HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Frau
Yulu Zhu

Erstellung von Beispiel-applikationen für einen mBot Ranger

Mittweida, WS2022/2023

Fakultät Ingenieurwissenschaften

BACHELORARBEIT

Erstellung von Beispiel-applikationen für einen mBot Ranger

Autor:
Frau

Yulu Zhu

Studiengang:
Mechatronik

Seminargruppe:
Me18wA-b

Erstprüfer:
Prof. Dr. ing. habil.A. Winkler

Zweitprüfer:
M. Sc.Christian Thormann

Einreichung:
Mittweida,03.04.2023

Verteidigung/Bewertung:
Mittweida, 2023

Faculty Engineering

BACHELORTHESIS

Developing of sample applications for an mBot Ranger

author:
Ms.

Yulu Zhu

course of studies:
Mechatronics

seminar group:
ME18wA-b

first examiner:
Prof. Dr. ing. habil.A. Winkler

second examiner:
M. Sc. Christian Thormann

submission:
Mittweida, 03.04.2023

defence/ evaluation:
Mittweida, 2023

Bibliografische Beschreibung:

Zhu, Yulu:
Erstellung von Beispiel-applikationen für einen mBot Ranger. - 2023. –70 Seiten, - 41 Seiten, S17,S12.
Mittweida, Hochschule Mittweida,Fakultät Ingenieurwissenschaften, Bachelorarbeit, 2023

Referat:

In meiner Bachelorarbeit habe ich mich mit der Erstellung von Beispielsapplikationen für eine mBot Ranger auseinandergesetzt. Das Ziel meiner Arbeit ist es, diese Beispielsanwendungen für mBot Ranger durch das Prüfen von Sensoren mit verschiedenen Aufgaben und die Programmierung von Software (mblock5 und Arduino) zu realisieren.
Das Endergebnis dieser Arbeit ist ein mBot Ranger, der mit Hilfe verschiedener Sensoren unterschiedliche Applikationen ausführen kann, wie z.B. Hindernissen ausweichen, einer Linie folgen, in die Richtung des lautesten Geräusches fahren, nach Norden zeigen, im Gleichgewicht stehen und einen Mindestabstand auf beiden Seiten einhalten.

Inhalt

Abbildungsverzeichnis

Abkürzungsverzeichnis

I2C	Inter-Integrated Circuit
IR	Infrarot-Led
LED	light-emitting diode
NTC	Negative Temperature Coefficient Thermistor
RGB	R-Rot, G-Grün und B-Blau
SCL	Serial Clock
SDA	Serial Data
USB	Universal Serial Bus

1 Motivation und Zielsetzung

1.1 Einführung

In der modernen Gesellschaft beeinflusst die Digitalisierung immer mehr Bereiche des täglichen Lebens und den gesellschaftlichen Wandel. Die digitale Revolution verändert nicht nur die Kommunikation und das Arbeitsleben, sondern auch die Industrie. Heutzutage werden gefährliche oder sehr monotone Aufgaben von Maschinen übernommen. In modernen Autoproduktionsanlagen haben Menschen nur Aufsichtsrechte. Roboter montieren die mBot präzise am Fließband. Es handelt sich um Maschinen, die von einer programmierbaren Steuerung kontrolliert werden und ihr Verhalten daher flexibel an ihre Umgebung anpassen können. Dies bezeichnet man im weitesten Sinne als Roboter. Die Roboter haben eine gemeinsame Struktur aus Sensorik und Aktorik. Es ist allgemein bekannt, dass Sensoren in vielen Ingenieurbereichen eine wichtige Rolle spielen und auch in der Robotik notwendig sind. Unter Sensoren versteht man, dass Roboter mithilfe von Sensoren ihre Umgebung wahrnehmen und ein Bild davon erstellen können. Dieses Bild besteht aus Messwerten. Bei der Aktorik ist es unterschiedlich. Hierbei wirken die Roboter durch Aktoren auf ihre Umgebung ein und führen Aktionen aus.

Am Anfang der Bachelorarbeit werden die Motivation und Zielsetzung vorgestellt. Dabei geht es um den Hintergrund der Roboter, die Aufgabenstellung sowie das Ziel dieser Bachelorarbeit. In Kapitel 2 wird der theoretische Rahmen der Arbeit dargelegt und betrachtet. Dieses Kapitel ist in zwei Teile unterteilt: Hardware und Software. Es wird der Aufbau des mBot Ranger, die grundlegende Theorie der Sensoren und die Software mBlock5 erläutert. In Kapitel 3 wird die Methodik der Untersuchung dargestellt. In diesem Teil sind die Erstellung von Beispielsapplikationen für den mBot Ranger und Lösungsverfahren vorgesehen. Im vierten Teil der Arbeit werden die Forschungsergebnisse präsentiert und dabei insbesondere auf die Punkte eingegangen, die hervorgehoben werden sollen. Zum Schluss folgt eine Zusammenfassung und ein Ausblick.

1.2 Aufgabenstellung

Die vorliegende Arbeit beschreibt die Durchführung von Applikationen für den mBot Ranger, um die verfügbaren Möglichkeiten von Sensorik und Aktorik deutlich aufzuzeigen. Dabei werden die unterschiedlichen Funktionsweisen der Sensoren mit Hilfe des mBot Ranger und der Software mBlock5 getestet. Die Hauptaufgabe besteht darin, sich auf den mBot Ranger zu konzentrieren, wenn die programmierten Befehle vom Computer in den mBot Ranger eingegeben wird. Der Roboter folgt dann den Anweisungen und führt die Befehle aus. Der mBot Ranger ist ein Anfängerroboter, der aufgrund seiner preisgünstigen, programmierbaren und seiner Anbindung an das Arduino-Universum viele Ziele einfach realisieren kann.

Als Teil dieser Aufgaben ist es notwendig,alle verschiedenen Aspekten der mBot-Technologie zu beherrschen, einschließlich der Einarbeitung in die mBot-Plattform und das mBlock-Programmierwerkzeug.Außerdem werden die verschiedenen Sensoren dokumentiert und Applikationen erstellt, die spezielle Aufgaben erfüllen, wie zum Beispiel die Ausweichung von Hindernissen, dem Folgerung von Linien und die Bewegung in Richtung der größten Lautstärke. Darüber hinaus werde die Anwendungen getestet, um sicherzustellen, dass sie ordnungsgemäß funktionieren.

Das Ziel dieser Arbeit ist es, verschiedene Funktionen zu implementieren, um den mBot effektiv und präzise arbeiten zu lassen. Einige der Ziele sind die Fähigkeit, Hindernisse zu erkennen und ihnen auszuweichen, der Fähigkeit, Linien zu folgen und den Mindestabstand zu halten. Die Temperatur wird auch mit einer 7-Segment-Anzeige angezeigt, und der Kompass wird in Richtung Norden ausgerichtet. Schließlich wird auch das Gleichgewicht des mBot mit einem Gyroskop sichergestellt, um eine stabile Bewegung zu ermöglichen.

2 Stand der Technik und Wissenschaft

2.1 Hardware

2.1.1 Aufbau von mBot Ranger

2.1.1.1 Vorstellung von mBot Ranger

Bevor der mBot Ranger vorgestellt wird, soll im Folgenden eine kurze Beschreibung des mBot gegeben werden.Der mBot ist ein Basisroboter und das erste Produkt der Firma Makeblock. Er wurde vom chinesischen Flugzeugkonstrukteur Jason Wang entwickelt und ist wahrscheinlich das Modell, mit dem die meisten Menschen in die faszinierende Welt der Robotik einsteigen werden.

Der mBot Ranger ist eine erweiterte Version des mBot. Bevor dieser Roboter zum Leben erweckt werden kann, muss er aus mehreren Komponenten zusammengesetzt werden. Der Zusammenbau des mBot Ranger ist jedoch auch ohne Vorkenntnisse möglich. Der mBot Ranger besteht aus fünf grundlegenden Bauteilen: einem Kettenfahrzeug mit sechs Rädern, einem Aluminium-Chassis, dem Me Auriga Board, einem Batterieplatz und zwei Motoren.

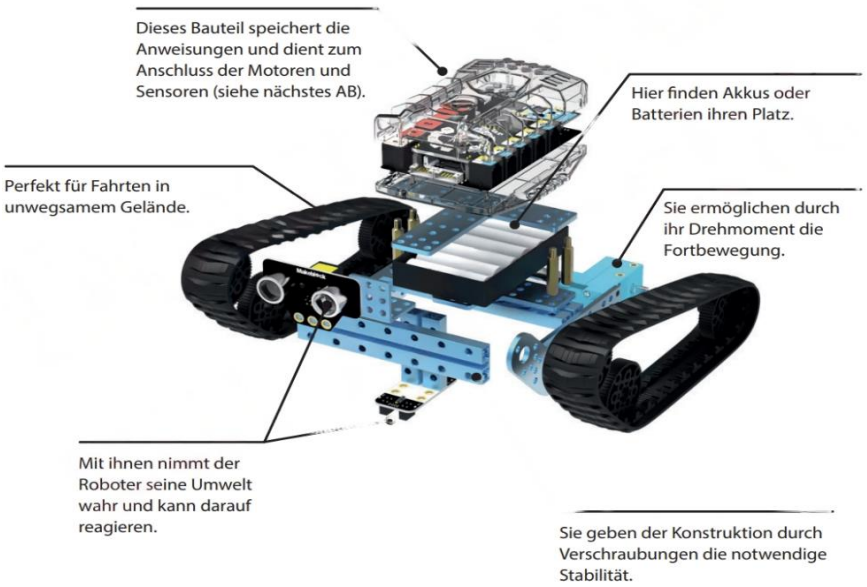


Abbildung 1: Aufbau von mBot ranger(Quelle[mBot ranger Bedienungsanleitung])

Das Me Auriga board ist das Gehirn von mBot Ranger. Es ist ein Mikrocontroller-Board, das auf der Arduino-Plattform basiert. Es verfügt über eine Vielzahl von Ein- und Ausgängen, die es ermöglichen, verschiedene Sensoren und Aktoren anzuschließen. Es verfügt auch über eine Bluetooth-Schnittstelle, die eine kabellose Verbindung zum Computer oder Smartphone ermöglicht, um mBot Ranger zu programmieren oder Daten auszutauschen. Die Batterie, die unter der Me Auriga board angebracht ist, liefert die benötigte Energie für den Betrieb von mBot Ranger. Es ist wichtig, eine leistungsstarke Batterie zu wählen, um eine ausreichende Laufzeit und Leistung zu gewährleisten. Die beiden Motoren auf der Unterseite des Chassis sind die Antriebsmotoren von mBot Ranger. Sie können unabhängig voneinander gesteuert werden, um verschiedene Bewegungen und Richtungswechsel zu ermöglichen. Durch die Steuerung der Motoren können wir mBot Ranger in jede gewünschte Richtung bewegen und ihn Hindernissen ausweichen lassen. Zusammen bilden diese Komponenten den mBot Ranger, einen vielseitigen und programmierbaren Roboter, der für Anfänger und Fortgeschrittene gleichermaßen geeignet ist.

