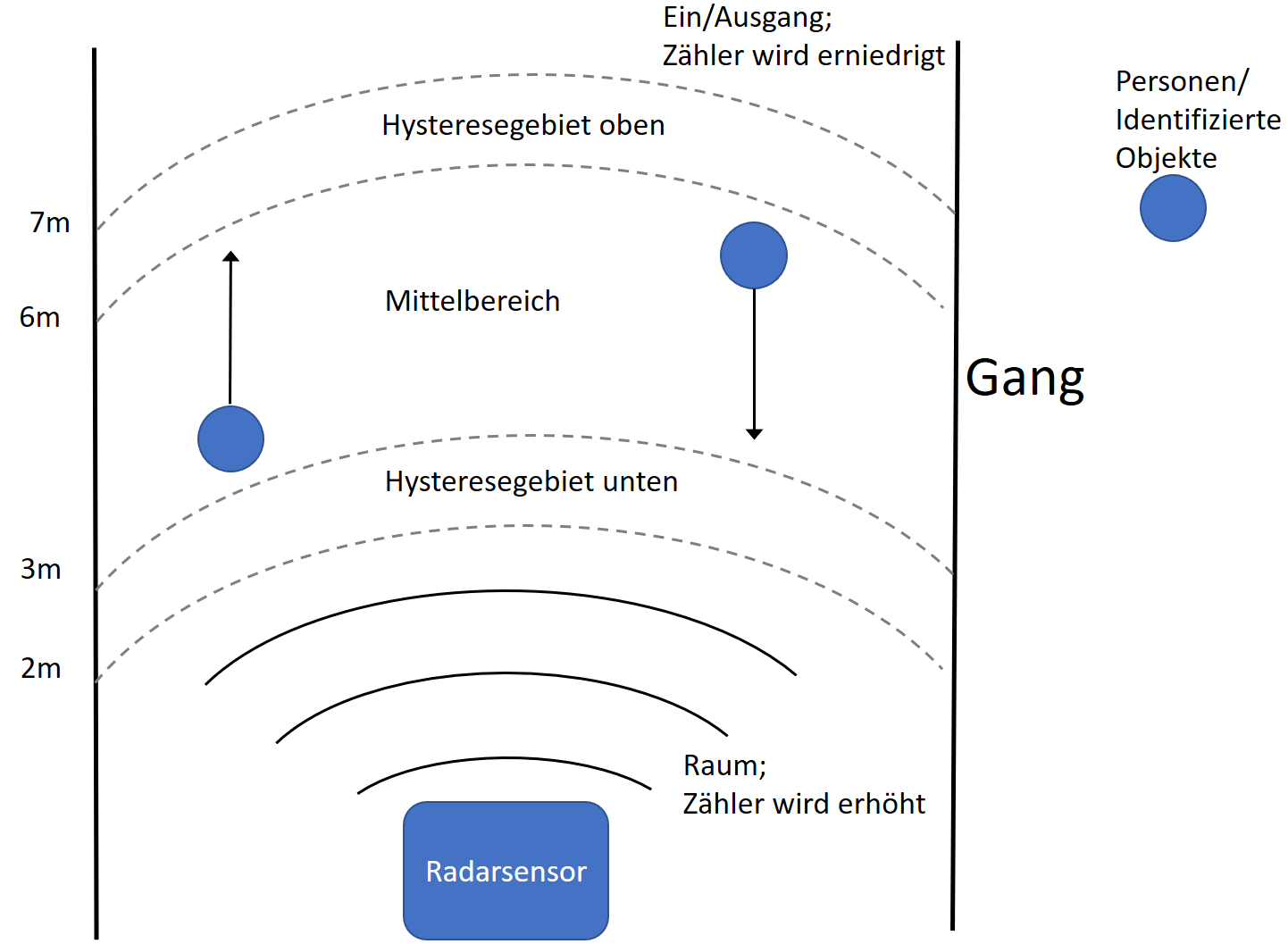


Figure 4: Konzept des Aufbaus

Der Bereich über 7 m wird als Ein- und Ausgang deklariert. Wenn eine Person erkannt wird, die sich vorher in dem nahen Hysterese-Bereich vom Radar aufgehalten und über 7m hinaus noch erkannt wird, wird der Raum Zähler erniedrigt. Dazu wird jedem erkannten Objekt im nahen Hysterese Bereich beim Attribut origin „Lower“ gegeben. Dieses Attribut wird dann auch abgefragt, wenn ein Objekt über der 7m Grenze erkennbar ist, um den Zähler zu erniedrigen.Der Zähler, der die Anzahl der Personen im Raum beschreibt, wird in der Variable Room\_Counter gespeichert.

