

## Objektdetektion mit Datensegmentierung an einem LIDAR auf einem Single-Board-Computer mithilfe des Robot-Operating-System

### Kolloquium

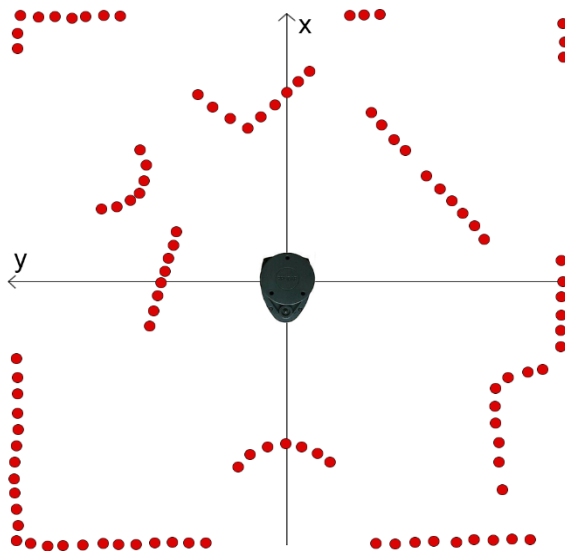
Tim Wennemann, 11110993

# Gliederung

1. Aufgabenstellung
2. Auswahl Segmentierung- und Klassifizierungsansatz
3. Implementierung
  - 3.1 Datenauslesung und Vorverarbeitung
  - 3.2 Segmentierung
  - 3.3 Klassifizierung
  - 3.4 Visualisierung
4. Evaluierung
5. Fazit

# Aufgabenstellung

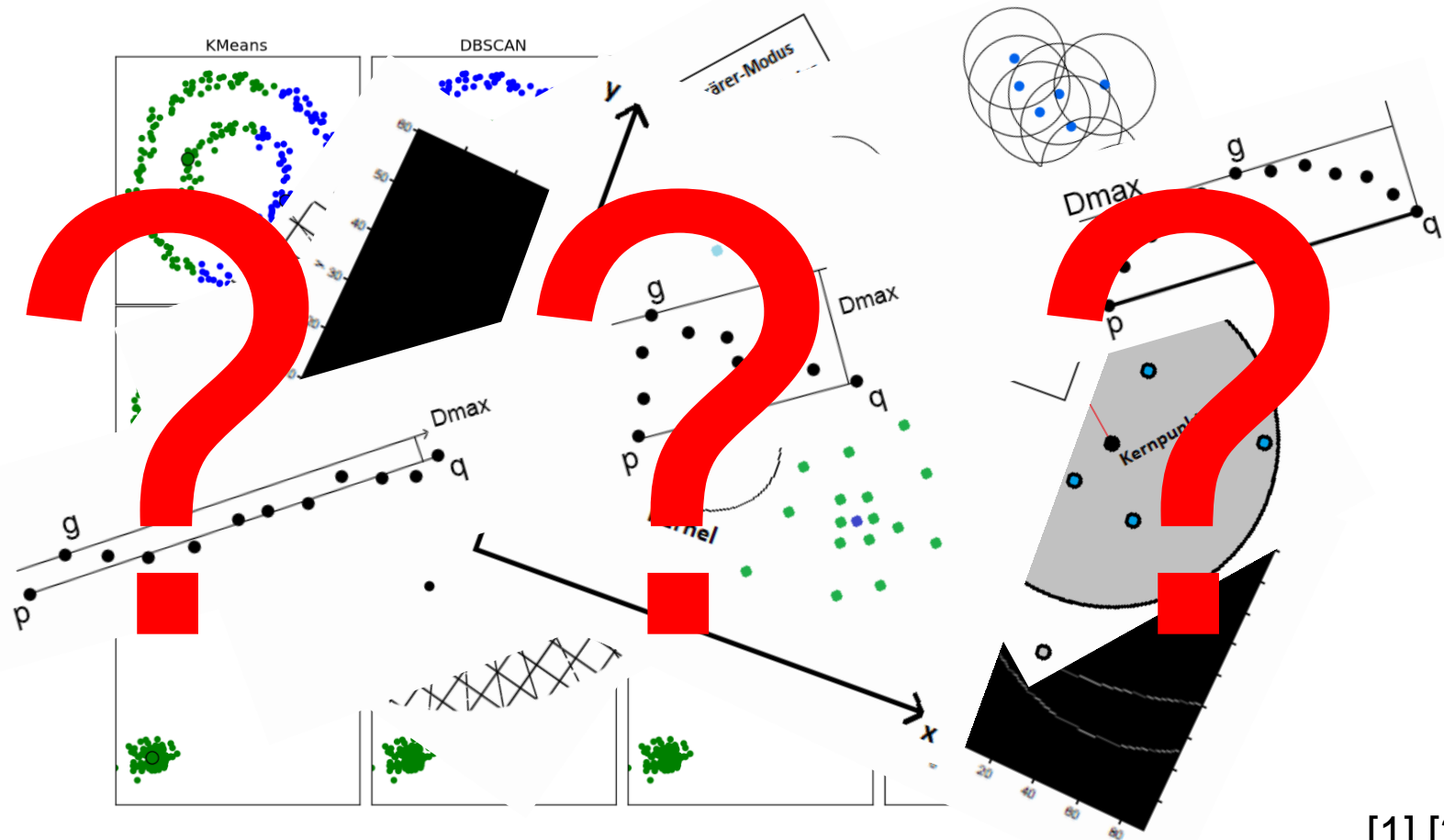
- Auswertung einer 2D Punktwolke, welche durch den 360° 2D Laserscanner RPLIDAR erstellt wurde.
  - Erkennen von Objekten in der Umgebung
  - Objekte werden der Form (rechteckig, kreisförmig) entsprechend Klassifiziert



Weitere Vorgaben:

- Umsetzung auf Raspberry Pi
- Implementierung im ROS
- Selbstgeschriebene Codes in Python
- Robuster Aufbau von Sensor und Raspberry Pi

# Auswahl Segmentierungs- und Klassifizierungsansatz



[1],[2]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans										
Dbscan										
Spektrales Clustering										
MeanShift										
Occupancy Grid + CC-Algorithmus										
RBNN-Segmentation										
ODAOA- Algorithmus										
Hough Transformation										
Graham Scan										

