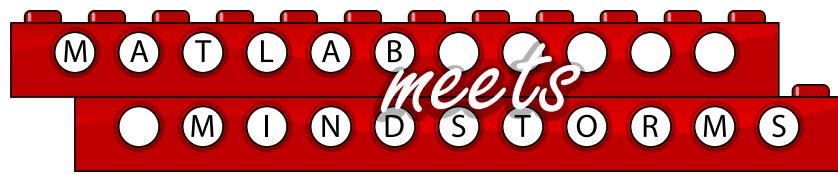


Projekt der Elektrotechnik und Informationstechnik



6. Projektversuch

- Ultraschallsensor -

5. Dezember 2016

Inhaltsverzeichnis

1 Einleitung	1
2 Grundlagen	2
2.1 Ultraschall-Messtechnik	2
2.2 Mindstorms EV3 Ultraschallsensor	2
2.3 MATLAB-Funktionen der Toolbox	3
2.4 Bestimmung des Bremsweges	3
2.5 Umgebungssensor	7
3 Durchführung	9

1 Einleitung

Mithilfe des Ultraschallsensors kann ein fahrbarer Roboter Hindernissen ausweichen, ohne diese mit einem Kontaktschalter berühren zu müssen. Wird der Ultraschallsensor drehbar auf dem Roboter angebracht, kann ein Scan der Umgebung durchgeführt werden, bevor der Roboter selbst bewegt wird.

Im Rahmen dieses Versuchs soll die Funktionsweise des Ultraschallsensors mit vorgegebenen MATLAB-Funktionen aus der RWTH-Mindstorms-EV3-Toolbox untersucht werden. Dabei werden auch die Beschränkungen des Sensors hinsichtlich des Winkelbereichs, in dem der Sensor Entfernung zuverlässig messen kann, sowie die Abhängigkeit der Messwerte von der Form der reflektierenden Fläche und ggf. vom Material betrachtet. Der Ultraschallsensor wird dann benutzt, um einen Roboter aufzubauen, der anhand der Messwerte in einem vorgegebenen Abstand vor einem Objekt anhält. Unter Verwendung dieses Roboters werden Versuche zur Bestimmung des Bremswegs durchgeführt. Das von der Fahrgeschwindigkeit abhängige Nachlaufen der Fahrmotoren kann über die in den Motoren eingebauten Tachozähler ausgelesen werden und dient zur Berechnung eines geschwindigkeitsabhängigen Bremspunktes. Der Zusammenhang zwischen Bremsweg und Fahrgeschwindigkeit ist auszuwerten und grafisch darzustellen.

Weiterhin soll mit Hilfe eines drehbaren Ultraschallsensors oder eines sich auf der Stelle drehenden Roboters ein radarähnlicher Scan der Umgebung durchgeführt werden. Dabei ist ggf. konstruktiv zu beachten, dass das Kabel zum Anschluss des Ultraschallsensors beim Drehen nicht aufgewickelt wird. Dies soll genutzt werden, um den Ausfahrtwinkel des Roboters aus einem geschlossenen Raum mit Ausgang zu bestimmen.

2 Grundlagen

2.1 Ultraschall-Messtechnik

Die Bestimmung von Entfernungen mit Ultraschall basiert auf der Messung der Laufzeit eines ausgesendeten Schallimpulses zwischen dem Sensor und dem reflektierenden Objekt. Abbildung 1 zeigt das Prinzip der Entfernungsmessung mittels Ultraschall.