Durchläuft eine Person hingegen zuerst den Hysterebereich oben und ist dann unterhalb der unteren Hyteresegrenze von 2 m des unteren Hysteresegebiets erkennbar (also im Raum Bereich), dann wird dies so bewertet, dass eine Person den Raum betritt und der Raum Zähler wird erhöht. Dabei wird dem Objekt das Attribut origin „Upper“ gegeben, wenn es das erste Mal im oberen Hysterebereich erkannt wird.

Aus diesem Grund ist auch wichtig, dass Objekte im nächsten Durchlauf weiterhin mit einer geringen Änderung erkannt werden, da das Attribut origin immer weitergegeben wird, um die Herkunft alter erkannter Personen zu wissen.

Wenn eine Person sich nur im Ein/Ausgangsbereich oder im Raum Bereich befindet und dort erkannt wird, wird diese Person nicht weiter verarbeitet. Falls eine Person im Mittelbereich erkannt wird, die davor nicht in einer der beiden Hysteregebiete aufgehalten hat, wird dies als Fehler bewertet und diese Person wird gelöscht.

Die Grenzen, die bestimmen ab welcher Distanz der Eingangsbereich definiert ist, sind in den Variablen Entrance\_HighRangeLimit und Entrance\_LowRangeLimit und haben aktuell die Werte 7 und 6, welche nach Tests natürlich noch geändert werden können. Die korrespondierenden Variablen für den Raumbereich sind Room\_HighRangeLimit und Room\_LowRangeLimit, welche die Werte 2 und 3 haben.

Die Darstellung der Targets\_aktuell erfolgt in Polarplots für jedes aktuelle Element, welche Abstand und Winkel in einen Graph kombinieren. In diesen Graphen stehen auch im Titel der aktuelle Raum Zähler (in der Variable Room\_Counter) und als Notiz die Geschwindigkeit sowie der erkannte Herkunftsort dieser Person. In der Praxis sieht das dann wie folgt aus (siehe Figure 5, Figure 6 und Figure 7). Hierbei ist auch gut zu sehen bei Figure 6, dass kein Herkunftsort zugewiesen werden konnte, da das Objekt als erstes bei etwa 5 m identifiziert worden ist, hingegen bei Figure 5 und Figure 7 konnte einer zugewiesen werden. Die Ziffer in den Titel der Abbildungen beschreibt dabei, welches Target dabei immer gemeint ist.