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0									
Dbscan	3									
Spektrales Clustering	0									
MeanShift	3									
Occupancy Grid + CC-Algorithmus	3									
RBNN-Segmentation	3									
ODAOA- Algorithmus	3									
Hough Transformation	3									
Graham Scan	0									

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2								
Dbscan	3	1								
Spektrales Clustering	0	2								
MeanShift	3	1								
Occupancy Grid + CC-Algorithmus	3	3								
RBNN-Segmentation	3	2								
ODAOA- Algorithmus	3	3								
Hough Transformation	3	2								
Graham Scan	0	3								

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2	1							
Dbscan	3	1	3							
Spektrales Clustering	0	2	2							
MeanShift	3	1	1							
Occupancy Grid + CC-Algorithmus	3	3	3							
RBNN-Segmentation	3	2	3							
ODAOA- Algorithmus	3	3	3							
Hough Transformation	3	2	2							
Graham Scan	0	3	3							

[3]



# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2	1	0						
Dbscan	3	1	3	0						
Spektrales Clustering	0	2	2	0						
MeanShift	3	1	1	0						
Occupancy Grid + CC-Algorithmus	3	3	3	0						
RBNN-Segmentation	3	2	3	0						
ODAOA- Algorithmus	3	3	3	2						
Hough Transformation	3	2	2	3						
Graham Scan	0	3	3	0						

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2	1	0	1					
Dbscan	3	1	3	0	0					
Spektrales Clustering	0	2	2	0	0					
MeanShift	3	1	1	0	1					
Occupancy Grid + CC-Algorithmus	3	3	3	0	0					
RBNN-Segmentation	3	2	3	0	0					
ODAOA- Algorithmus	3	3	3	2	2					
Hough Transformation	3	2	2	3	2					
Graham Scan	0	3	3	0	0					

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2	1	0	1	0				
Dbscan	3	1	3	0	0	0				
Spektrales Clustering	0	2	2	0	0	0				
MeanShift	3	1	1	0	1	0				
Occupancy Grid + CC-Algorithmus	3	3	3	0	0	0				
RBNN-Segmentation	3	2	3	0	0	0				
ODAOA- Algorithmus	3	3	3	2	2	3				
Hough Transformation	3	2	2	3	2	3				
Graham Scan	0	3	3	0	0	3				

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2	1	0	1	0	4	8	18	9
Dbscan	3	1	3	0	0	0	7	6	35	6
Spektrales Clustering	0	2	2	0	0	0	4	8	20	8
MeanShift	3	1	1	0	1	0	6	7	28	7
Occupancy Grid + CC-Algorithmus	3	3	3	0	0	0	9	3	45	3
RBNN-Segmentation	3	2	3	0	0	0	8	5	40	4
ODAOA- Algorithmus	3	3	3	2	2	3	16	1	67	1
Hough Transformation	3	2	2	3	2	3	15	2	62	2
Graham Scan	0	3	3	0	0	3	9	3	36	5

[3]

# Auswahl Segmentierungs- und Klassifizierungsansatz

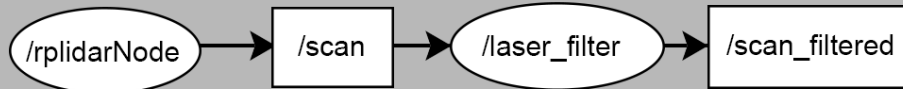
## Gewichtete Punktebewertung

<div>Anforderung</div> <div>Verfahren</div>	Erkennen einer Unbekannten Anzahl von Objekten	Umsetzung auf Kleinrechner möglichst	Eignung auf 2D Punktwolke von RPLIDAR	Klassifizierung von Kreis und Rechteck	Positions-bestimmung	Aussage über Ausrichtung	Ungewichtet		Gewichtet	
							Einfache Punktebewertung		Gewichtete Punktebewertung	
							Summe	Rang	Summe	Rang
Gewichtung	5	5	5	5	3	2				
Kmeans	0	2	1	0	1	0	4	8	18	9
Dbscan	3	1	3	0	0	0	7	6	35	6
Spektrales Clustering	0	2	2	0	0	0	4	8	20	8
MeanShift	3	1	1	0	1	0	6	7	28	7
Occupancy Grid + CC-Algorithmus	3	3	3	0	0	0	9	3	45	3
RBNN-Segmentation	3	2	3	0	0	0	8	5	40	4
ODAOA- Algorithmus	3	3	3	2	2	3	16	1	67	1
Hough Transformation	3	2	2	3	2	3	15	2	62	2
Graham Scan	0	3	3	0	0	3	9	3	36	5

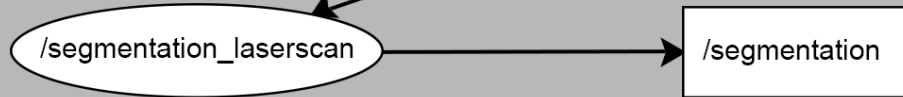
[3]