Darüber hinaus werden die Komponenten über einfache mechanische und elektrische Verbindungen zusammengefügt. Die elektrischen Verbindungen werden über ein ausgeklügeltes farbcodiertes Stecker-Buchsen-System hergestellt. Für die mechanischen Verbindungen werden die Stützkonstruktionen aus robusten Aluminiumprofilen unterschiedlicher Länge aufgebaut und bieten Halt für die Aufbauten, die ich am mBot Ranger befestigen möchte. In der Regel werden Schrauben, Muttern, Schraubenschlüssel und ein kleiner Multifunktionsschraubendreher benötigt. Wenn man den mBot Ranger auf der Seite aufgebaut hat, kann man den mBot Ranger programmieren. In der Programmiersprache Scratch können die einzelnen Befehle mit der Maus aus einem Vorrat ziehen und damit Programmabläufe zusammenstellen. Scratch gehört zu mblock5. Die einzelnen Befehle werden Blöcke genannt, die von oben nach unten abgearbeitet werden und wie Puzzleteile zusammenpassen. Die Codeblöcke werden zusammengefügt und nacheinander von mBot Ranger ausgeführt.

2.1.1.2 Me Augrina board

Me Augrina-Board ist ein Microcontroller, der als Steuerzentrale für den mBot Ranger dient und bereits mit mehreren Sensoren (Temperatur, Gyroskop, Schall) ausgestattet ist. Als Aktuator empfängt er Befehle und führt Aktionen aus. Er bietet mehrere Anschlüsse (RJ25/Ports) für Motoren und Sensoren und verfügt über eine USB-Schnittstelle (zur Verbindung mit dem PC) sowie einen 12er-LEDRing. Der Microcontroller ist in einem transparenten Kunststoffgehäuse verbaut und wird über einen Stromanschluss mit eingebautem Akku betrieben. Das Programm wird über ein USB-Kabel auf den Mikrocontroller hochgeladen. Die Daten der Sensoren werden über RJ-25 Buchsen aufgenommen und als Messwerte ausgegeben, wie in Abbildung 2 dargestellt. Auf der Oberseite befindet sich ein Starterknopf, der zum Ein- und Ausschalten des Roboters dient.

Auf der Rückseite befinden sich zwei Tasten in unterschiedlichen Farben: rot und schwarz. Die Rot Tatste ist der Stromschalter und die Schwartz Taste ist Reset. Die Serie von Microcontroller-Boards verwendet den ATmega2560.

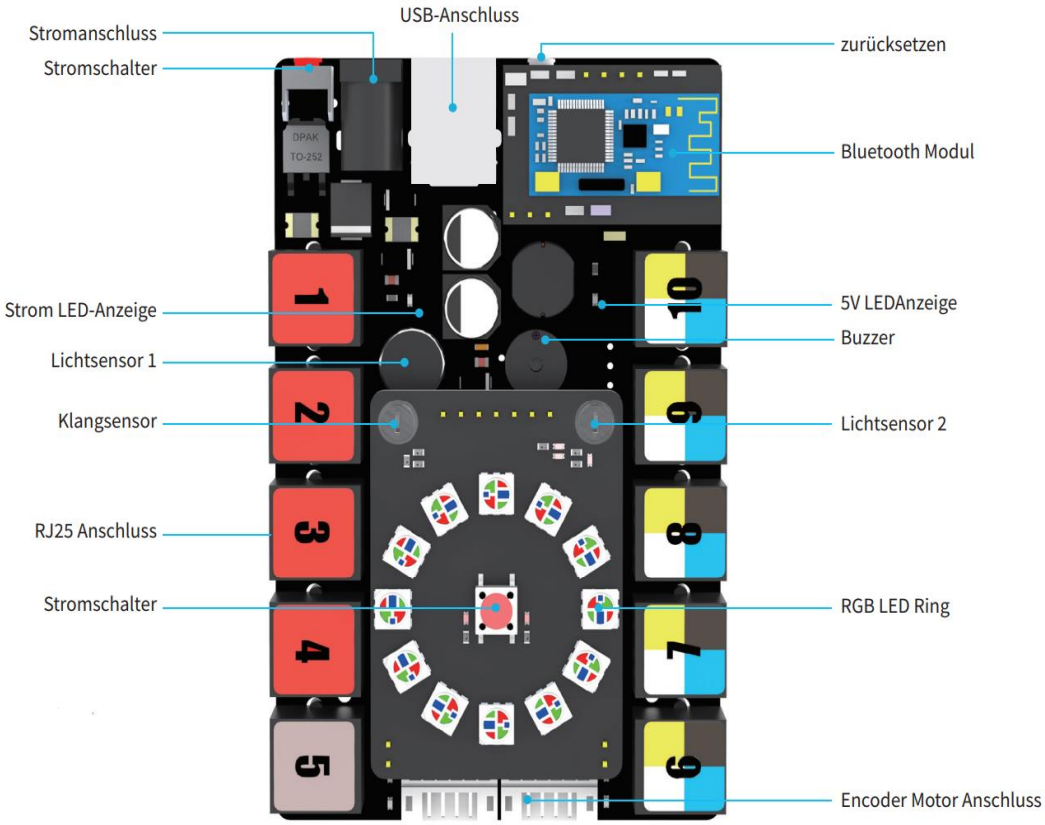


Abbildung 2: Me Augrina board(Quelle[mBot ranger Bedienungsanleitung])

2.1.1.3 Klangsensor (Schallsensor)

In Abbildung 3 oben auf dem ME Augrina Board ist ein kleiner schwarzer Kreis zu sehen - das ist ein Klangsensor. Der Me Sound Sensor basiert auf einem Mikrofon und kann dazu verwendet werden, Umgebungsgeräusche zu erfassen. Durch den Einsatz des LM386

Leistungsverstärkers und des Elektromikrofons ist der Schallsensor in der Lage, analoge Werte im Bereich von 0 bis 1023 auszugeben.

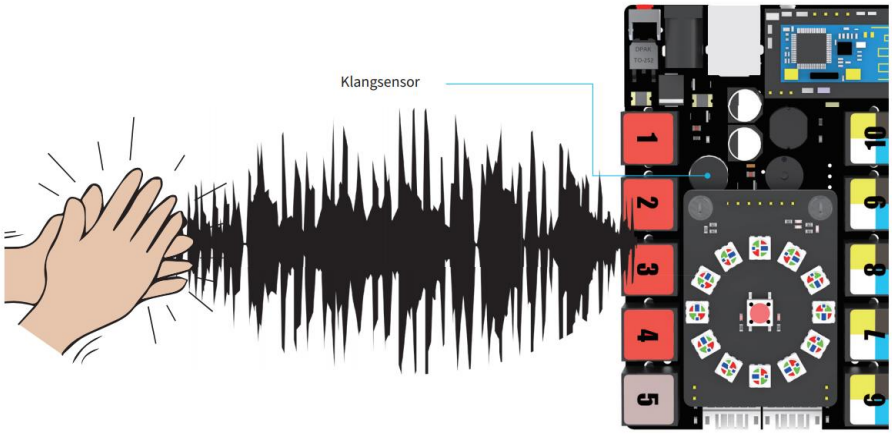


Abbildung 3:Schallsensor(Quelle[mBot ranger Bedienungsanleitung])

2.1.1.4 Licht Sensor

Die Lichtsensoren befinden sich auf dem Augrina Board. In Abbildung 4 kann man deutlich erkennen, dass es zwei Lichtsensoren gibt. Der Lichtsensor erfasst die Intensität des Lichts. Er liefert Werte im Bereich von 0 bis 1023 zurück, wobei 0 Dunkelheit und 1023 die maximal mögliche Helligkeit darstellt, die der Sensor verarbeiten kann. Der Lichtsensor ist ein analoger Sensor. Die Helligkeit ist eine Größe, die keine natürliche Unterteilung hat und beliebige Zwischenwerte annehmen kann. Analoge Sensoren haben die Eigenschaft, dass sich ihre Eingangswerte kontinuierlich verändern können. Viele Messwerte sind eigentlich analoger Natur, so z. B. die Temperatur oder die Geschwindigkeit, mit der sich der Roboter bewegt. Analogwerte lassen sich sehr schön mit einem Zeigerinstrument darstellen. Die Auslenkung des Zeigers auf einer Skala kann sich in beliebig feinen Schritten verändern.

Ein Mikrocontroller kann mit analogem Input jedoch von Haus aus wenig anfangen. Er kann nur die digitalen Zustände 0 und 1 verarbeiten. Das Signal muss daher für den mBot umgewandelt werden und eine Zahl aus dem Helligkeitswert gemacht werden. Hierbei kommt ein sogenannter Analog-Digital-Wandler zum Einsatz. Dieser ist bereits im Helligkeitssensor integriert, sodass man sich um die konkrete Schaltung keine Gedanken machen müssen. Der Wandler wandelt die kontinuierliche Größe Helligkeit in eine Zahl zwischen 0 und 1023 um, die man auslesen und weiterverarbeiten kann. In der Praxis von Robotern ist diese Umwandlung ein wichtiger Schritt, da sehr viele externe Messwerte analoger Natur sind und zuerst umgewandelt werden müssen.

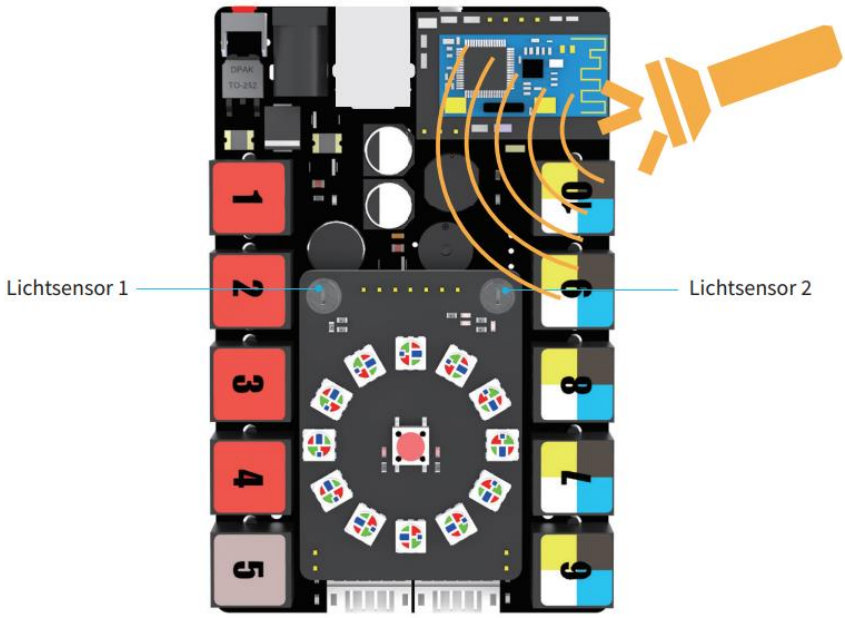


Abbildung 4: Licht sensor(Quelle[mBot ranger Bedienungsanleitung])

2.1.1.5 Me Ultraschallsensor

Der Ultraschallsensor ist in Abbildung 5 zu sehen. Die Größe des Me-Ultraschallsensors beträgt 56*36*31mm. Er wird mit Audinuo C zur einfachen Programmierung geliefert. Der Sensor wird über eine 6-Pin RJ25 Schnittstelle angeschlossen und ist mit einem gelben Tag gekennzeichnet. Er ist kompatibel mit dem Me Base Shield und dem Me Baseboard. Zur Befestigung gibt es drei M4-Bohrungen, die mit Makeblock Balken kompatibel sind. Auf der Platine sind LEDs zur Anzeige von Debugging-Informationen und Feedback vom ME Augrina Board montiert.