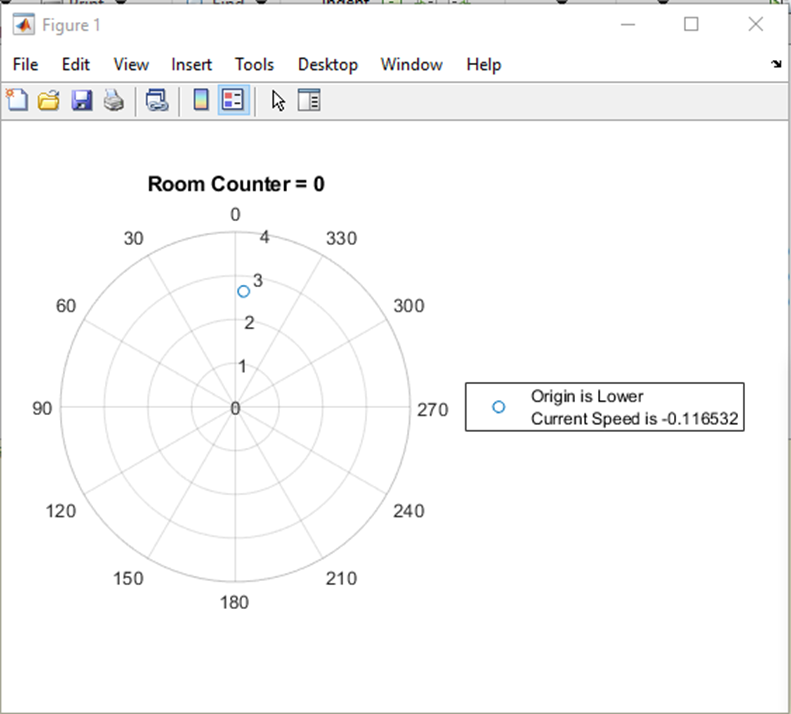


Figure 5: Graph von Objekt 1

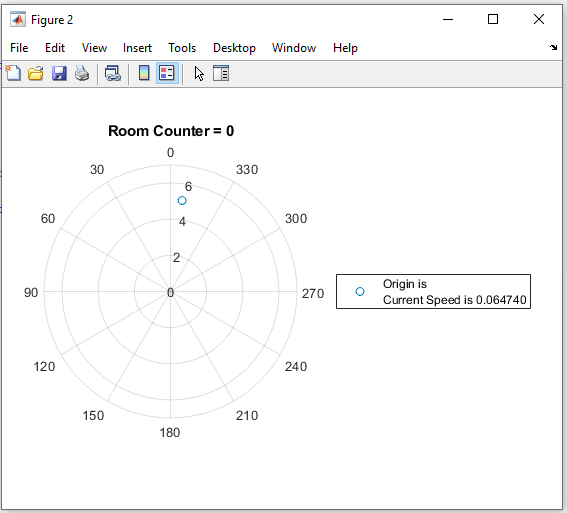


Figure 6: Graph von Objekt 2

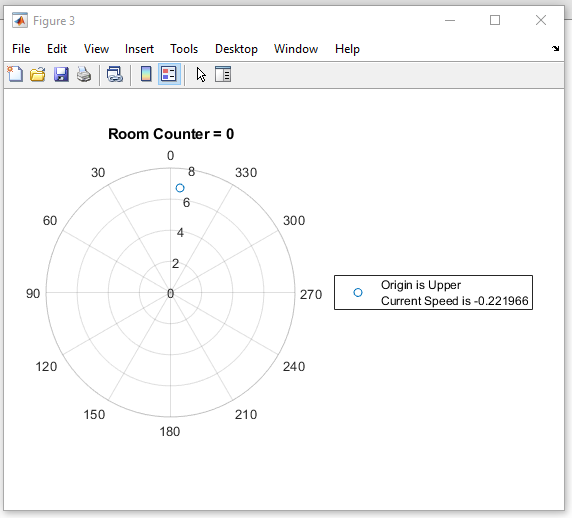


Figure 7: Graph von Objekt 3

## Modultest

Dieser Teil beschreibt einen einfachen Test, der sowohl das Tracking als auch die Zählung umfasst. Dazu wird ähnlich wie in der Beschreibung von High Level Sprachen wie VHDL ein Test Case entwickelt, der mit sehr einfachen Daten und Schritten arbeitet. Dabei werden in diesem Test 2 Personen simuliert, die durch das Messgebiet laufen. Person 1 startet bei 6,5 m, hat einen Winkel von 9 ° und. läuft dabei Richtung Raum. Er bewegt sich mit -0.5m pro Durchlauf und ändert seinen Winkel um 1 ° in jedem Durchlauf. Person 2 startet bei 2,5 m sowie einem Winkel von 19° und läuft mit jedem Messschritt +0.5m Richtung Ausgang und ändert seinen Winkel um -1 °. Wenn Person 2 6 m erreicht hat, bleibt er an dieser Position stehen. Obwohl beide Personen in diesem Fall theoretisch aneinander vorbeilaufen, können diese von einander unterschieden werden aufgrund des unterschiedlichen Winkels. Die angezeigte Geschwindigkeit ist willkürlich gewählt und soll im Folgenden ignoriert werden. In Figure 8 wird das Ausgabefeld nach dem ersten Durchlauf angezeigt. Die korrespondierenden Figure 8 zeigt die Graphen zur Startposition. Im ersten Durchlauf können noch keine Herkunftsorte angezeigt werden.