# Implementierung

## Datenauslesung und Vorverarbeitung



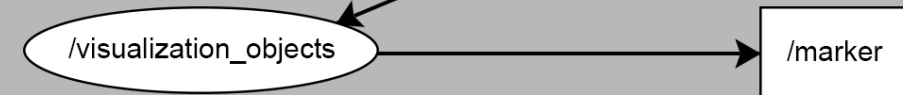
## Segmentierung



## Klassifizierung



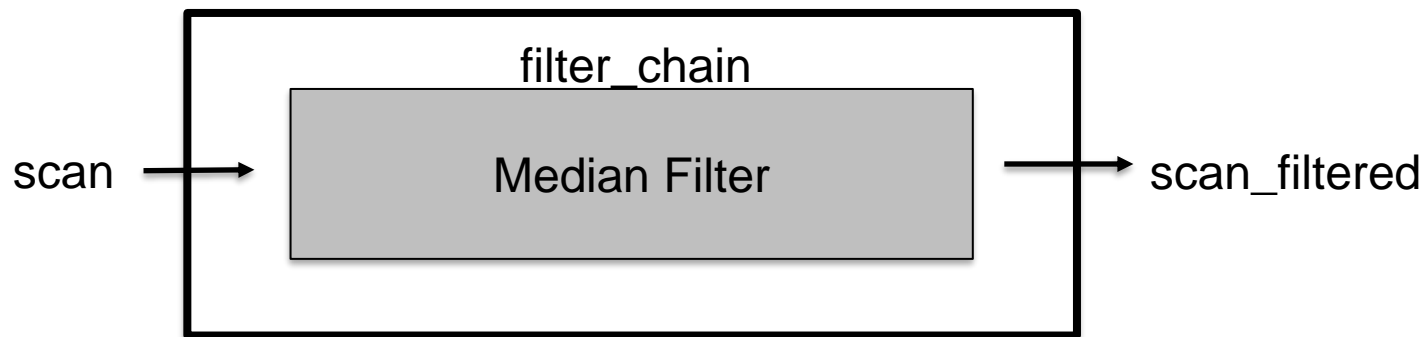
## Visualisierung



# Implementierung

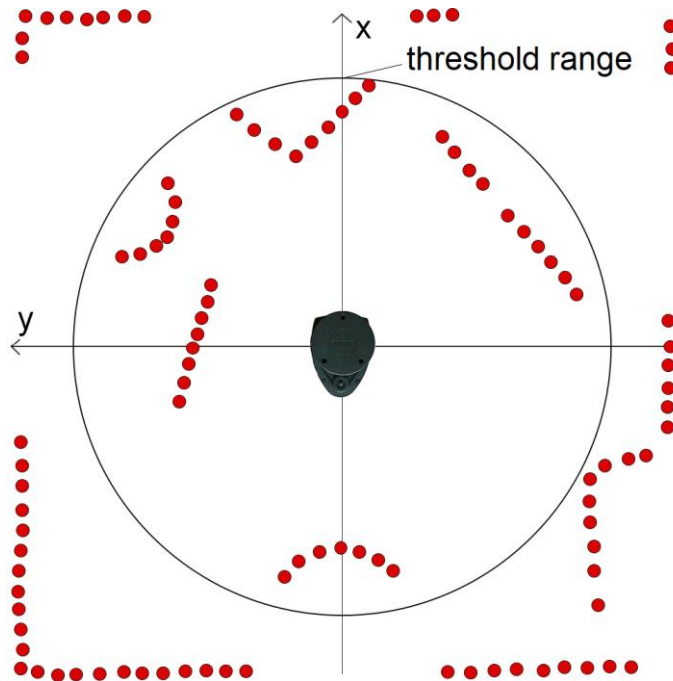
## Datenauslesung und Vorverarbeitung

- Auslesen des „Roh-Scans“ über das RPLIDAR Treiber-Paket „rplidar\_ros“
- Ausgabe als Ros Datentyp „LaserScan“ auf der Topic *scan*
- Filtern von Rauschen/Ausreißern der LaserScan Daten über das Ros package „laser\_filters“.



*Median Filter* vergleicht mehrere Messungen miteinander und erkennt somit Ausreißer, welche von den anderen Messungen Abweichen. [4], [5]

# Implementierung Segmentierung



Messpunkte liegen vor in *ranges* Liste  $R[ ]$ .

Diese enthält die gemessenen Reichweiten von jedem Punkt  $i$ .

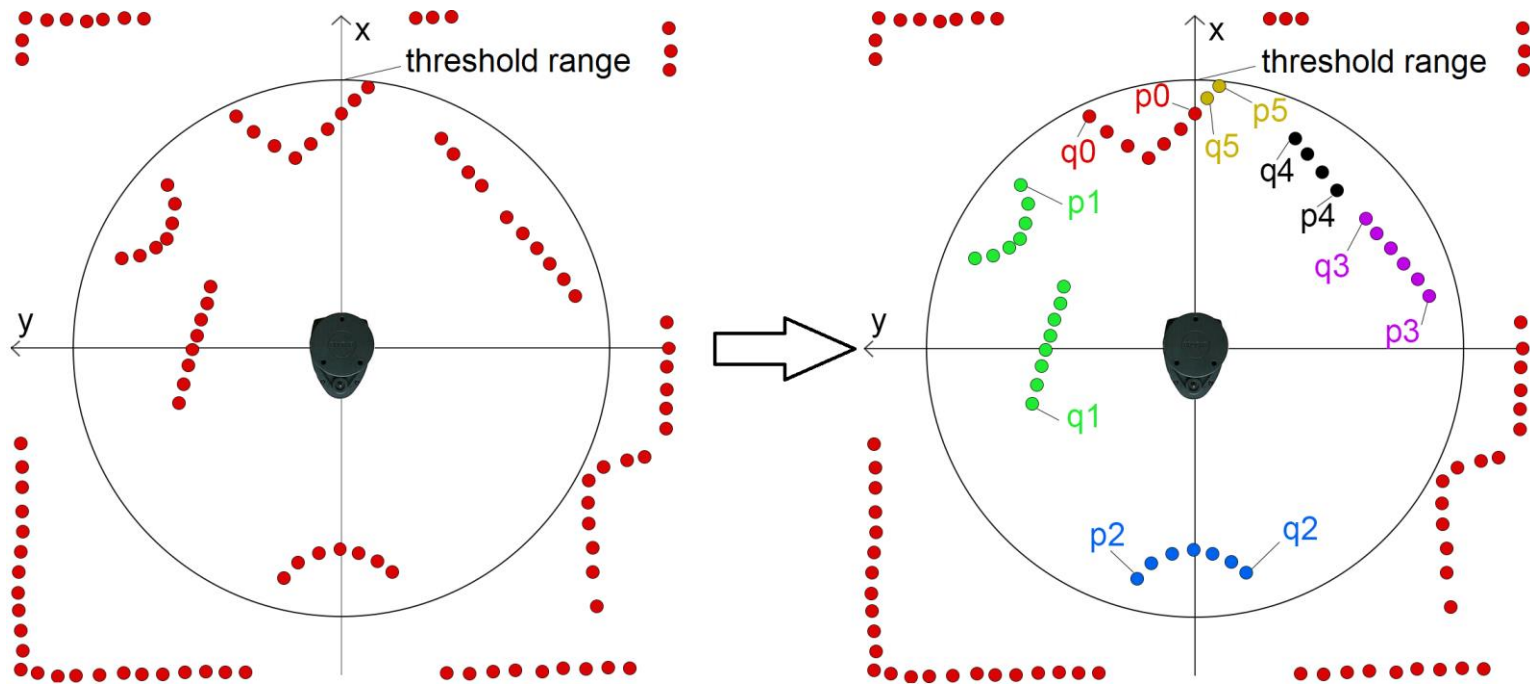
Über  $angle\_increment = 1^\circ$ , welches den Winkelschritt zwischen zwei Messpunkten angibt, können die Messpunkte im Raum platziert werden.