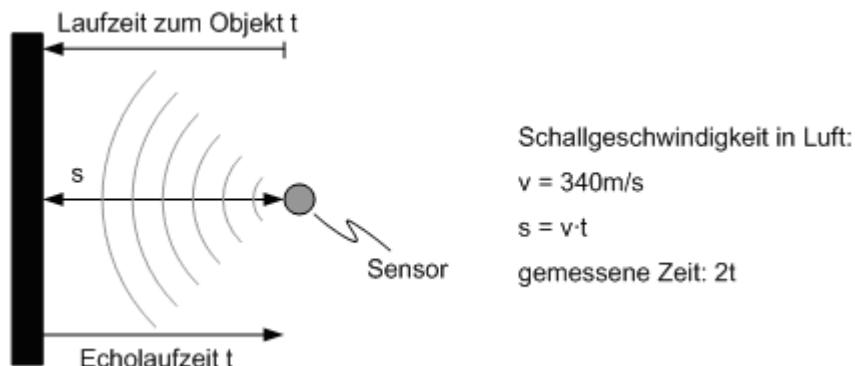


Abbildung 1: Prinzip der Ultraschallmessung

Die Frequenz der in Luft ausgesendeten Schallwellen liegt typischerweise bei 30kHz bis 60kHz und ist somit für das menschliche Gehör nicht wahrnehmbar. Eine bekannte Anwendung sind die Abstandswarner in den Stoßfängern moderner Automobile. Dabei werden die Schallwandler sowohl zum Senden als auch für den Empfang benutzt. Zur Bestimmung der Entfernung vom reflektierenden Objekt geht man von einer Schallgeschwindigkeit von 340m/s (in Luft) aus.

2.2 Mindstorms EV3 Ultraschallsensor

Der Mindstorms EV3 Ultraschallsensor ist mit zwei Schallwandlern aufgebaut, die nebeneinander angeordnet sind. Der Schaltplan des Sensors zeigt, dass einer der beiden Schallwandler (der linke) nur zum Senden und der andere nur zum Empfangen verwendet wird. Der Sensor verfügt über einen eigenen Mikrocontroller, der die Daten aus der Ultraschallmessung aufbereitet und als Entfernungsangabe (in cm oder inch) mit einer Auflösung von 0,1 cm bzw inch ausgibt. Die Präzision der Werte beträgt +/-1cm. Die Sendevorgänge werden automatisch ausgelöst und die gemessenen Entfernung dabei regelmäßig aktualisiert. Während der Ultraschallsensor sendet, leuchten die beiden roten Ringe um den Schallwandlern. Die Amplitude der Echos wird nicht erfasst.



Abbildung 2: LEGO Mindstorms EV3 Ultraschallsensor (©LEGO)

Nach unseren Informationen sendet der Mindstorms-Ultraschallsensor bei einem Sendevorgang 11 aufeinander folgende Pulse bei einer Grundfrequenz von 40kHz. Damit dauert die gesamte Pulsfolge $275\mu s$. Der Empfang der Schallechos beginnt, sobald die Pulsfolge vollständig gesendet ist. Daher können nur Echos ausgewertet werden, deren Laufzeit mehr als $275\mu s$ beträgt. Dies entspricht einer Laufstrecke von 9,35cm in Luft. Berücksichtigt man Hin- und Rückweg, so ergibt sich eine minimal messbare Entfernung von 4,7cm. Abbildung 2 zeigt ein Bild des LEGO Mindstorms EV3 Ultraschallsensors.

2.3 MATLAB-Funktionen der Toolbox

Der Ultraschallsensor wird an einen Sensorport des Bricks angeschlossen. Über das entsprechende Portelement kann dann der Sensormodus (cm oder inch) gewählt und der aktuelle Messwert ausgelesen werden. Die Funktion `mode` des Sensorobjektes wählt den Sensormodus, so dass anschließend mit der Funktion `value` Daten ausgelesen werden können. Standardmäßig ist für den Modus cm eingestellt. Um den Sensor in den inch-Modus zu versetzen, lautet der Aufruf z.B. wenn der Sensor am Port4 des EV3-Objekts `ev3` angeschlossen ist:

```
ev3.sensor4.mode = DeviceMode.UltraSonic.DistIN;
```

Der Befehl `value` gibt bei Aufruf mit

```
s = ev3.sensor4.value;
```

die gemessene Entfernung `s` in cm oder inch zurück. Der Ultraschallsensor liefert Entfernungswerte bis maximal 255. Der Wert 255 wird geliefert, wenn kein Echo erfasst wurde. Dieses Verhalten ist bei der Durchführung zu überprüfen.

2.4 Bestimmung des Bremsweges

Für den fahrbaren Roboter, der im Rahmen dieses Versuches verwendet wird, ist der Zusammenhang zwischen Geschwindigkeit und Bremsweg zu ermitteln. Dazu soll der Roboter in mehreren Durchläufen mit unterschiedlichen Geschwindigkeiten auf ein Hindernis