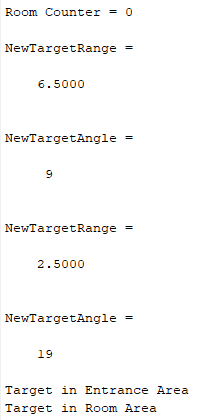


Figure 8: Ausgabefeld nach dem ersten Durchlauf

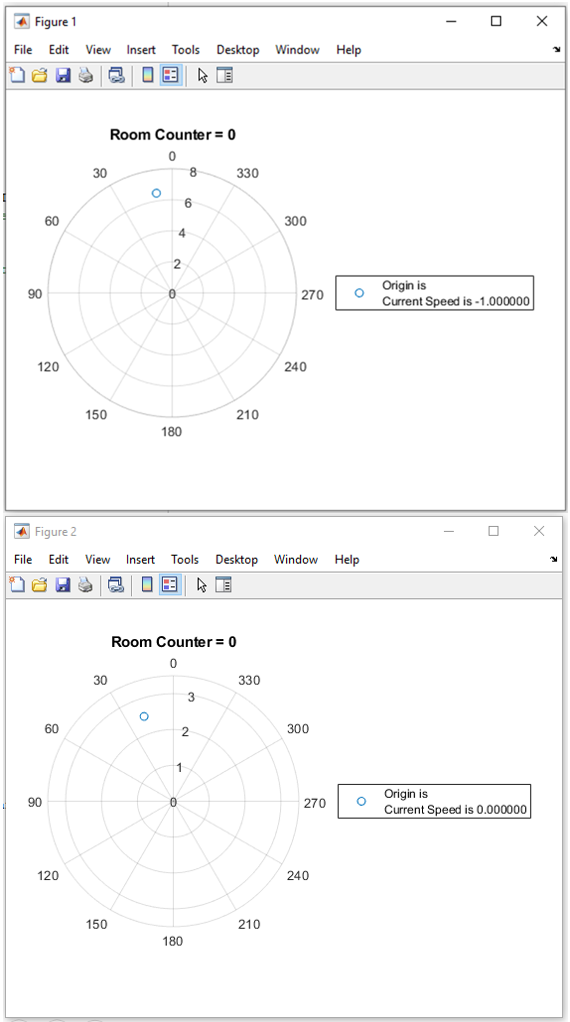


Figure 9: Graphen zur Start Position

In Figure 10 ist nun eine kritische Position dargestellt: Dieselben Objekte 1 und 2 haben nun die selbe Distanz von 4,5 m und einen leicht unterschiedlichen Winkel von jeweils 5 ° und 15 °.

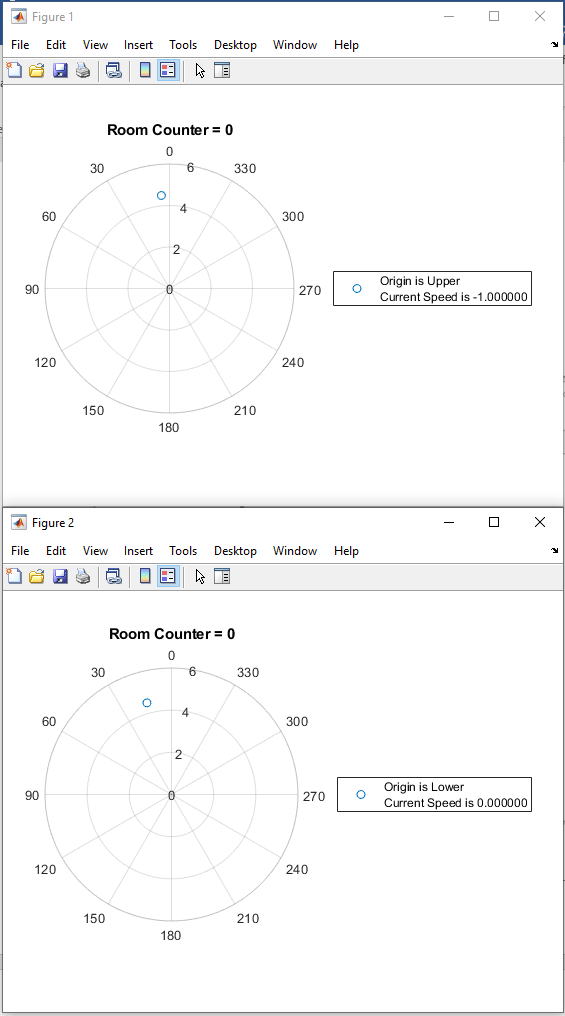


Figure 10: Kritische Position, beide Position im selben Abstand und ähnlichen Winkel

Figure 11 zeigt den Status im nächsten Durchlauf. Wie zu sehen ist, wurde die kritsche Position korrekt aufgelöst und die Ziele haben immer noch ihre korrekten Ursprungsorte.