# Implementierung

## Segmentierungsschritt 1

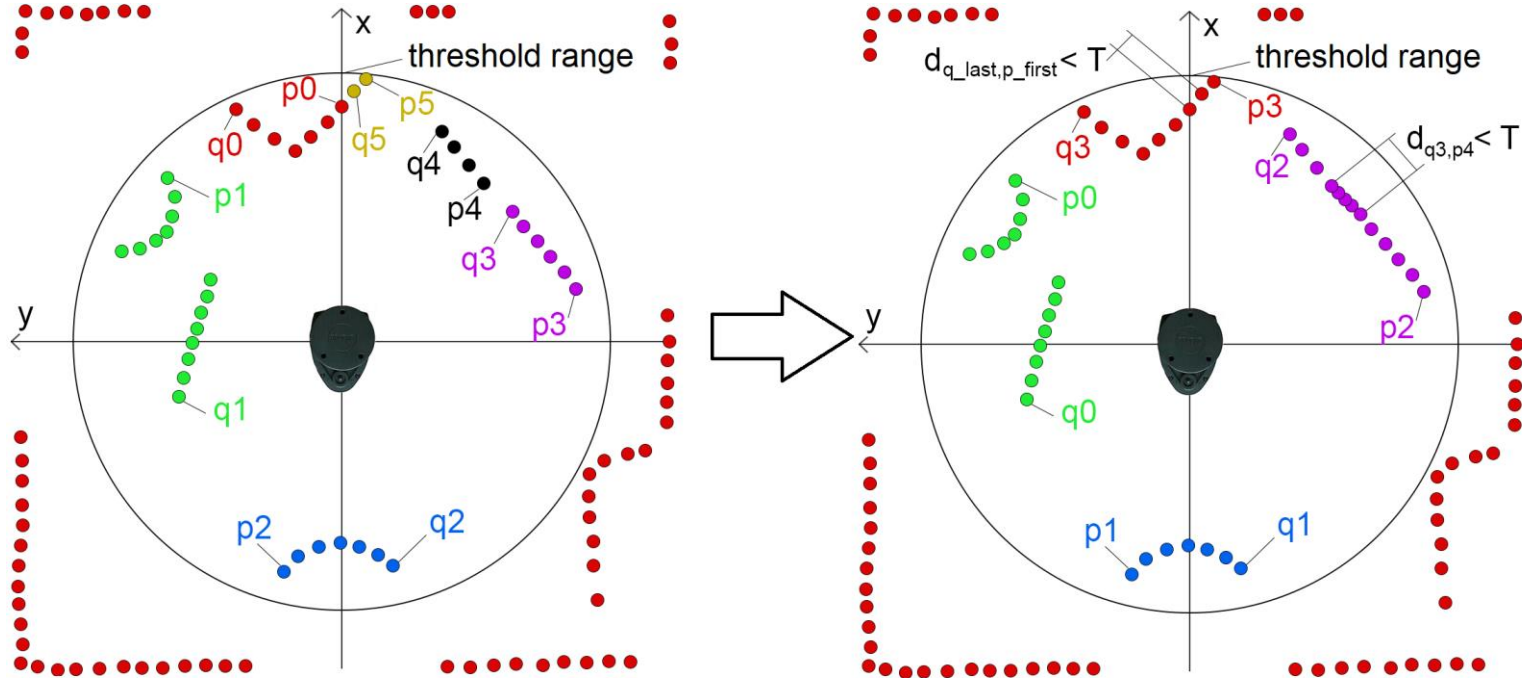
- Festlegen der Segment Start- und Stopppunkte, welche sich innerhalb von *threshold\_range* befinden.



# Implementierung

## Segmentierungsschritt 2

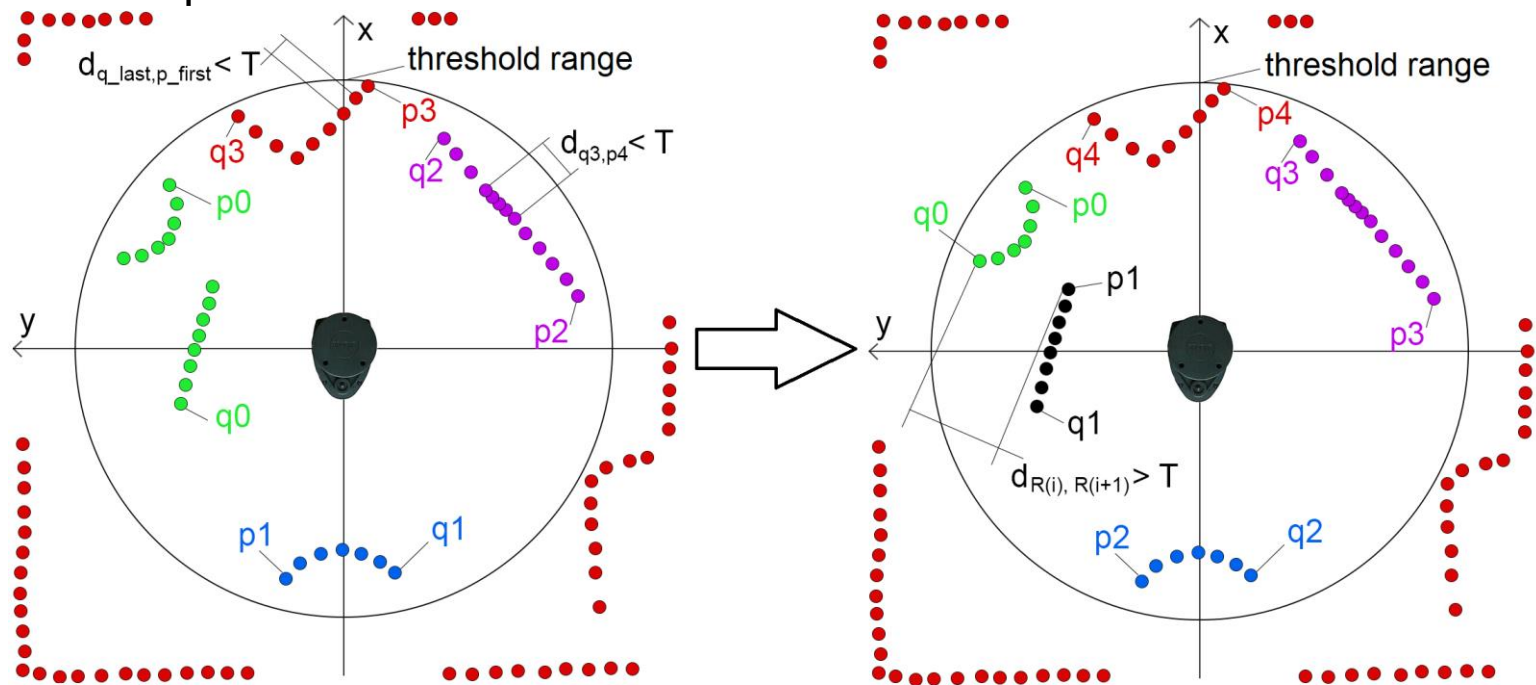
- Überprüfen der Euklidischen Abstände zwischen Stopppunkt  $q(j)$  und Startpunkt  $p(j + 1)$ .
- Überschreiben der ranges-Werte, welche sich zwischen zu verbindenden Stopp- und Startpunkt befinden.