Abbildung 5: Ultraschallsensor(Quelle[mBot ranger Bedienungsanleitung])

Als nächstes werden die grundlegende Theorie und Funktionsweise des Ultraschallsensors beschrieben. Als Abstandssensor ist der Ultraschallsensor in der Lage, Objekte berührungslos zu erkennen und deren Abstand zum Sensor zu messen. Beim Sendevorgang sendet der Ultraschallsensor zyklisch einen kurzen, hochfrequenten Schallimpuls aus. Dieser breitet sich in der Luft mit Schallgeschwindigkeit aus. Trifft der Ultraschallimpuls auf ein Objekt, wird er reflektiert. Das entstehende Echo wird vom Sensor wieder aufgenommen und aus der Zeitspanne zwischen Aussendung und Empfang des Schallimpulses wird die Entfernung zum Objekt berechnet (siehe Abb.6 und Abb.7). Der Ultraschallsensor kann Objekte aus verschiedenen Materialien wie Metall, Holz oder Kunststoff detektieren. Nur schallabsorbierende Materialien wie Watte , schräge Oberflächen können vom Ultraschallsensor schlecht erfasst werden.

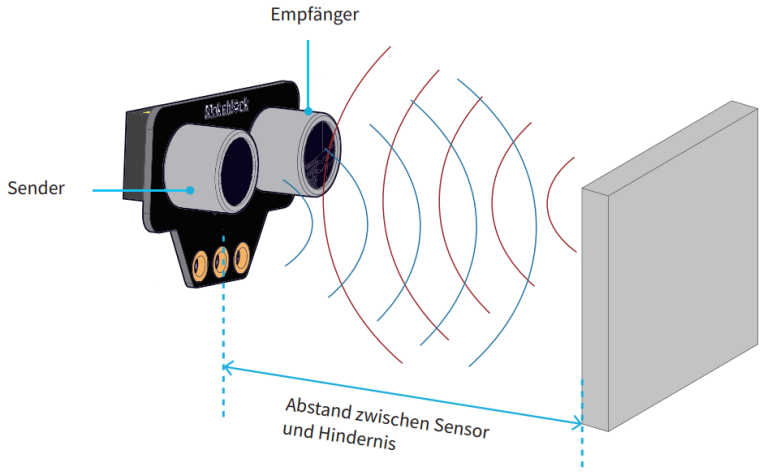


Abbildung 6: Funktionsweise 1(Quelle[mBot ranger Bedienungsanleitung])

Die Reichweite des Ultraschallsensors liegt zwischen 3 cm und 400 cm, und der Öffnungswinkel beträgt 30 Grad (Ultraschallsensoren senden Schallimpulse innerhalb eines bestimmten Öffnungswinkels aus). Die Entfernung zu Objekten außerhalb dieses Schallkegels kann nicht bestimmt werden. Er wird mit 5V an Gleichspannung betrieben. Außerhalb der Winkel- und Abstandsbereiche erkennt der Me-Ultraschallsensor die Hindernisse und Objekte nicht.

Alles in allem haben die Ultraschallsensoren zahlreiche Anwendungen, wie z.B. Einparkhilfe-Sensoren in Autos und Annäherungsalarme, Hinderniserkennung und Positionierung.

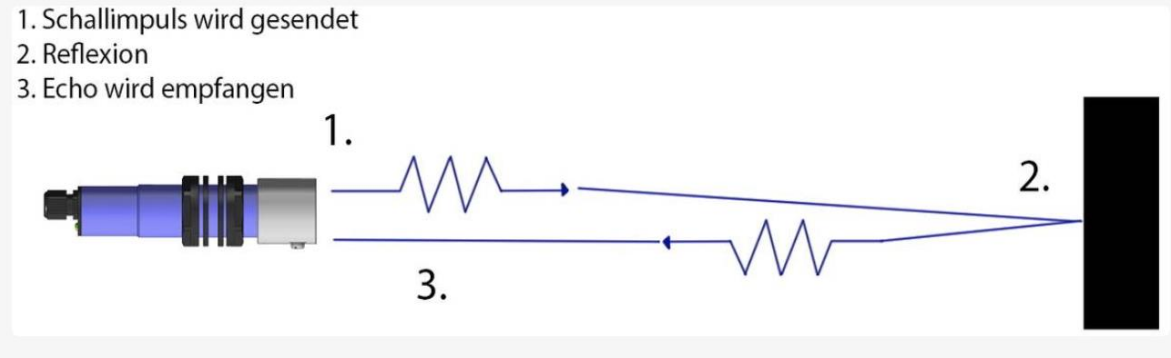


Abbildung 7: Funktionsweise 2 (Quelle [Waycon Unternehmen])

2.1.1.6 Linienfolgers-sensor(IR-Sensor)

Der Line Folger Sensor wird auch als IR-Sensor bezeichnet, der mit dem Me Base Shield kompatibel ist. Die Größe des Me Line Follower Sensors ist 48*24*24 cm. Er verfügt über zwei 2,54-mm-Jumperkabelanschlüsse und drei M4-Montagebohrungen im Abstand von 16 mm, die mit Makeblock-Halterungen kompatibel sind. Dieser Sensor ist mit drei LEDs ausgestattet. Eine der LEDs zeigt die Stromversorgung an, die anderen beiden LEDs zeigen das Signal des Sensors an.

Abbildung 8: Linienfolgers-sensor(Quelle[mBot ranger Bedienungsanleitung])

Die Abbildung 9 zeigt ,dass Me Linienfolge Sensor ist eine wensentlichen IR-Sensor ,Es besteht aus Zwei Teile : IR-LED sendet als Quelle die Licht aus und der Empfänger (IR Photodiode/phototransistor) detektiert die Intensität des reflektierten Lichtes. Der Anteil des reflektierten Lichts hängt nicht nur vom Abstand zwischen Sensor und Oberfläche ab, sondern auch von der Oberflächenbeschaffenheit, der Farbe und der Ausrichtung der Oberfläche. Das Grundprinzip des Linienfolgesensors besteht darin, dass er das Licht

reflektiert, das vom Hintergrund zurückgeworfen wird, und diese Messwerte an den mBot zur Anzeige in Infrarot zurücksendet. Zum Beispiel: je heller der Untergrund, desto größer der Messwert. Wenn der IR-Sensor des mBot Ranger sich im Bereich des weißen Untergrunds befindet, wird die Photodiode immer beleuchtet. Anhand dieser Messwerte muss der Roboter entscheiden, wo sich die Kante der Linie befindet und in welche Richtung er fahren muss. Der Linienfolgesensor wird in einem Bereich von 1 bis 2 cm detektiert und wird mit einer Spannung von 5V versorgt. Beim mBot Ranger erkennt der Linienfolgesensor die Linien und fährt entlang dieser Linien.

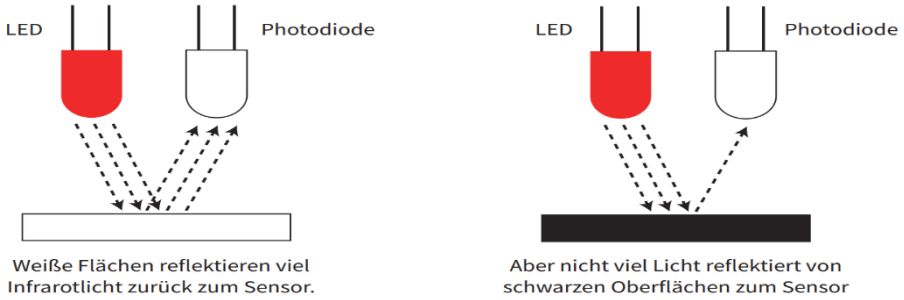


Abbildung 9: Prinzip von Led und Photodiode(Quelle[mBot ranger Bedienungsanleitung])

2.1.1.7 Gyroskopsensor

Der Gyroskopsensor befindet sich direkt auf dem Auriga-Board (siehe Abb.10) und enthält einen 3-Achsen-Beschleunigungssensor, einen 3-Achsen-Winkelgeschwindigkeitssensor sowie einen Bewegungsprozessor. Er bietet einen I2C-Anschluss für die Kommunikation und seine weiße ID zeigt an, dass er sich im I2C-Kommunikationsmodus befindet und an den Anschluss mit der weißen ID am Makeblock Auriga angeschlossen werden sollte.

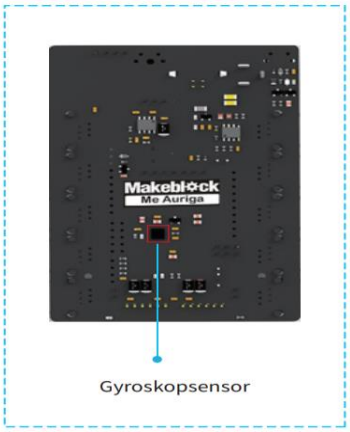


Abbildung 10: Gyroskopsensor(Quelle[mBot ranger Bedienungsanleitung])

Der 3-Achsen-Beschleunigungsmesser und -Kreiselsensor von Me Gyroskopsensor ist ein ideales Modul für die Erkennung von Roboterbewegungen und Körperhaltungen. Er kann in selbstbalancierenden Wagen, 4-Achsen-Flugzeugen, Robotern und mobilen Geräten eingesetzt werden und bietet einen hochdynamischen Messbereich und einen geringen Stromverbrauch.