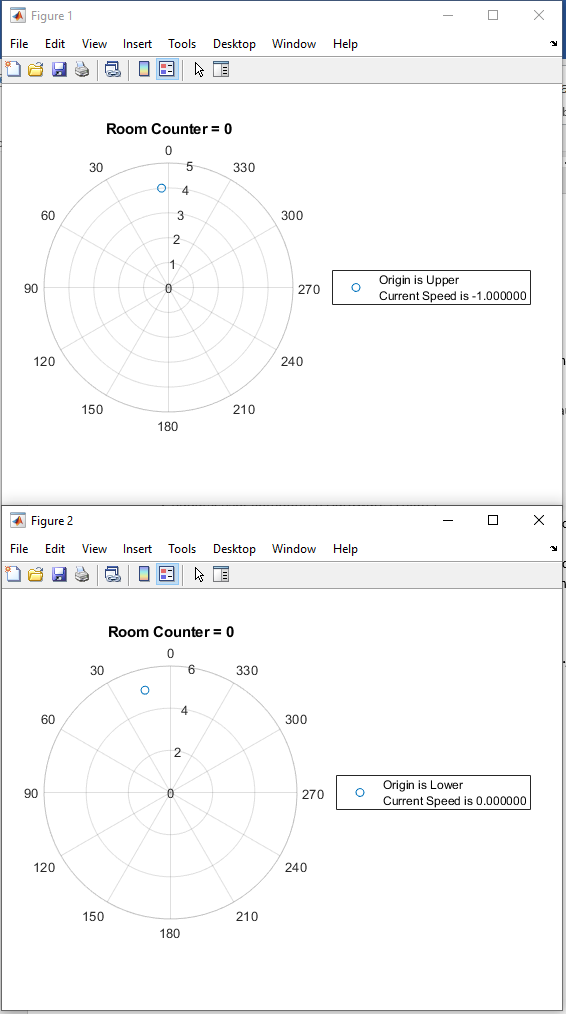


Figure 11: Aufgelöste Kritische Position

In Figure 12 wird nun die Endposition des Modultests präsentiert. Die Person 1 ist nun in den Raumbereich eingetreten und aus diesem Grund hat sich der Raum Zähler erhöht. Einmal dort erkannt, wird er aber direkt gelöscht, sodass er keinen Herkunftsort im letzten Durchlauf besitzt.