# Implementierung

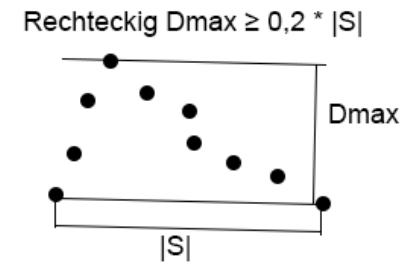
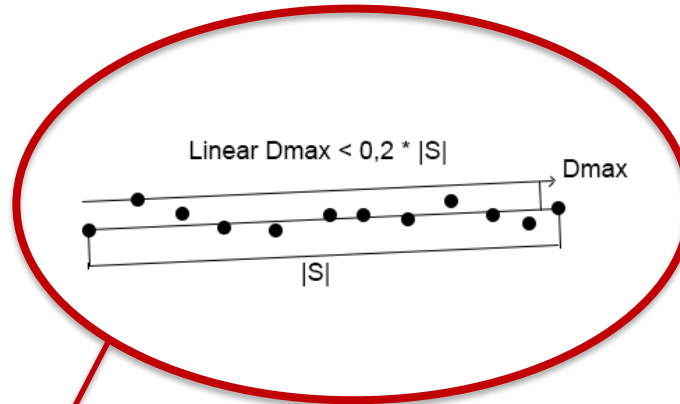
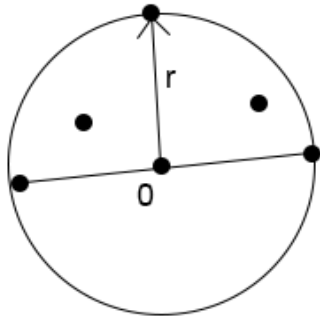
## Segmentierungsschritt 3 + 4

- Überprüfen der Euklidischen Abstände zwischen benachbarten Segmentpunkten  $i$  und  $i + 1$
- Entfernen von Segmenten mit einer zu geringen Anzahl an Messpunkten.

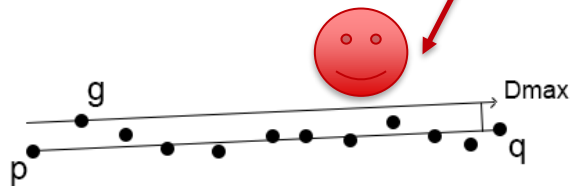


# Implementierung Klassifikation

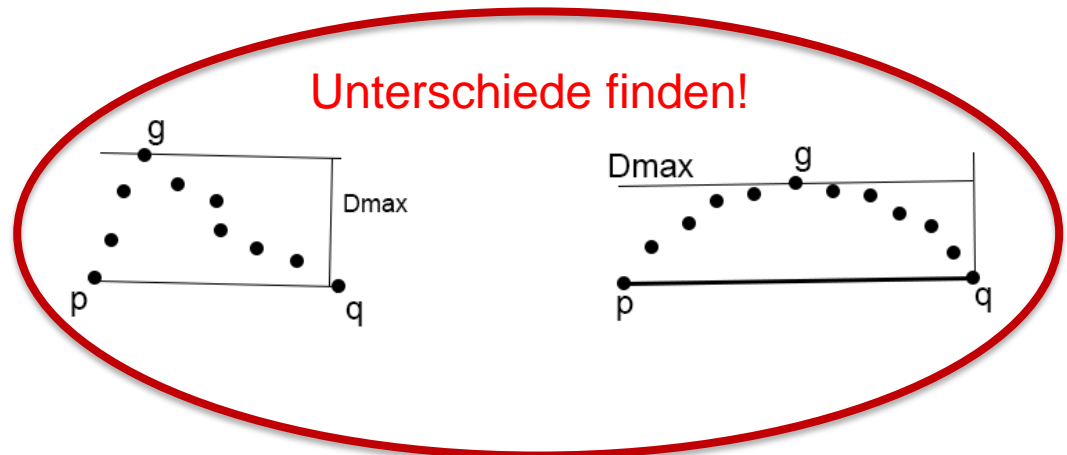
ADAOA- Algorithmus:



Eigene Umsetzung:

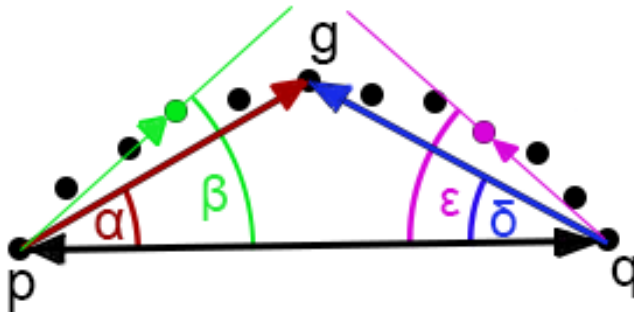


Unterschiede finden!