zufahren und kontinuierlich den Abstand messen. Um die tatsächlich gefahrene Geschwindigkeit des Roboters zu ermitteln, müssen die Fahrmotoren im geregelten Modus betrieben werden. Dies erfolgt durch die Eigenschaft `motor.SpeedRegulation = 'On'`. In diesem Fall ergibt sich die Winkelgeschwindigkeit des Motors als lineare Funktion des eingestellten Power-Wertes `f_speed`. Die Fahrgeschwindigkeit des Roboters kann aus einer Messung des Radumfangs, einer Zeitmessung und dem Auslesen des Tachocounts erfolgen. Für einen Motor wurde im ungeregelten und geregelten Betrieb die tatsächliche Winkelgeschwindigkeit in Abhängigkeit des am Motor eingestellten Power-Wertes (0-100) gemessen. Bei einer vollen Umdrehung des Motors erhöht sich der Tachocount des Motors um 360. Die Ergebnisse der Messung sind in Abbildung 3 dargestellt. Man erkennt, dass im geregelten Betrieb die Winkelgeschwindigkeit linear vom eingestellten Power-Wert abhängt mit

$$\text{Winkelgeschwindigkeit [Grad/s]} = 10 \cdot \text{POWER}$$

Abhängig von der Last des Motors ergibt sich eine unterschiedliche Ausprägung des Kurvenverlaufs und der lineare Anstieg beim geregelten Betrieb endet bei einem bestimmten Power-Wert. Der lineare Kurvenverlauf muss für einen gegebenen Roboter individuell bestimmt werden.

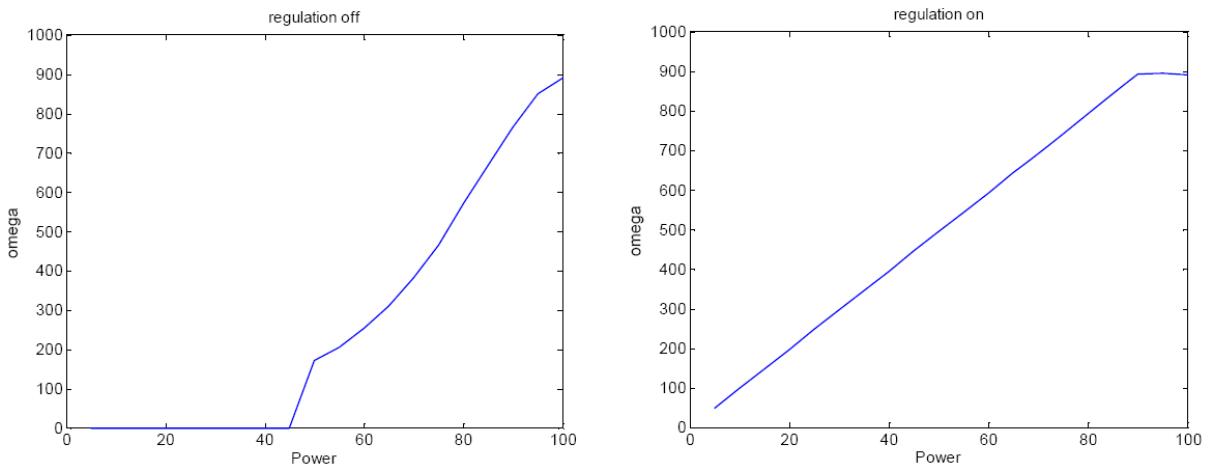


Abbildung 3: Winkelgeschwindigkeit eines einzelnen Motors im Leerlauf in Abhängigkeit des Power-Wertes. (links: ungeregelt, rechts: geregelt)

Die tatsächlich gefahrene Geschwindigkeit lässt sich durch wiederholtes Auslesen des Winkelsensors und der Bestimmung der Zeit (in Sekunden) zwischen zwei Messungen mit den MATLAB-Befehlen `tic` und `toc`. `toc` misst immer die Zeit seit dem letzten `tic` in Sekunden. `toc` kann für ein `tic` auch mehrfach verwendet werden.

```
tic;
<BEFEHLE>
delta_t = toc; % Zeit für Ausführung von <BEFEHLE> in Sekunden
```

Beim Versuch sollen die beiden Fahrmotoren synchron betrieben werden, damit das Fahrzeug beim Starten und Bremsen keine Abweichung in eine Richtung bekommt. Zwei Motoren A und B startet man z.B. durch den Aufruf:

```

motorR = ev3.motorA;
motorL = ev3.motorB;

motorR.brakeMode = 'Brake';
motorR.speedRegulation = 'Off'; % necessary for synced start
motorR.power = 20;

motorR.syncedStart(motorL, 'turnRatio', 0);

```

Mit `motorR.syncedStop()` können die Motoren auch synchron angehalten werden. Mittels des `turnRatio` kann man Parallelität bzw. Gegenläufigkeit der beiden Motoren steuern. bei einem Wert von 0, drehen beide Motoren mit gleicher Geschwindigkeit in die gleiche Richtung.