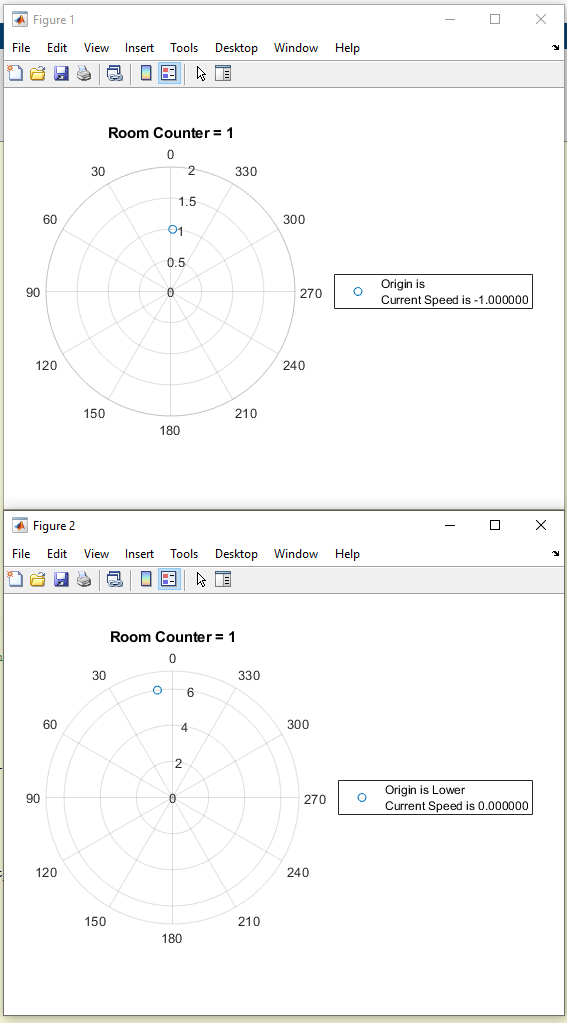


Figure 12: Endposition, Person 1 ist in den Raum eingetreten

## Problemfälle, Schwierigkeiten und Lösungen

Im vorherigen Abschnitt wurde mit dem Modultest gezeigt, dass die Grundfunktionalität des Trackings und Personenzählens gegeben ist. Jedoch war dieser Test mit sehr geeigneten und vereinfachten Positions- und Winkeldaten durchgeführt worden. In diesem Abschnitt werden nun mögliche Problemfälle in der Realität, Schwierigkeiten und mögliche Lösungen erörtert, die aus Komplexitätsgründen nicht durchgeführt worden sind.

Einer der möglichen Problemfälle könnte so aussehen: 2 Personen laufen geradeaus entgegengesetzt, Person 2 erscheint auf der Position von Person 1 im vorherigen Durchlauf und Person 1 erscheint auf der Position von Person 2 des vorherigen Durchlaufs. Es wird jedoch aufgrund der letzten Position angenommen, dass Person 1 und 2 stehengeblieben sind und dann sich umdrehen. Dies ist natürlich irreführend und kann mit dem aktuellen Programm nicht detektiert werden.

Eine andere Schwierigkeit, die bei Tests in der Realität aufgetreten ist, sind verschiedene Anzahlen von erkannten Objekten zu unterschiedlichen Zeitpunkten. Das könnte z.B. sein, dass im vorherigen Durchlauf nur 1 Person erkannt wurde und im nachfolgenden Durchlauf dann 3 oder andersherum. Dies führt dann zu Problemen bei der korrekten Zuordnung von neuen zu alten Daten und welche Daten dann gelöscht werden sollten.

Zudem kann es trotz Durchschnittsbildung des Abstandes und Winkels noch zu vereinzelten Aussetzern und Artefakten kommen. Bei eine einzelnen Aussetzer, also wenn eine Person in einem Durchlauf nicht erkannt wird, wird der Herkunftsort nicht mehr übertragen und damit gelöscht. Es kann nicht mehr nachollzogen werden, dass dieselbe Person schon länger im System gewesen ist und damit wird die Gesamtfunktionalität umgangen. Die Störsicherheit ist außerdem nur im Freien oder in sehr großen Räumen ohne weitere Objekte gegeben.

Es gibt verschiedene Lösungen zu genannten Problemen: Durch ein feste Installierung von weiterem Empfangs- und Sendeantennen könnte zumindest das letzt genannte Problem umgangen werden, obwohl man damit die Flexibilität des aktuellen Systems verlieren würde. Man müsste auch die Winkelerkennung damit verbessern und könnte diese auch noch mehr in dem Bearbeitungsalgorythmus integrieren. Ansonsten wäre noch eine Korrelation mit weiteren Sensoren, z.B. Kameras, Infrarot-Sensoren, Ultraschall-Sensoren oder Lichtschranken sehr interessant. Dieser Ansatz wird deswegen auch in der Automobilbranche verwendet bei der Umgebungsüberwachung, bei der man sich nicht auf ein System verlässt

## Ausblick

Mögliche Ansätze, wie dieses Projekt fortgeführt und verbessert werden könnte, gibt es in hoher Anzahl. Neben der genannten weiteren Integration von mehr Sensoren auch anderer Art und größerem Augenmerk auf den Winkel zum Objekt, kann auch die Software allgemein dieses Projekts deutlich verbessert werden. Dazu könnte eine Parameterstudie, welche denn nun die am besten geeigneten Parameter bei der Aufnahme der Radardaten sind (also z.B. Anzahl von Chirps oder Frames), dienlich sein. Aber auch bei der Nachbearbeitung der Rohdaten ist noch viel Potenzial, um die besten Daten danach weiter zu verarbeiten.

Die Welt der Tracking-Algorithmen, nicht nur in der Radar-Technik, ist zudem sehr groß. Hier könnte neben den schon genannten Filtern wie der Kalman-Filter auch neuere Techniken wie künstliche Intelligenz, maschinelles Lernen oder neuronale Netzwerke von großer Hilfe sein, um die Personen besser aus den Rohdaten zu identifizieren. Die Komplexität allerdings übersteigt den gesetzten Rahmen dieser Arbeit.

# Zusammenfassung

Literaturverzeichnis

**There are no sources in the current document.**

Anhang