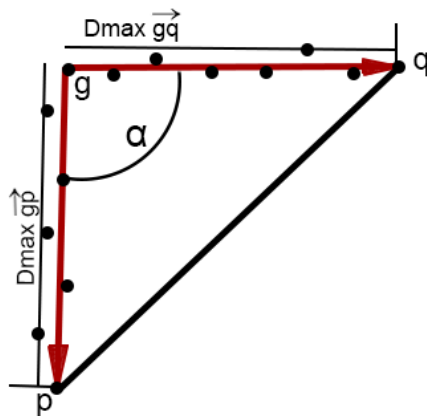
# Implementierung Klassifikation

## Kreisklassifikation



Bei einem Kreis müssen die Winkel  $\beta$  und  $\varepsilon$  größer sein, als die jeweils zu vergleichenden Winkel  $\alpha$  und  $\delta$ .

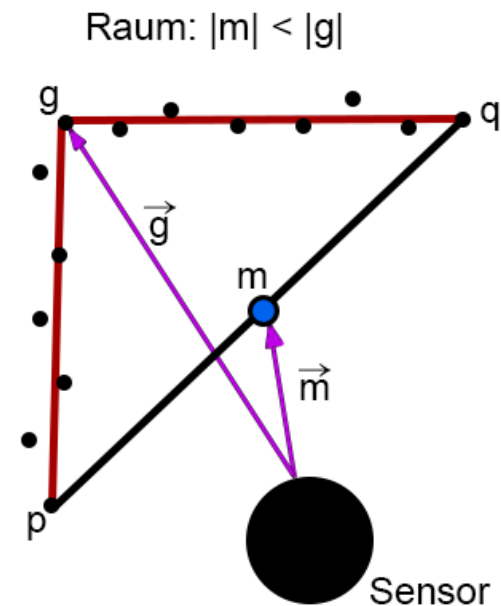
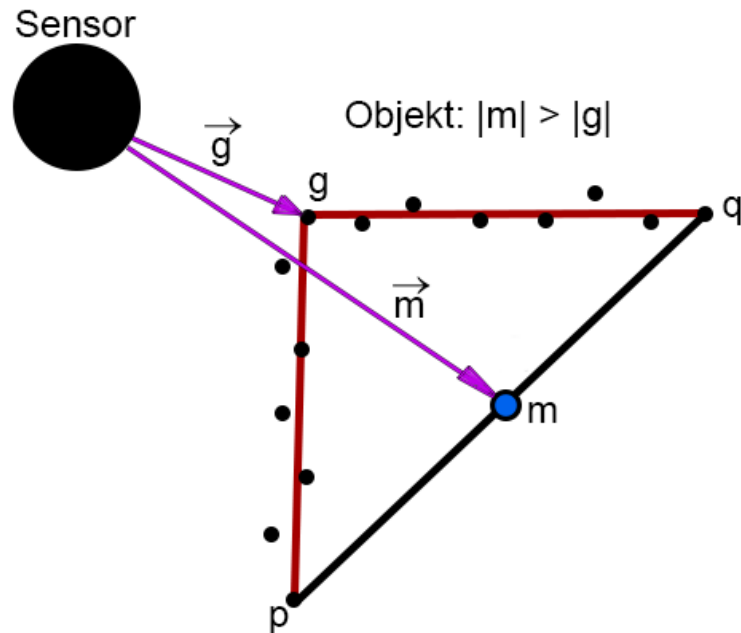
## Rechteckklassifikation:



- „Kanten“ müssen als Linien klassifiziert werden.
- Winkel  $\alpha$  muss zwischen  $80^\circ$  und  $100^\circ$  liegen.

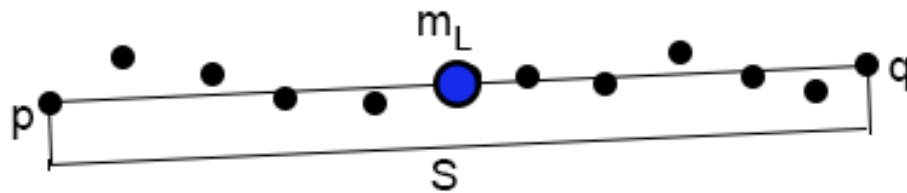
# Implementierung Klassifikation

## Objekt- oder Raumklassifizierung

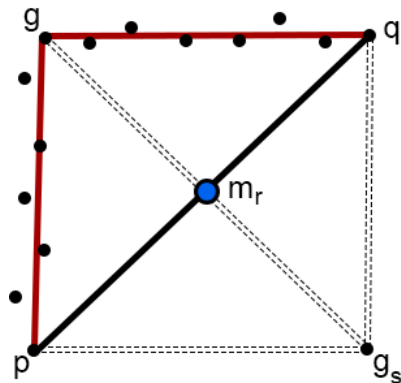


# Implementierung

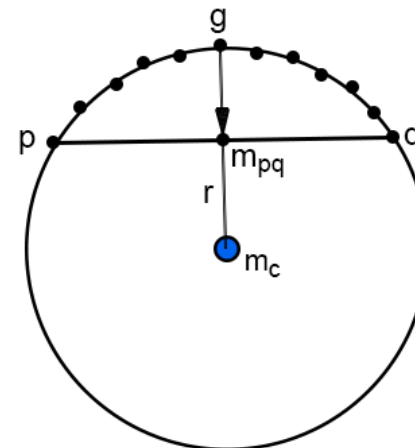
## Position und Maße der Objekte



Übergabe von  $p, q$  und  $m_L$ .