Der Me Gyroscope-Sensor verfügt über zahlreiche Funktionen. Der 3-AchsenWinkelgeschwindigkeitssensor unterstützt einen Messbereich von ±250, ±500, ±1000 und ±2000°/s (dps). Er ist für Benutzer aller Altersgruppen geeignet und kann einfach über den Blau Anschluss angeschlossen werden. Die meisten Entwicklungsboards, einschließlich Arduino, werden von diesem Steckertyp unterstützt. Die Programmierung des Sensors ist über die mBlock GUI möglich

2.1.1.8 Temperatursensor

Der mobt Ranger verfügt über einen eingebauten Temperatursensor, der ein winziges Thermometer (einen NTC-Thermistor) enthält und die Umgebungstemperatur erfasst. Dieser Sensor ist direkt auf dem Makeblock Augrina Board installiert, wie in der Abbildung dargestellt. Die Temperaturänderungen werden durch eine Änderung des Widerstands angezeigt. Mit dem Makeblock Temperatursensor kann man Temperaturen im Bereich von -40°C bis 125°C messen(siehe Abb.12). Die wasserdichte Abschirmung des Sensors, bestehend aus einer Edelstahl- und PVC-Kunststoffummantelung, ermöglicht Messungen auch in säure- und laugenhaltigen Flüssigkeiten. [8](siehe Abb.11).

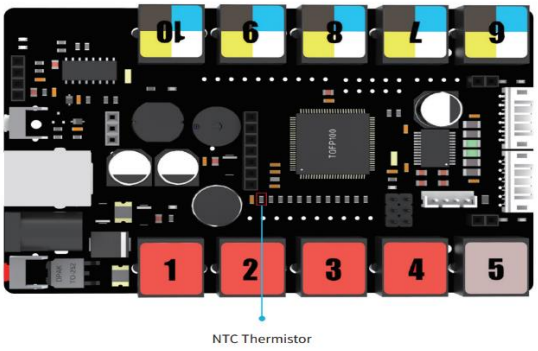


Abbildung 11: Temperatursensor (Quelle[mBot ranger Bedienungsanleitung])

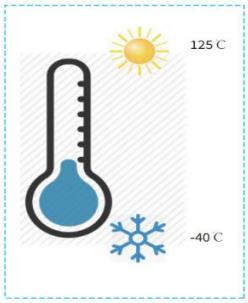


Abbildung 12: Messungsbereich von Temperatur(Quelle[mBot ranger Bedienungsanleitung])

2.1.1.9 Kompasssensor

Ein elektronischer Kompass, auch digitaler Kompass genannt, verwendet dagegen Sensoren, um die Ausrichtung des Moduls im Raum zu erkennen und daraus die Himmelsrichtung zu bestimmen. Ein solches Modul ist der Me-Kompasssensor von Makeblock, der einen 3-Achsen-Magnetometer enthält. Dieser Sensor ist in der Lage, das Erdmagnetfeld in allen drei Dimensionen zu messen und so die Ausrichtung des Sensors im Raum zu bestimmen. Durch die Verwendung von Kalibrierungstechniken kann der Me-Kompasssensor eine sehr genaue Ausrichtung und damit eine genaue Bestimmung der Himmelsrichtung erreichen. Der Sensor kann über einen RJ25-Anschluss(weiß-Anschluss) mit dem Makeblock-System verbunden werden und unterstützt die mBlock-GUI-Programmierung. Als Nächstes wird die Struktur des Kompasses und seine Funktionsweise eingeführt. Der Sensorchip auf dem Kompassmodul ist ein HMC5883L von Honeywell. Für eine präzise Messung benötigt der Sensor eine stabile und konstante Versorgungsspannung. Deshalb ist auf dem Kompassmodul eine eigene Spannungsregelung verbaut. Die Kommunikation mit dem Modul erfolgt über den I²C-Bus, für den lediglich zwei Anschlüsse (SDA und SCL) sowie eine Spannungsversorgung erforderlich sind. Da die Kommunikation über den I²C-Bus komplex sein kann, wurde die Abfrage über eine Bibliothek realisiert. Der Kompasssensor kann nicht nur die Ausrichtung der Himmelsrichtung messen, sondern auch die Ausrichtung in den drei Dimensionen x, y und z. In der aktuellen Version von mBlock ist jedoch nur die Ausrichtung der Himmelsrichtung verfügbar. Die offizielle Produktbeschreibung lautet: „3-Achsen-Digitalmagnetometer für den Einsatz in schwachen Magnetfeldern“. Der Sensor kann also sehr schwache magnetische Felder messen.

Abbildung 13: Kompass Sensensor(Quelle[mBot ranger Bedienungsanleitung])

2.2 Software

2.2.1 MBlock 5

Mblock5 wird am häufigsten mit dem mBot Ranger verwendet. Es enthält viele Bibliotheken, die mit Blöcken verbunden sind. Bevor man die Blöcke verwendet, muss man den mBot Ranger vorbereiten. Um den mBot Ranger erfolgreich anzusteuern, muss man die Scratch-Entwicklungsumgebung um eine Erweiterung (Extension) erweitern. Außerdem muss man eine entsprechende Software (Arduino C) auf den mBot Ranger oder genauer gesagt auf das mCore-Board laden. Diese Software übernimmt die Kommunikation mit der Scratch-Entwicklungsumgebung und sorgt dafür, dass der mBot Ranger unsere Befehle versteht. Im Gegensatz zu normaler Softwareentwicklung mit C++ ist es nicht erforderlich, eine einzige Zeile Code zu schreiben, um ein Programm zu erstellen. Stattdessen liegen alle Anweisungen in Form von vordefinierten Blöcken vor, die wir verwenden können. Anstatt Codezeilen zu tippen, können wir die Blöcke mit der Maus auf dem Bildschirm zusammenfügen. Die Funktionalität der einzelnen Blöcke ist selbsterklärend. Die verschiedenen Blöcke passen über Nasen und Kerben ineinander wie Puzzleteile, sodass man sofort sehen kann, wie die Blöcke zusammenarbeiten können.

Anhand des folgenden Beispiels (siehe Abbildung 14) kann man deutlich sehen, wie die Blöcke zusammengesetzt sind und wie der Code automatisch erstellt wird.

Der Ablauf von mblock 5 besteht aus den folgenden Schritten: Zuerst wird der mBot Ranger mit dem USB-Anschluss verbunden, dann werden die Blöcke ausgewählt und verbunden. Dann wird der Code automatisch erstellt und auf den mBot Ranger hochgeladen.

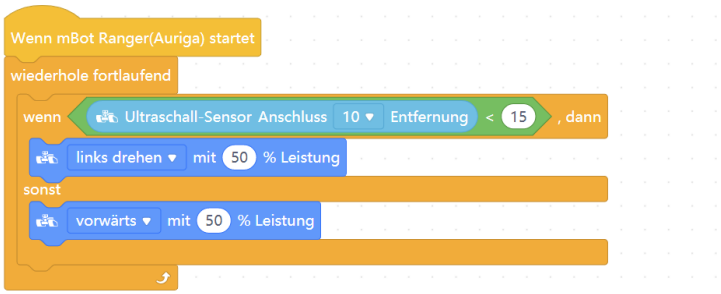


Abbildung 14:ein Beispiel von mblock(Quelle[Ausweichung von Hindernisse ,mblock5])

2.2.2 Arduino IDE

Es gibt verschiedene Möglichkeiten, um den mBot zu programmieren oder mit ihm zu interagieren, wobei jede Methode ihre Vor- und Nachteile hat. Neben der Verwendung von mBlock 5 kann auch Arduino zum Einsatz kommen. Der mBot Ranger kann eigenständig arbeiten, ohne dass eine Verbindung zu einem Computer erforderlich ist. Arduino IDE ist eine Open-Source-Software-Entwicklungsumgebung, die es Benutzern ermöglicht, Programme für Arduino-Mikrocontroller-Boards zu schreiben und hochzuladen. Es ist eine benutzerfreundliche Plattform, die auf C++ basiert und es Benutzern ermöglicht, schnell und einfach auf die Hardware zuzugreifen und zu programmieren.

Wenn man das Programm direkt in die Arduino-Software schreibt und es auf den mBot Ranger hochlädt (im sogenannten Arduino-Modus), kann man unabhängig von der Entwicklungsumgebung mBlock arbeiten. Das unteren genannte Beispiel(siehe Abb.15) zeigt die Verwendung der Arduino-Software.