Wenn die mit dem Ultraschallsensor gemessene Entfernung einen vorgegebenen Wert erreicht, sollen die Fahrmotoren gestoppt und der aktuelle Tachozähler der Fahrmotoren ausgelesen werden. Der zusätzlich zurückgelegte Bremsweg kann durch ein Maßband oder durch das Auswerten des Ultraschallsensors nach dem Ausrollen ermittelt werden. Abbildung 4 verdeutlicht den entsprechenden Aufbau. Um die Länge des Bremsweges messen zu können, muss die automatische Motorbremse ausgeschaltet werden (`motor.brakeMode = 'Coast'`).

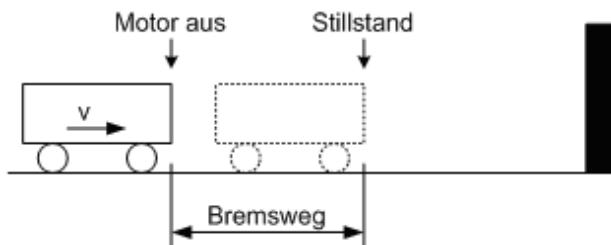


Abbildung 4: Bremsweg

Für die Länge des Bremsweges wird ein einfaches physikalisches Modell vorgeschlagen: Unter der Annahme, dass der ausgeschaltete Motor ein konstantes Bremsmoment erzeugt und die kinetische Energie des Fahrroboters quadratisch von der gefahrenen Geschwindigkeit abhängt, kann man einen quadratischen Zusammenhang zwischen der Länge des Bremsweges und der gefahrenen Geschwindigkeit annehmen.

2 Grundlagen



Abbildung 5: Aufbau zur Bestimmung des Bremsweges

Unter Berücksichtigung dieses Zusammenhangs kann die Ansteuerung des Roboters so gewählt werden, dass dieser in einem vorgegebenen Abstand vor einem Hindernis stehen bleibt. Dazu muss berechnet werden, wie lang (bei einer gegebenen Geschwindigkeit) der zu erwartende Bremsweg ist, und der Motor entsprechend früher abgeschaltet werden. Abbildung 5 zeigt einen entsprechenden Versuchsaufbau mit einem fahrbaren Roboter.

Der Bremsweg s des Roboters ermittelt sich wie folgt:

$$s = \frac{1}{2}at^2 \quad \text{mit} \quad t = \frac{v_0}{a}$$

Dabei ist a die im Idealfall konstante Bremsbeschleunigung und v_0 die Geschwindigkeit zu Beginn des Abbremsens. Daraus ergibt sich für die Länge des Bremsweges eine quadratische Abhängigkeit von der Ausgangsgeschwindigkeit v_0 .

Die Bestimmung eines mathematischen Zusammenhangs zwischen Geschwindigkeit und Bremsweg auf Basis der gemessenen Werte erfolgt mit dem MATLAB-Befehl `polyfit`. Durch Aufruf mit

```
p = polyfit(x,y,n);
```

werden die x- und y-Werte durch folgendes Polynom der Ordnung n angenähert:

$$y = p(1) \cdot x^n + p(2) \cdot x^{n-1} + \dots + p(n) \cdot x + p(n+1)$$

Die Annäherung erfolgt dabei mit der Methode kleinster Fehlerquadrate, bei welcher der Fehler zwischen der gemessenen Kurve und der Polynomapproximation minimiert wird. Der Rückgabewert von `polyfit` ist ein Vektor mit $(n + 1)$ Elementen, der die Koeffizienten des Polynoms enthält. Im Versuch soll der Bremsweg durch ein Polynom 2. Grades angenähert werden.