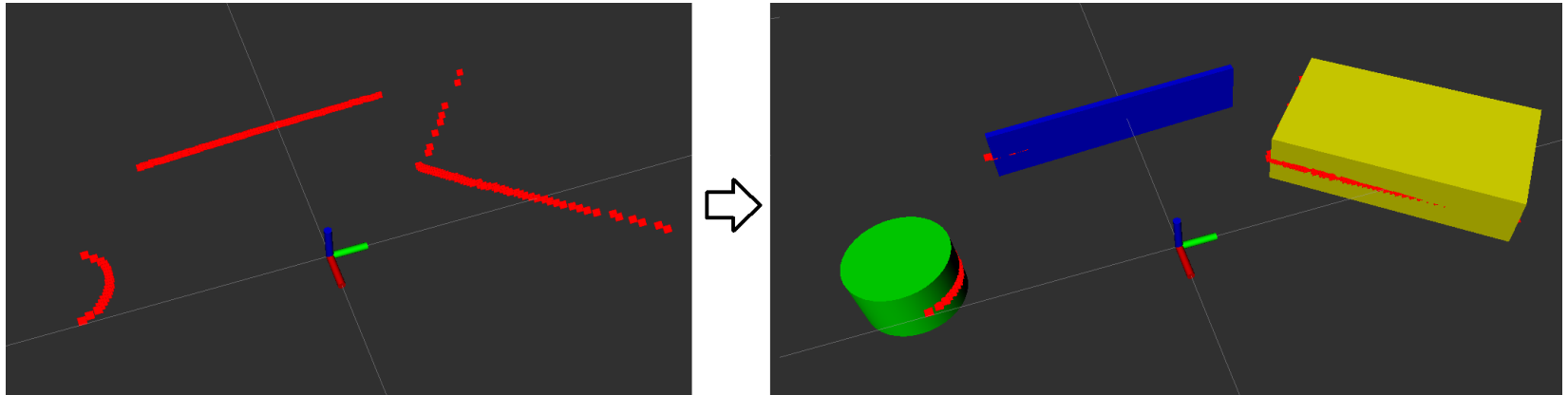


Übergabe der Eckpunkte sowie des Mittelpunktes.



Übergabe von Mittelpunkt und Kreisradius.

# Implementierung Visualisierung

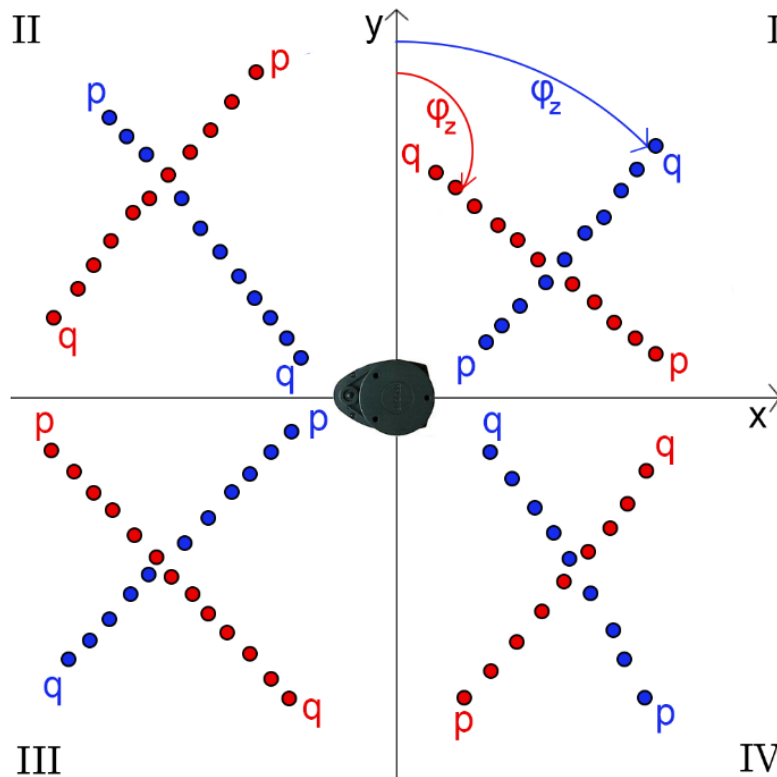


- Darstellung der Klassifizierten Objekte im *rviz*.
- Verwendung des ROS Datentyps „Marker“



# Implementierung Visualisierung

Marker Orientierung:



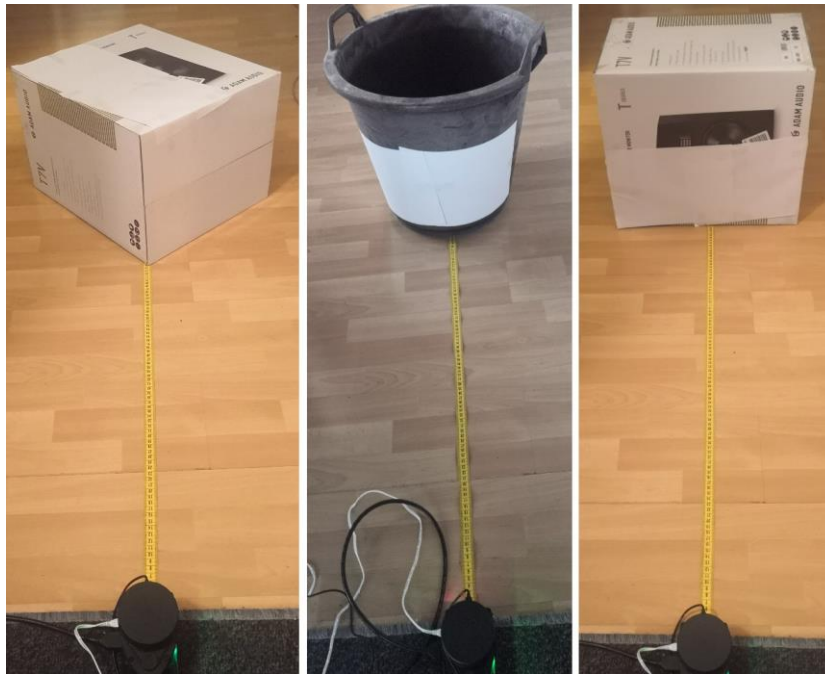
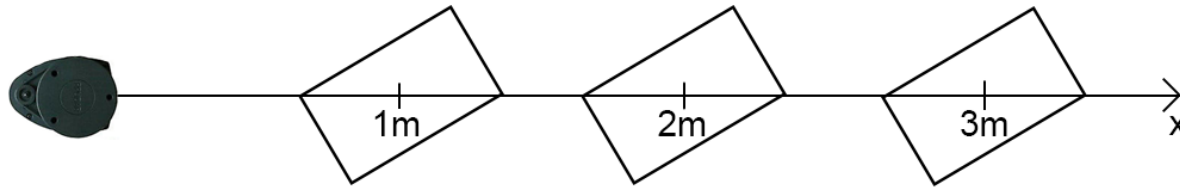
Je nach Objektpositionierung zum Sensor müssen Vektoren  $\overrightarrow{pq}$  **oder**  $\overrightarrow{qp}$  verwendet werden.