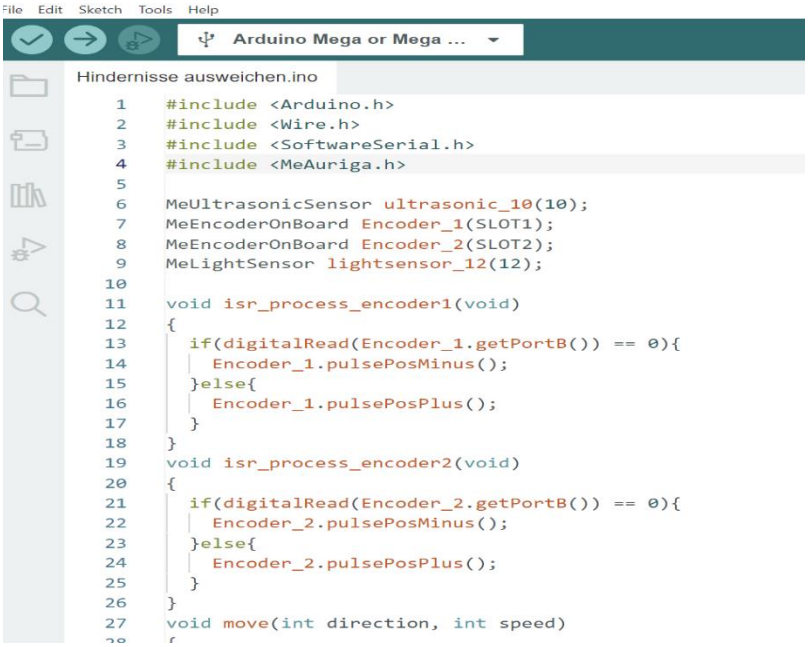


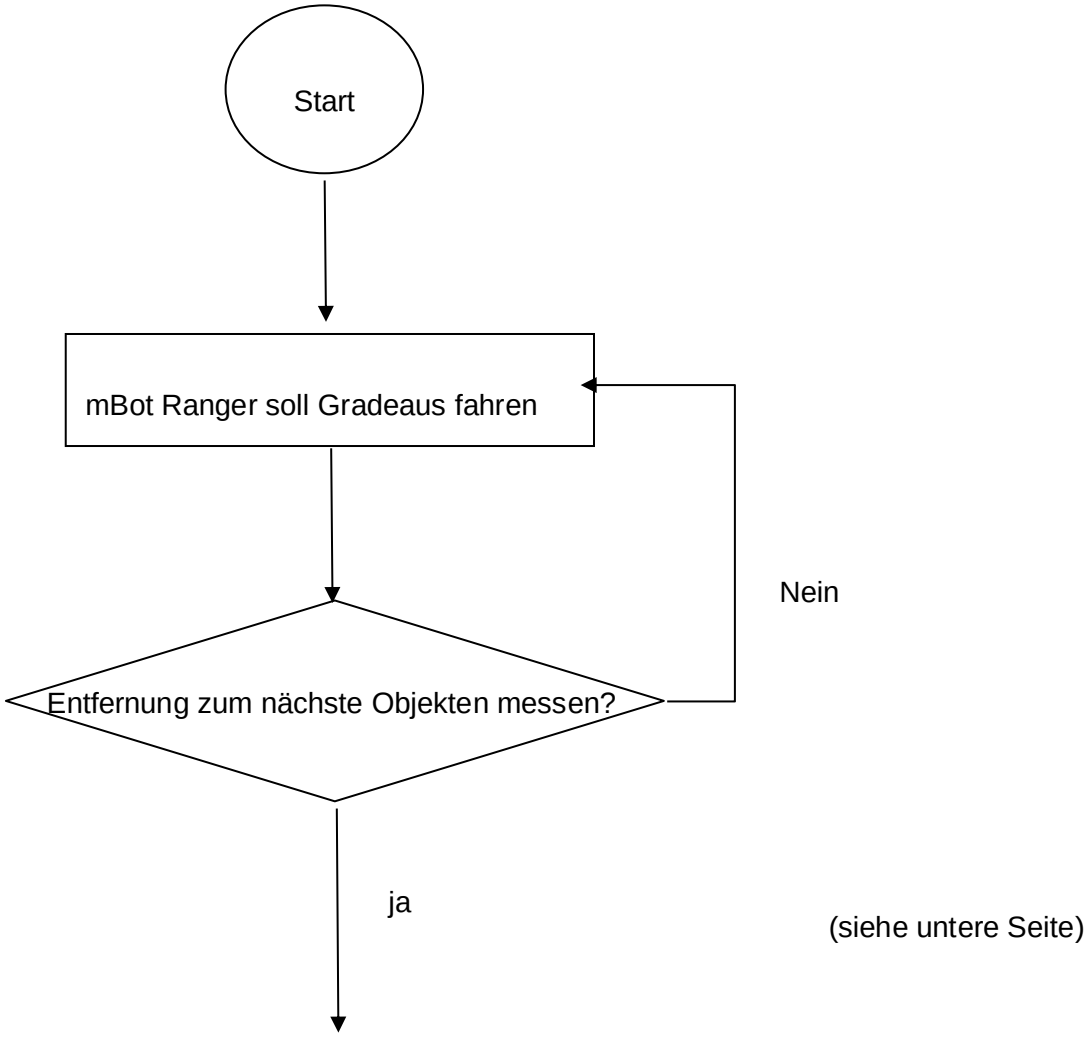
Abbildung 15:ein Beispiel von Arduino(Quelle[Ausweichung von Hindernisse, Arduino])

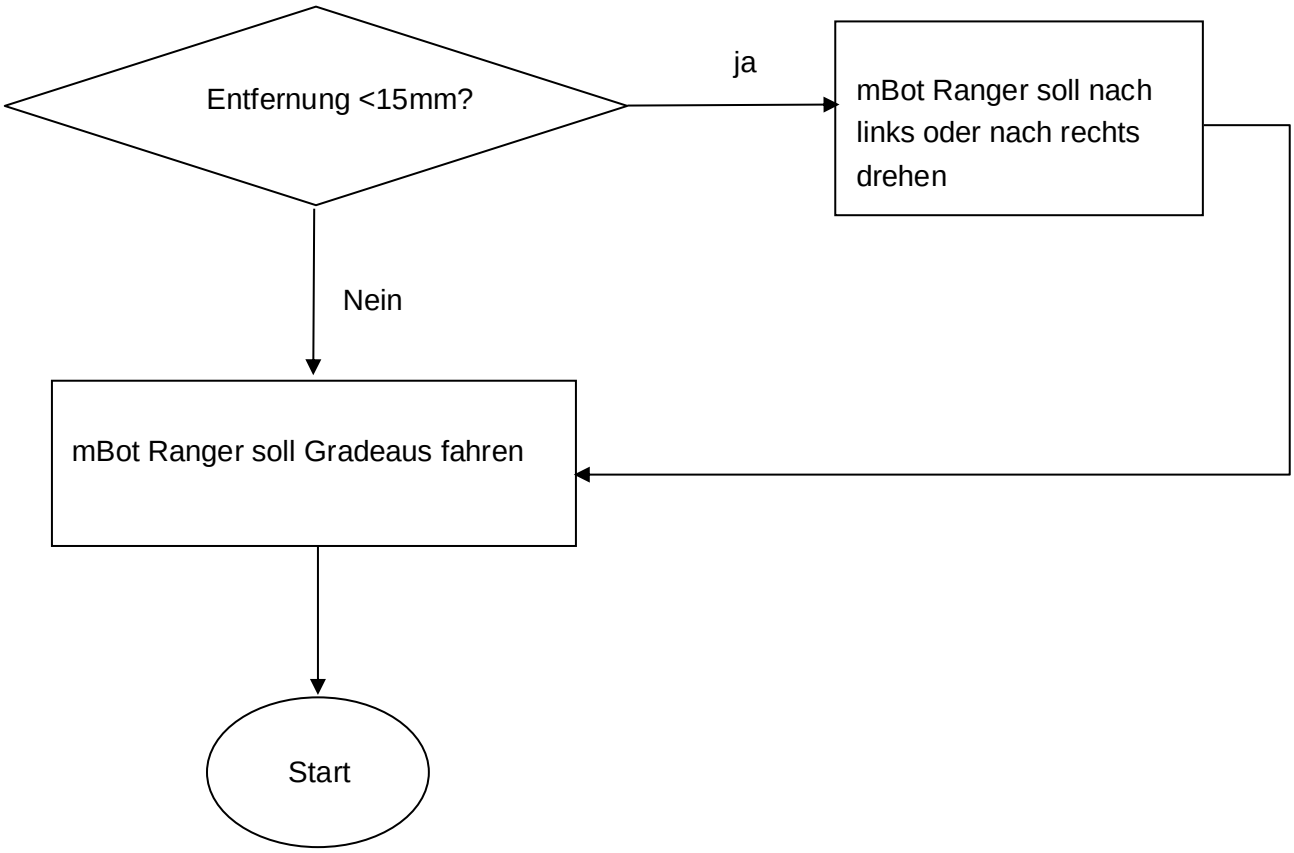
3 Applikationen mit Lösungsverfahren

3.1 Ausweichung von Hindernisse

Der mBot Ranger soll in der Lage sein, Hindernisse eigenständig zu erkennen und ihnen auszuweichen, indem er einen Front-Ultraschallsensor verwendet, der die Entfernung zum vor ihm liegenden Objekt misst. Um dieses Problem zu lösen, gibt es folgende Idee: Der Ultraschallsensor erkennt Objekte durch Ultraschall und der Abstand, bei dem eine Detektion erfolgt, wird auf 15mm eingestellt. Bevor die Blöcke erstellt werden, wird ein Algorithmus entwickelt, der die Daten des Ultraschallsensors in dieser Anwendung nutzt, um die gesamte Steuerung des mBot Ranger zu übernehmen. Der Algorithmus wird in ein Programm integriert, sodass der Abstand zu möglichen Hindernissen kontinuierlich mithilfe des Ultraschallsensors gemessen wird und der mBot Ranger versucht, das Hindernis zu umfahren, wenn ein Grenzwert unterschritten wird. Anschließend wird der Algorithmus skizziert und die Blöcke werden entsprechend einfach erstellt.

Hier ist eine Grafik von Algorithmus:





Nach dem obere Grafik werden die Blöcke erstellt. Die Blöcke wird wie in der Abbildung.16 dargestellt.

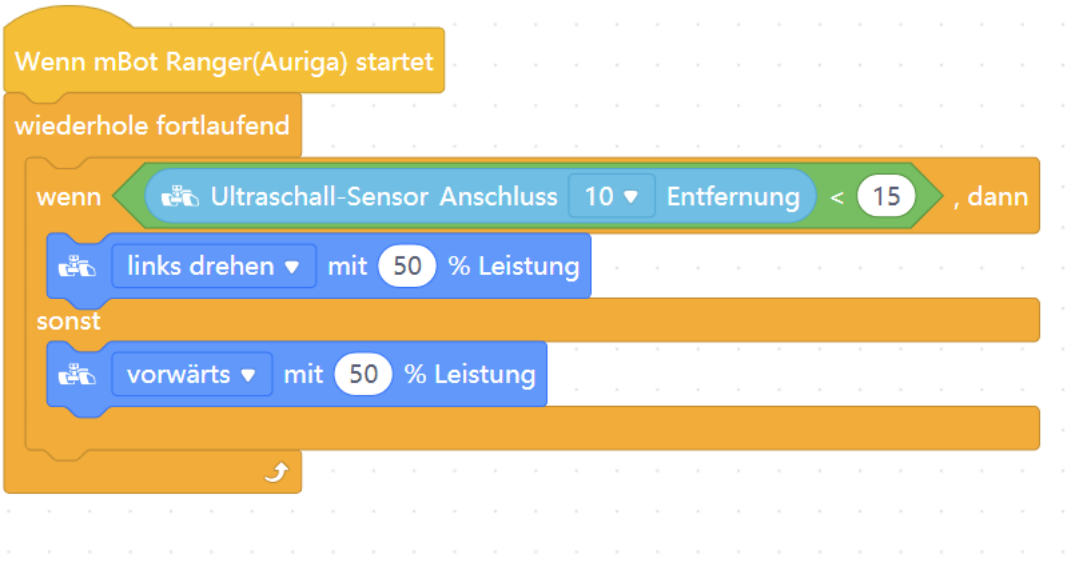


Abbildung 16: Coldeblöcke 1(Quelle[Ausweichung von Hindernisse, mblock5])

Es gibt immer noch ein Problem: Wenn der mBot Ranger Hindernisse durch den vorderen Ultraschallsensor erkennt, fährt er immer nach links. Die Richtung des mBot Rangers ist festgelegt. Deshalbe soll der mBot Ranger zufällig nach links oder nach rechts drehen , wenn der Roborter jedes mal die Hindernisse ausweicht. Es ist nicht genug für mblock5, um diese Aufforderung zu erfüllen , damit kommt es zur Anwendung von Arduino IDE.

Bei der Codierung von Aurdino werden die wichtigsten Teile dorthin verlagert.