2.5 Umgebungssensor

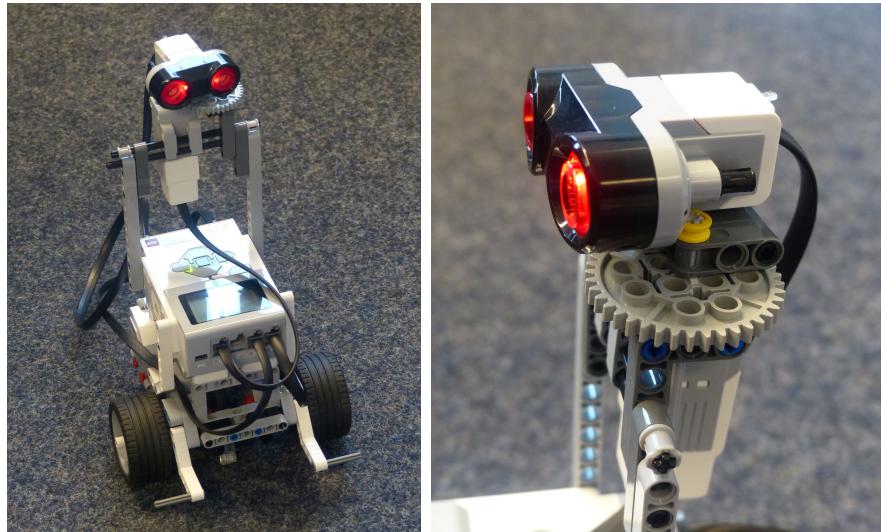


Abbildung 6: Modell mit drehbarem Sensor

Zur Erstellung eines Profils der Umgebung soll der Ultraschallsensor den gesamten Winkelbereich von 360° erfassen. Dazu kann entweder der Sensor mit einem weiteren Motor drehbar montiert werden (Siehe Abbildung 6). Dabei ist konstruktiv zu beachten, dass das angeschlossene Sensorkabel sich entsprechend aufwickeln kann. Alternativ kann auch der Roboter selbst eine 360° -Drehung durchführen. Dazu müssen sich die Fahrmotoren gegenläufig um einen Winkel drehen, der abhängig von der Bauweise (Reifengröße, Achsabstand, Drehpunkt) und dem Untergrund experimentell ermittelt werden muss. In beiden Fällen sollen möglichst viele Sensordaten und der Blickwinkel aus der dazu gehörenden Position des Motors aufgenommen werden. Abbildung 7 verdeutlicht den entsprechenden Aufbau.

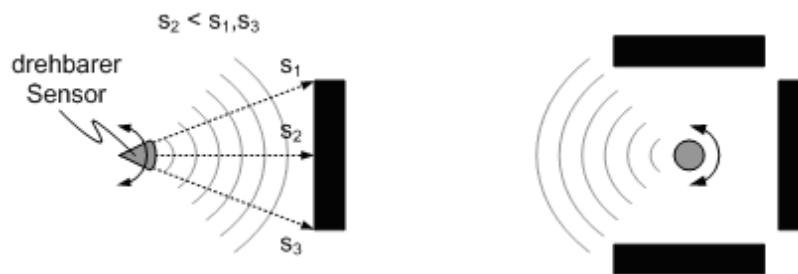


Abbildung 7: Umgebungsscanner

Die aufgenommenen Daten können unter Verwendung der MATLAB-Funktionen `polarplot` dargestellt werden:

2 Grundlagen

```
polarplot(theta,rho);
```

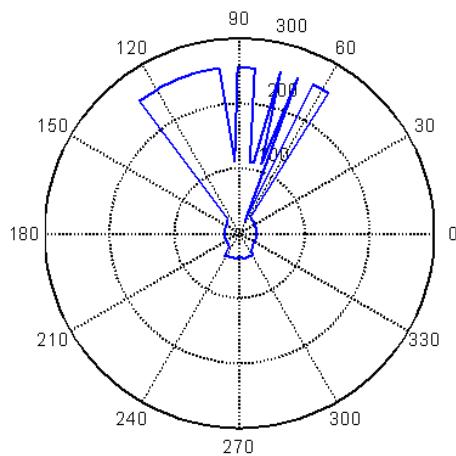


Abbildung 8: Polar-Plot des mit Ultraschall gemessenen Entfernungsprofils

Dabei ist `rho` ein Vektor mit den Abstandswerten und `theta` ein Vektor mit den entsprechenden Winkeln im Bogenmaß. Abbildung 8 zeigt das Diagramm eines Beispielprofils, das innerhalb eines zu einer Seite geöffneten Kartons aufgenommen wurde (siehe Abbildung 9). In diesem Beispiel wurden die Originalwerte dargestellt, die maximal den Wert 255 annehmen. Um einen Profilverlauf mit einer deutlicheren Trennung zwischen Hindernissen und Öffnungen zu erhalten, kann man z.B. alle Messwerte größer als 100 auf 100 begrenzen. Das lässt sich in Matlab einfach und effizient so formulieren:

```
d(d>100) = 100;
```