Muss an den jeweiligen Quadranten angepasst werden!

# Evaluierung

## Versuchsbeschreibung



Drei verschieden zu klassifizierende Objekte werden jeweils bei 1,2 und 3 Meter Abstand gemessen.

20 Messungen pro Objekt & Entfernung.

Prüfen auf Objekterkennung, Positionierung, sowie Maßberechnung.

# Evaluierung

## Versuchsergebnis

Objekt	Rund Eimer						
Durchmesser	37cm						
Oberfläche	Papier						
Farbe	Weiß						
Realer Abstand R	Erkannt zu	$\bar{R}$ [cm]	$\sigma_R$ [cm]	$\bar{\varphi}$ [°]	$\sigma_{\varphi}$ [°]	$\bar{D}$ [cm]	$\sigma_D$ [cm]
100cm	100%	100,5	1	-0,5	0,9	37,8	2,2
200cm	100%	199,5	1,9	-0,2	0,9	38,3	3,8
300cm	75%	292,4	1,8	-0,5	1	30,8	5,9

Objekt	Rechteck Karton								
Länge a	46cm								
Breite b	39,5cm								
Oberfläche	Papier								
Farbe	Weiß								
Realer Abstand R	Erkannt zu	$\bar{R}$ [cm]	$\sigma_R$ [cm]	$\bar{\varphi}$ [°]	$\sigma_{\varphi}$ [°]	$\bar{a}$ [cm]	$\sigma_a$ [cm]	$\bar{b}$ [cm]	$\sigma_b$ [cm]
100cm	100%	98,7	0,4	0,4	0,4	43,5	0,9	37,6	1,1
200cm	100%	196,8	1,2	1,9	0,4	41,7	2,8	35,0	1,8
300cm	85%	296,8	2,2	-0,6	0,3	41,7	5,0	34,6	2,2

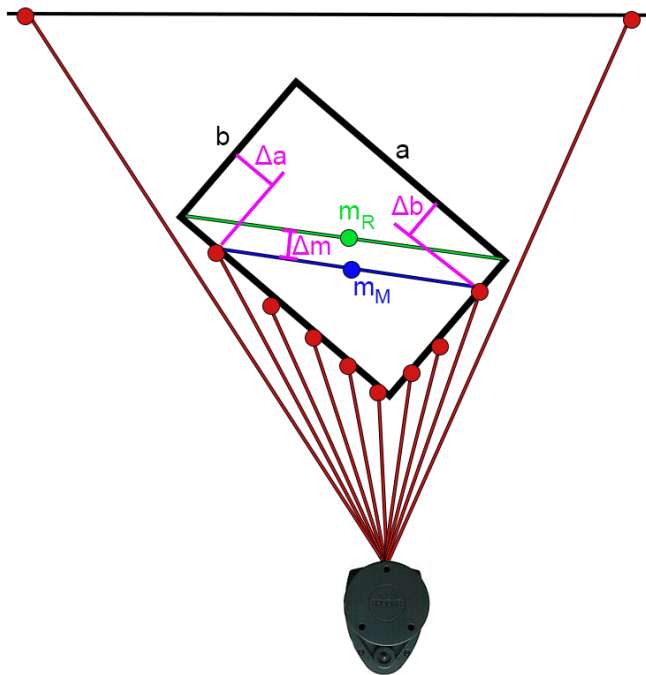
# Evaluierung

## Versuchsergebnis

Objekt	Linie Karton						
Länge	46cm						
Oberfläche	Papier						
Farbe	Weiß						
Realer Abstand	Erkannt zu	$\bar{R}$ [cm]	$\sigma_r$ [cm]	$\bar{\varphi}$ [°]	$\sigma_\varphi$ [°]	$\bar{a}$ [cm]	$\sigma_a$ [cm]
100cm	100%	100,0	0,2	-0,8	0,3	45,0	1,3
200cm	100%	199,8	0,2	-1,0	0,1	42,0	0,0
300cm	100%	299,6	0,3	-1,0	0,1	41,7	1,2

- Alle Objekte werden bei 1 und 2 Meter Abstand zu 100% erkannt.
- Objektmaße und Positionierung können mit geringen Abweichungen bestimmt werden.
- Mit steigenden Abständen verschlechtert sich das Messergebnis.

# Evaluierung Fehlerquelle




- Messtoleranzen
- Abweichungen im Versuchsaufbau
- $\Delta a$ ,  $\Delta b$  und  $\Delta m$  resultieren aus den Winkelschritten zwischen den Messpunkten.



- Mit steigendem Abstand - größere Messfehler

# Fazit

- Objektsegmentierung sowie Klassifizierung funktioniert mit einem 2D Laserscanner!
- Verbesserungsmöglichkeiten der Software → Verminderung der Fehler
- Einsatz in der Fahrzeugtechnik?!
  - Einsatz mehrerer 2D Laserscanner, um aus 2D Ebenen eine 3D Punktwolke erstellen zu können
  - Laserscanner zur Segmentierung und Positionieren verwenden, zur Klassifizierung Kamertechnik verwenden.
- Persönlich: Großer Lerneffekt! 

## Quellen

- [1] **scikits learn. (o.J.): Comparing different clustering algorithms on toy datasets.** [https://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/auto\\_examples/cluster/plot\\_cluster\\_comparison.html](https://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/auto_examples/cluster/plot_cluster_comparison.html) - zuletzt abgerufen am 25.07.2020
- [2] **O.V.: Line Detection by Hough transformation**, S. 1-7, 2009
- [3] **Technische Universität Braunschweig. (o.J.): Gewichtete Punktebewertung.** <https://methodos.ik.ing.tu-bs.de/methode/GewichtetePunktebewertung.html> - zuletzt abgerufen am 25.07.2020
- [4] **O.V. (o.J.): rplidar package.** <http://wiki.ros.org/rplidar> - zuletzt abgerufen am 25.07.2020
- [5] **Tully Foote. (o.J.): laser\_filters package.** [http://wiki.ros.org/laser\\_filters](http://wiki.ros.org/laser_filters) - zuletzt abgerufen am 25.07.2020