Code:

```
1.      void move(int direction, int speed)
2.{
3.  int leftSpeed = 0;
4.  int rightSpeed = 0; //intialisierung von Speed
5.  if(direction == 1)
6.  {
7.    leftSpeed = -speed;
8.    rightSpeed = speed;
9.  } // Die Richtung is Vorwarts
10.   else if(direction == 2)
11.   {
12.     leftSpeed = speed;
13.     rightSpeed = -speed;
14.   } //Die Richtung is Rückwärts
15.   else if(direction == 3)
16.   {
17.     leftSpeed = -speed;
18.     rightSpeed = -speed;
19.   }//Die Richtung is nach Links
20.   else if(direction == 4)
21.   {
22.     leftSpeed = speed;
23.     rightSpeed = speed;
24.   } //Die Richtung is nach Rechts
```

```
1.    if (cm1 < 15 && flag == 0) {
2.      flag = 1;// Das Objekt befindet sich auf der Vorderseite des mBot Ranger
3.      angle_p = angle;// der aktuellen Winkel
4.      rand = random(3, 5);// zufällig nach links und rechts
5.} //  Detektierungsbereich einstellen
6.if (flag == 1) {
7.      flag2 = rand;
8.      move(rand, 50 / 100.0 * 255);
9.      if (abs(angle_p - angle) >= 90) flag = 0;
10.    } // wenn ult1 vorne eine Objekt detektieren, werde der Roboter zufällig
   nach rechts oder Links um 90 grad fahren.
```

Dieser Code verwendet eine Zufallsfunktion und bestimmt den Winkel des Gyroskopsensors, um den mBot Ranger um 90 Grad zufällig nach links und rechts abzulenken.

3.2 Der mBot ranger folgt einer Linie

Der mBot Ranger verfügt über einen eingebauten Linien-Finder, der in der Lage ist, den Untergrund nach hellen und dunklen Flächen zu erkennen. Der Roboter folgt mithilfe des Linien-Folgersensors einer vorgegebenen Spur, die auf ein Stück Papier aufgezeichnet wird. Die vorgegebene Spur ist als eine ovale schwarze Form definiert. Wenn der mBot Ranger der schwarzen Linie folgt, leuchten die beiden LEDs am unteren Sensor je nach Helligkeit des Papiers blau auf. Die Funktionsweise des Linien-Folgersensors besteht darin, dass die Sensoren bei einem schwarzen Untergrund erlöschen und bei einem weißen Untergrund leuchten. Je nach Abweichung von der Spur kann es zu unterschiedlichen Kombinationen kommen, die in Richtungsänderungen während der Fahrt umgesetzt werden können. Es gibt insgesamt vier Möglichkeiten der LED-Beleuchtung. Vor der Erstellung des Programms sollte die Funktionsweise des Linien-Folgersensors genau verstanden werden.

1. Möglichkeit: Genau in der Spur. Die beiden Led leuchten nicht nach dem Abb.17

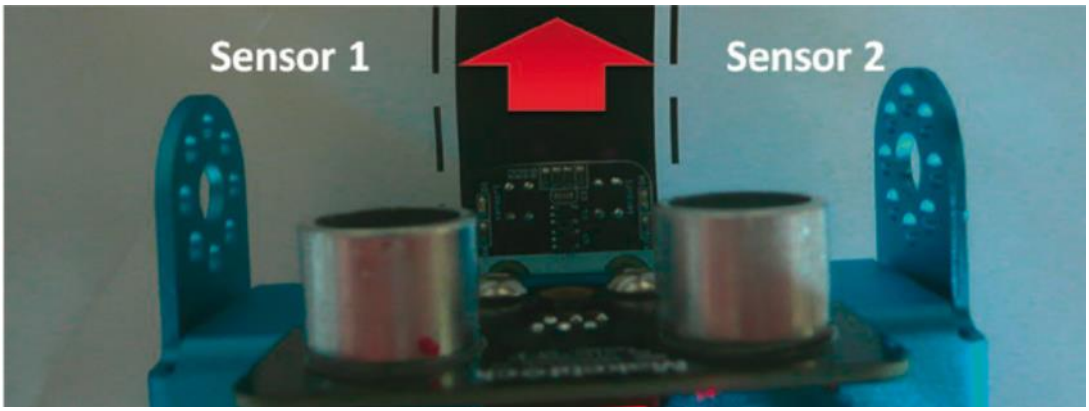


Abbildung 17: Linienfolgers-Sensor (Quelle[Open Robotik für Maker])

2. Möglichkeit: Rechts neben der Spur, die rechts Led lechtet, aber die links Led leutet nicht nach dem Abb.18

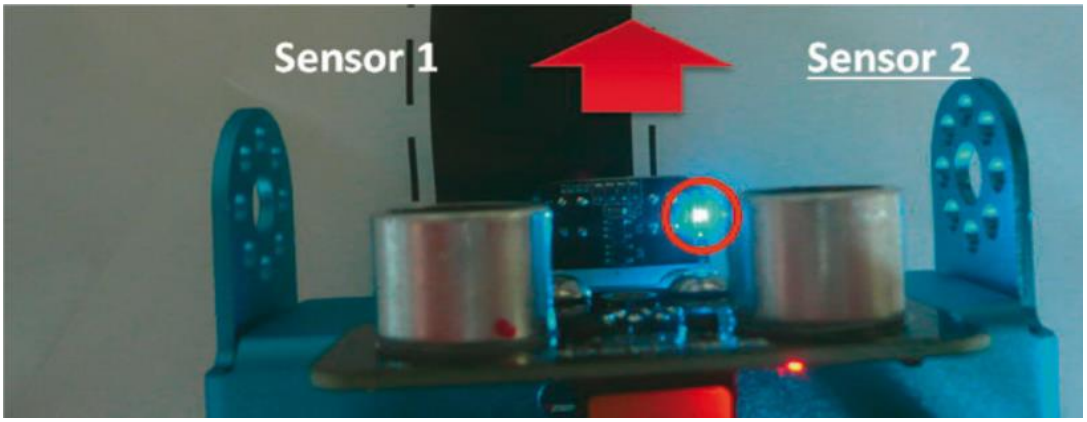


Abbildung 18: Linienfolgers-Sensor (Quelle[Open Robotik für Maker])

3. Möglichkeit:Links neben der Spur, die links Led lechtet, aber die rechts Led leutet nicht nach dem Abb.19

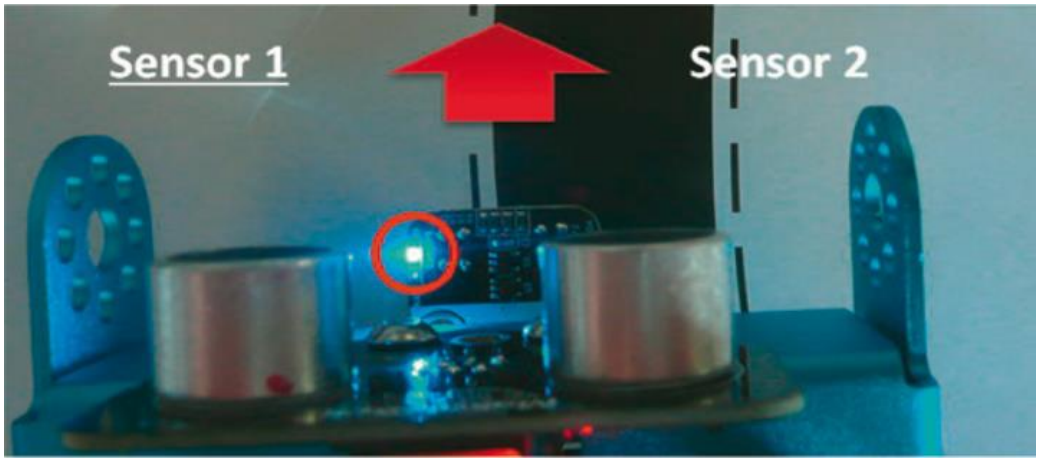


Abbildung 19: Linienfolgers-Sensor (Quelle[Open Robotik für Maker])

4. Möglichkeit: komplett neben der Spur, die beiden led leuchten nach dem Abb.20

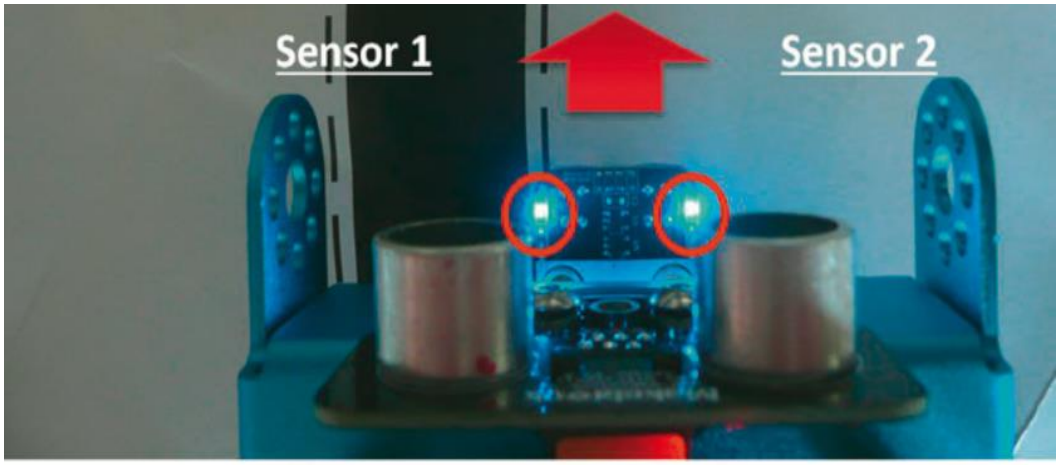
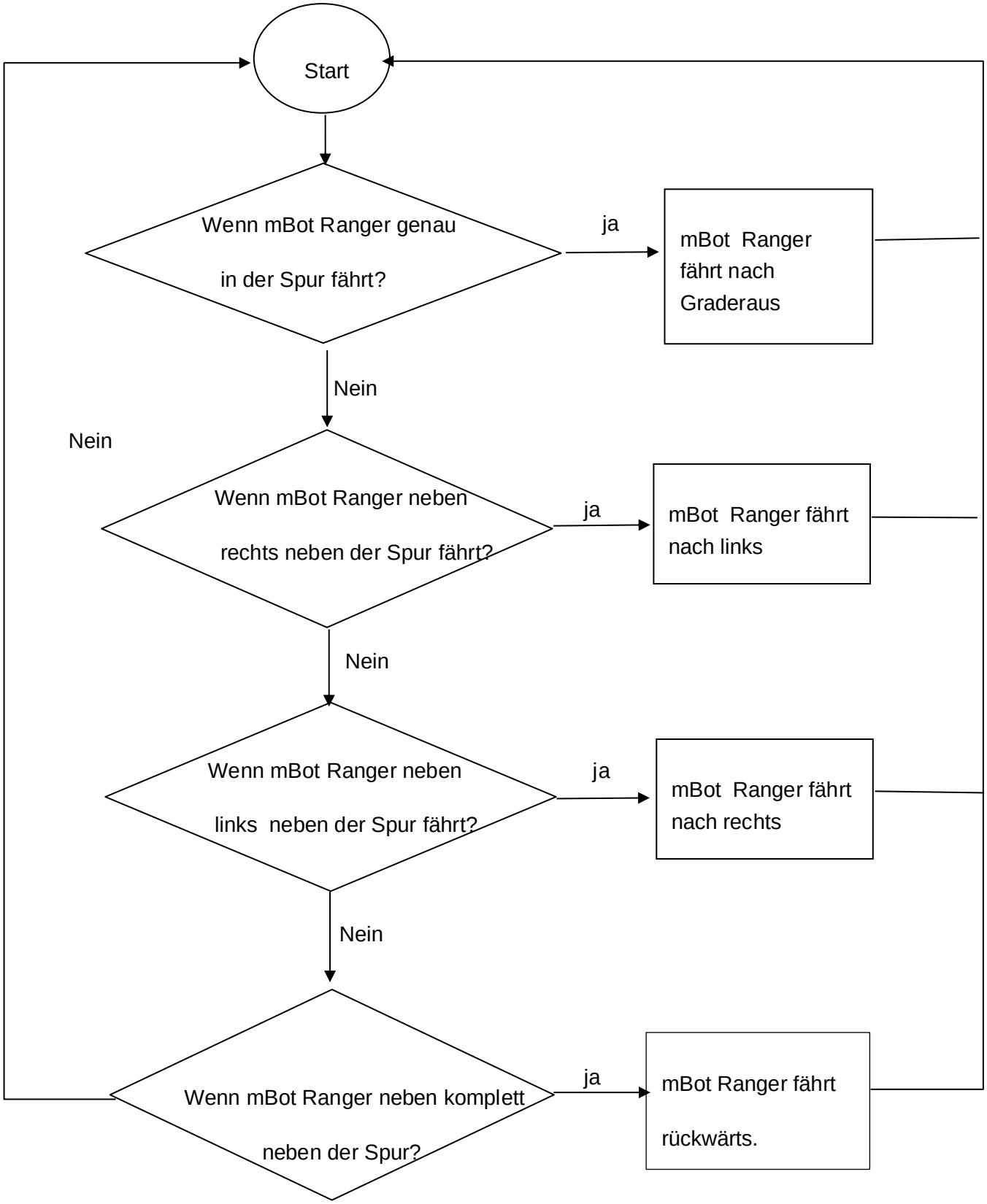