Abbildung 9: Versuchsaufbauten zum Umgebungsscanner

3 Durchführung

- a) Schließen Sie den EV3-Ultraschallsensor z.B. an Port 4 des EV3-Bricks an und lesen Sie die Messwerte mit Hilfe der Funktion `value` aus. Nehmen Sie eine Messreihe mit Entfernung zu einem festen Hindernis auf und bestimmen Sie zu den mit dem Ultraschallsensor gemessenen Werten auch den tatsächlichen Abstand mit Hilfe eines Maßbands. Bestimmen Sie dazu auch die Lage des Nullpunkts im Sensor. Tragen Sie die Werte in die Tabelle ein und stellen Sie die Ergebnisse in MATLAB grafisch dar. (Dauer: ca. 30 Minuten)

Meß-abstand	5	10	20	30	40	50	70	100	150	200	250	300
Sensor-abstand												

- b) Ermitteln Sie, bis zu welchem Winkel verschiedene Objekte (z.B. Glasflasche, Karton) außerhalb der Zentraallinie des Sensors noch erkannt werden. Variieren Sie dazu den Winkel zwischen Sensor und Messobjekt bei konstantem Abstand zwischen den beiden und stellen Sie das Ergebnis grafisch dar. (Dauer: ca. 30 Minuten)
- c) Bauen Sie einen fahrbaren Roboter auf und montieren Sie den Ultraschallsensor daran. Dieser sollte durch einen Motor drehbar sein oder aufgrund einer 360°-Drehung des Roboters einen 360°-Scan der Umgebung ermöglichen (nötig für Aufgabenteile f und g). Zunächst soll die Messrichtung des Ultraschallsensors mit der Fahrrichtung des Roboters übereinstimmen. Steuern Sie den Fahrroboter mit Hilfe der MATLAB-Befehle so, dass er ungebremst auf eine Wand zufährt und die (synchrone) Motoren mit dem Kommando `motor.syncedStop()`; abschaltet, sobald der Abstand von der Wand 40cm oder weniger beträgt. (Dauer: ca. 90 Minuten)
- d) Bestimmen Sie den Bremsweg des Roboters wie unter c) als Funktion der Geschwindigkeit (Power). Messen Sie dazu bei verschiedenen Geschwindigkeiten den Weg, den der Roboter noch zurücklegt, wenn man die Motoren stoppt. Um genauere Werte zu erhalten, machen Sie zu jeder Geschwindigkeit 3 Versuche und bilden Sie den Mittelwert der Ergebnisse. Nutzen Sie dazu die MATLAB-Funktion für den Mittelwert. Prüfen Sie, ob sich ein einfacher mathematischer Zusammenhang zwischen der Fahrgeschwindigkeit und dem gemessenen Bremsweg ergibt. Nähern Sie die Funktion des Bremsweges durch ein Polynom 2. Grades an. Benutzen Sie dazu die Funktion `polyfit`. (Dauer: ca. 90 Minuten)

Power	20	30	40	50	60	70	80
Sensor-abstand							
Meß-abstand							

3 Durchführung

p2	p1	p0

- e) Ändern Sie die Steuerung des Roboters aus Punkt d) so, dass der Bremsweg berücksichtigt wird, um den gegebenen Abstand zur Wand genauer einzuhalten, indem der Roboter geschwindigkeitsabhängig früher bremst. (Dauer: ca. 30 Minuten)

Power	20	30	40	50	60	70	80
Sensor-abstand							

- f) Programmieren Sie den Roboter so, dass der Ultraschallsensor eine Drehung von 360° durchführt. Führen Sie dabei einen Scan der Umgebung durch, indem Sie abhängig vom Winkel des Schallsensors die gemessene Distanz aufzeichnen. Erstellen Sie auf diese Weise in MATLAB mit der Funktion **polar** ein Profil der Umgebung des Roboters. (Dauer: ca. 120 Minuten)
- g) Bauen Sie einen Raum aus Aktenordnern oder Brettern, der genau einen Ausgang hat. Stellen Sie den Roboter mit beliebiger Blickrichtung hinein. Programmieren Sie den Roboter so, dass er nach dem 360° -Scan anhand der Daten automatisch den Ausgang des Kartons findet und hinaus fährt. Zusatzaufgabe: Erweitern Sie das Programm so, dass der Roboter bei mehreren Ausgängen den breitesten Ausgang nimmt. (Dauer: ca. 120 Minuten)