Abbildung 20: Linienfolgers-Sensor (Quelle[Open Robotik für Maker])

Deshalb jetzt steht die Erarbeitung von eine Ablauf von Algorithmus.



Bei unteren liegende Abb.21 wird parallel die mblock erstellt.

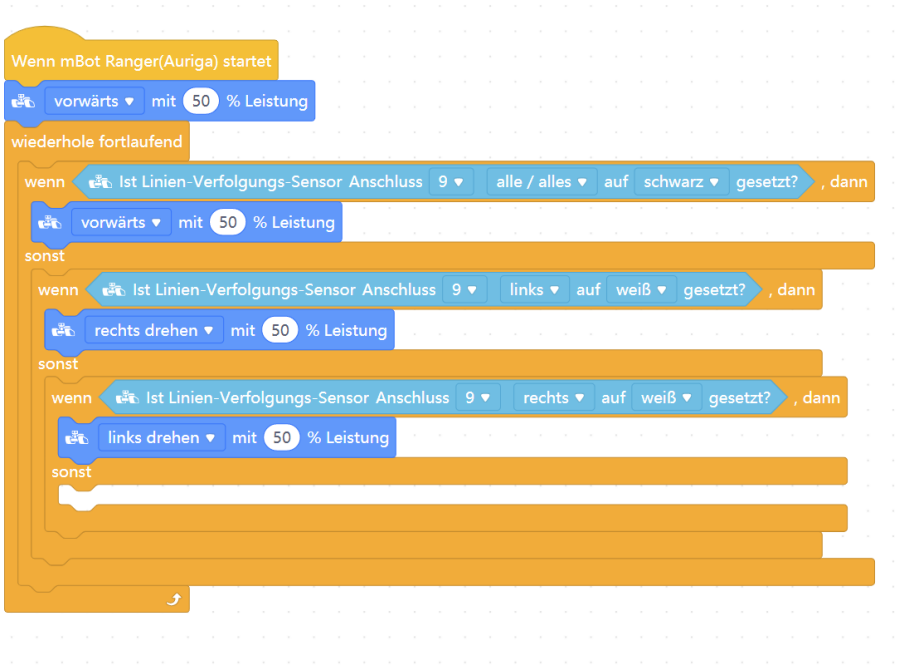


Abbildung 21: Coldblöcke 2 (Quelle[der mBot Ranger folgt einer Linie, mblock5])

Der Linienfolgersensor hat zwei Sensoren, die aus jeweils einer IR-LED und einem IR-Phototransistor bestehen.

Dieser Block beschreibt, dass der mBot Ranger mit einer Geschwindigkeit von 50% fährt, wenn der Linienfolger-Sensor an Anschluss 9 angeschlossen ist und auf beiden Seiten Schwarz erkannt wird. Das bedeutet, dass eine IR-LED Licht auf die Oberfläche ausstrahlt und der IR-Phototransistor das Schwarz absorbiert. Die beiden LEDs der Sensoren leuchten nicht. In diesem Fall ändert sich der Zustand des mBot Rangers nicht. Wenn der mBot Ranger jedoch links von der Spur fährt, leuchtet die rechte LED und der IR-Phototransistor erkennt das reflektierte Licht. Der mBot Ranger wird sich dann nach rechts bewegen. Wenn der Linienfolger-Sensor auf der rechten Seite Weiß erkennt, wird der mBot Ranger nach links fahren.

3.3 Fahren zur größte Lautstärke

Diese Applikation wird mit dem Schallsensor durchgeführt. Je nach unterschiedlicher Intensität des Geräuschs reagiert der mBot Ranger auch unterschiedlich. Der mBot Ranger ist mit einem Schallsensor am Board ausgestattet. Hierbei werden zwei spezifische Voraussetzungen festgestellt:

1. Der mBot Ranger muss zur größten Lautstärke fahren.
2. Da es nur einen Klangsensor zur Hand gibt, kann nur zwei verschiedene Klänge an verschiedenen Orten vergleichen werden.

Eine Beispiel:

Am Punkt A werden die Intensität des Geräusches gemessen. Dann fährt der mBot Ranger 1 Sekunde lang Gradeaus und hält hier für 3 Sekunden an. Nun befindet sich der mBot Ranger am Punkt B. Hier wird erneut die Schallintensität durch den inneren Schallsensor gemessen. Vergleicht man die beiden aktuellen unterschiedlichen lokalen Schallwerte, so bewegt sich mBot Ranger im eine senkund rückwärts,wenn der Schall von Punkt A größer ist als der Schall von Punkt B. Wenn der Schall von Punkt B am größten ist, muss mBot Ranger stehen bleiben.

Allerdings gibt es hier ein Problem: Der Schallintensitätssensor misst die Schallintensität kontinuierlich, was zu einem Wert führt, der sich von Sekunde zu Sekunde ändert. Das macht es schwierig, die beiden Werte direkt zu vergleichen. Deshalb wurde eine kleine Optimierung vorgenommen: die Werte der Schallintensität werden pro Sekunde aus dem Sensorausgang aufsummiert und danach werden die Durchschnitte berechnet. Auf diese Weise wird die Abweichung von den Werten nicht zu groß und führt nicht zu großen Fehlern.

Bei dem mblock Sketch gibt es solche Abläufe :

Abbildung 22: Codeblöck 3(Quelle[Fahren nach zur größte Lautstärke,mblock5])

Diese Anwendung basiert hauptsächlich auf Schallsensoren, die es dem mBot Ranger ermöglichen, sich in Richtung des lautesten Geräuschs zu bewegen. Mit weiteren Entwicklungen können Schallsensoren in vielen Bereichen eingesetzt werden, wie zum Beispiel bei schallgesteuerten Lichtern und Türen, die man im Alltag oft sieht. In der Industrierobotik kann man den Roboter per Sprache steuern, indem man ihn zum Beispiel nach links oder rechts dreht oder Gegenstände greifen lässt.

3.4 Die Anzeige von LED Nach Norden

In dieser Anwendung wird Kompass Sensor zum Einsatz kommen. Ein Kompasssensor ist ein Instrument zur Bestimmung der Himmelsrichtung. Der Kompasssensor funktioniert wie ein herkömmlicher und klassischer mechanischer Magnetkompass und zeigt mithilfe des Erdmagnetfeldes die Richtung zum magnetischen Nordpol an. Der Sensor kann nicht nur den Winkel der Himmelsrichtung bestimmen, sondern auch die Winkel in allen drei Dimensionen x, y und z. Um den Kompasssensor korrekt zu nutzen, muss man zunächst eine Kalibrierung durchführen und den Sensor in Richtung Norden ausrichten. Dies kann mithilfe des digitalen Kompasses in mBlock erfolgen.

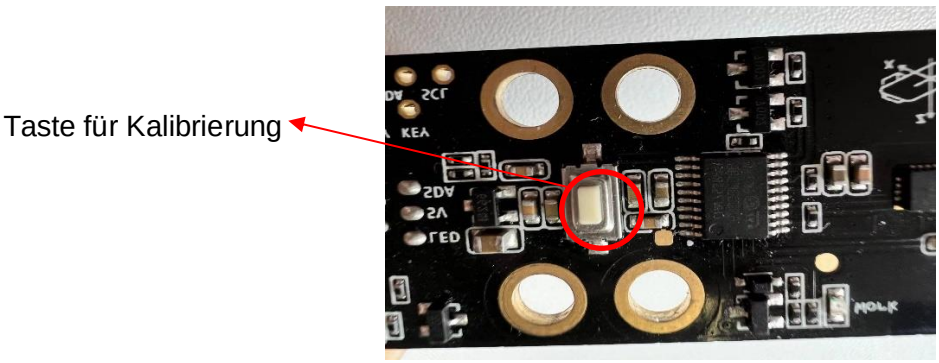


Abbildung 23: Kalibrierung von Kompass Sensor(Quelle[Labor, HS Mittweida])

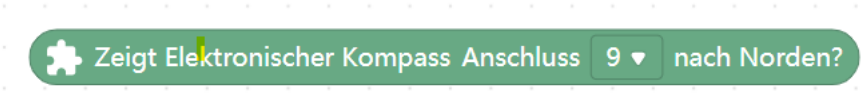


Abbildung 24: Elektonischer Kompass(Quelle[mblock5])

Nach Abbildung 23 ist deutlich zu erkennen, dass eine Anzeige des Winkels in Richtung Norden vorhanden ist. Der Kompasssensor wird vorne am mBot Ranger angebracht und zeigt den Winkel in Richtung Norden an.

Dann nach dem folgende Codeblöcke kann man deutlich ansehen,dass man die Leertaste drückt, alle Leds werden ausgeschaltet. Weil die RGB-Led aus 12 Leds in einer Kreis besteht, dann wird dem Kreis in 12 Bereiche unterteilt. Die Formel für diese Berechnung lautet:

$$aktuelle\ Bereich = \frac{die\ aktuelle\ Winkel\ nach\ Norden}{30^{\circ}}$$

Wenn mBot Ranger genau sich auf dem Nord(Winkel =0) befindet, die aktuelle Bereich ist currentangle. Aber die Led ist die Zahl 3 nach dem folgeden Abbildung 25.

Wenn der mBot Ranger langsam gegen den Uhrzeigersinn gedreht wird,dann die aktuelle Bereiche ist 1 und leutet die Led 2 in Rot.Das bedeutet,dass die Richtung von Led 2 jetzt nach Norden ist. wenn der mBot Ranger eine Umdrehung macht,leutet die LED leuchtet die LED in der Richtung der Drehung in Rot auf, die nach Norden zeigt.

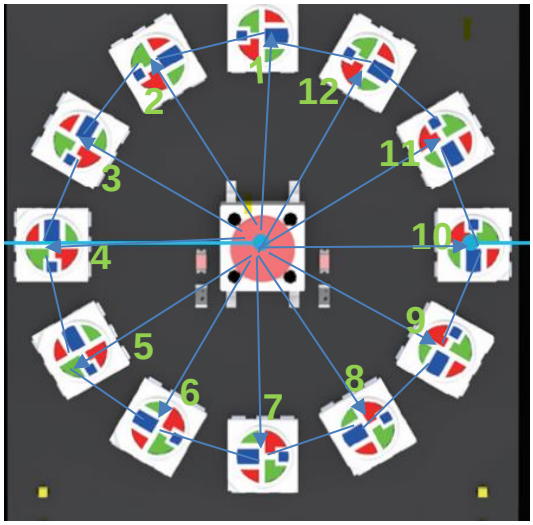


Abbildung 26:RGB-LED(Quelle[mBot ranger Bedienungsanleitung])

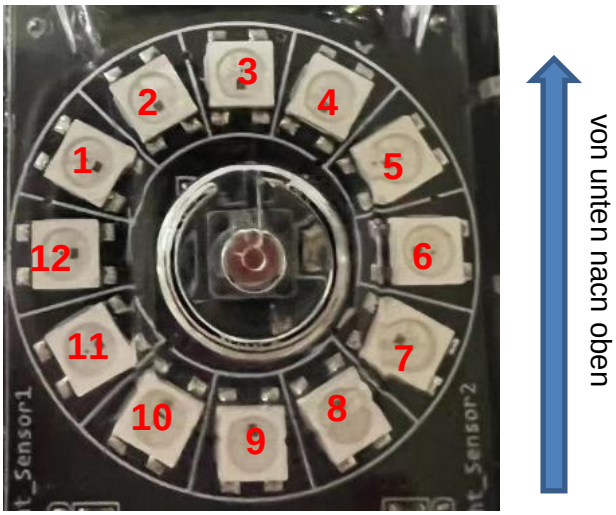


Abbildung 25:RGB-LED(Quelle[mBot Ranger, Mittweida])

Entsprechend der oben stehenden Abb. 25 ist dies der spezifische Winkelbereich und die Nummer von LEDs, der den 12 Bereichen entspricht, die innerhalb eines Kreises geteilt sind:

1.Bereich:	0°≤ ɵ <30° (LED 3)	7.Bereich:	180°≤ ɵ <210° (LED 9)
2.Bereich:	30°≤ ɵ <60° (LED 2)	8.Bereich:	210°≤ ɵ <240°(LED 8)
3.Bereich:	60°≤ ɵ <90° (LED 1)	9.Bereich:	240°≤ ɵ <270°(LED 7)
4.Bereich:	90°≤ ɵ <120° (LED 12)	10.Bereich:	270°≤ ɵ <300°(LED 6)
5.Bereich:	120°≤ ɵ <150° (LED 11)	11.Bereich:	300°≤ ɵ <330°(LED 5)
6.Bereich:	150°≤ ɵ <180°(LED 10)	12.Bereich:	330°≤ ɵ <360°(0°)(LED 4)

ɵ ist der Winkel nach Norden.

Hier ist folgende eine Beispiel von Abbilddung 26 :

Die Winkel jetzt durch Kompass Sensor nach Nord ist gleich 347.74 .Es befindet jetzt in der aktuelle Bereich 11, dann leutet Led 4.

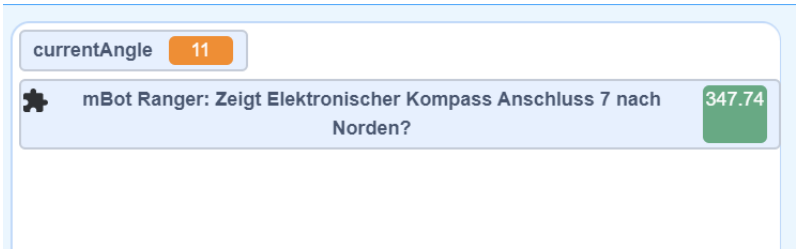


Abbildung 27: Anzeige von den Winkel(Quelle[Anzeigen nach Norden,mblock5])

Das Rechnungsverfahren:

ɵ=347.74° aktuelle Bereich = $\frac{347.74^{\circ}}{30^{\circ}} \approx 11$

Das bedeutet: Der aktuelle Bereich ist der 12. Bereich und die LED mit der Nummer 4 leuchtet rot.

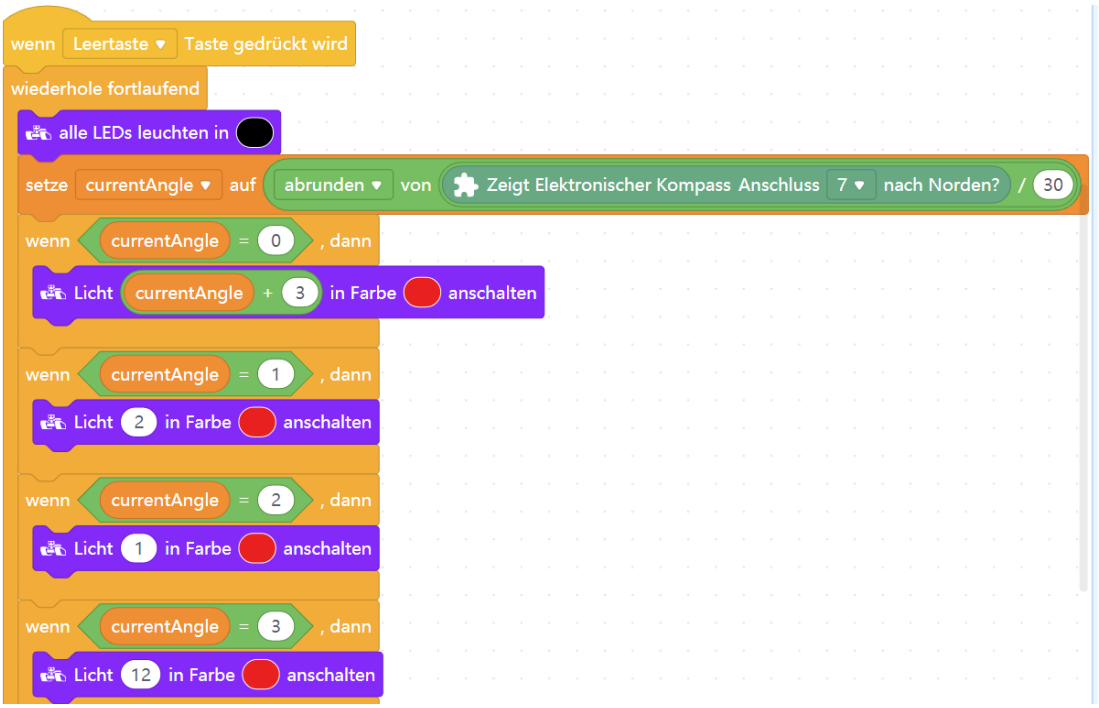


Abbildung 28: Coldeblöcke 4 (Quelle[Anzeigen nach Norden,mblock5])

3.5 Temperatur mit 7 Segement Anzeige

Der Temperatursensor wird in das Me-Augrina Board integriert und dient hauptsächlich zur Messung der Temperatur. Um die gemessene Temperatur anzuzeigen, kann eine externe 7-Segment-Anzeige an den Port 8 des mBot Ranger angeschlossen werden. Auf diese Weise kann die aktuelle Temperatur angezeigt werden.

Hier ist eine aktuelle gemessende Temperatur:

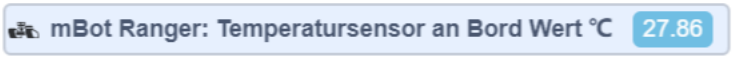


Abbildung 29: Temperaturswert an Bord (Quelle[Temperatur mit 7 Segement Anzeige,mblock5])

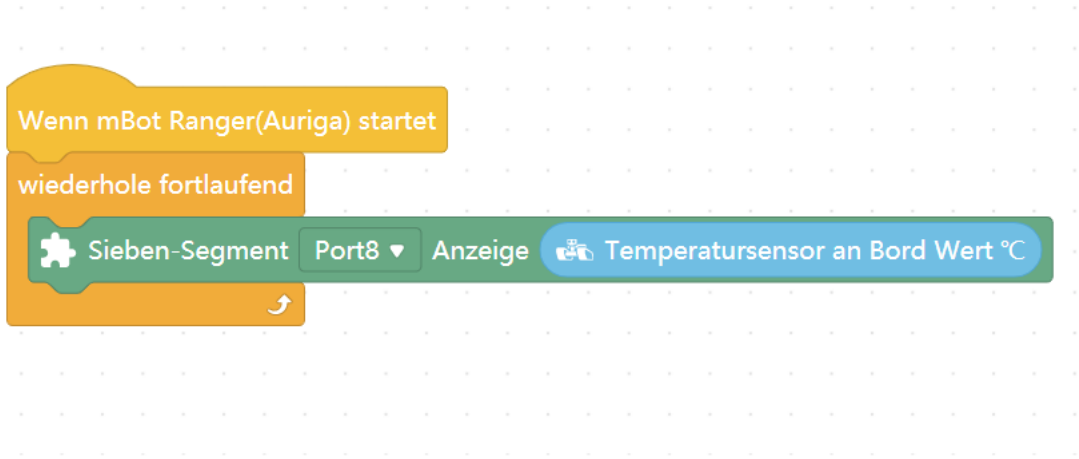
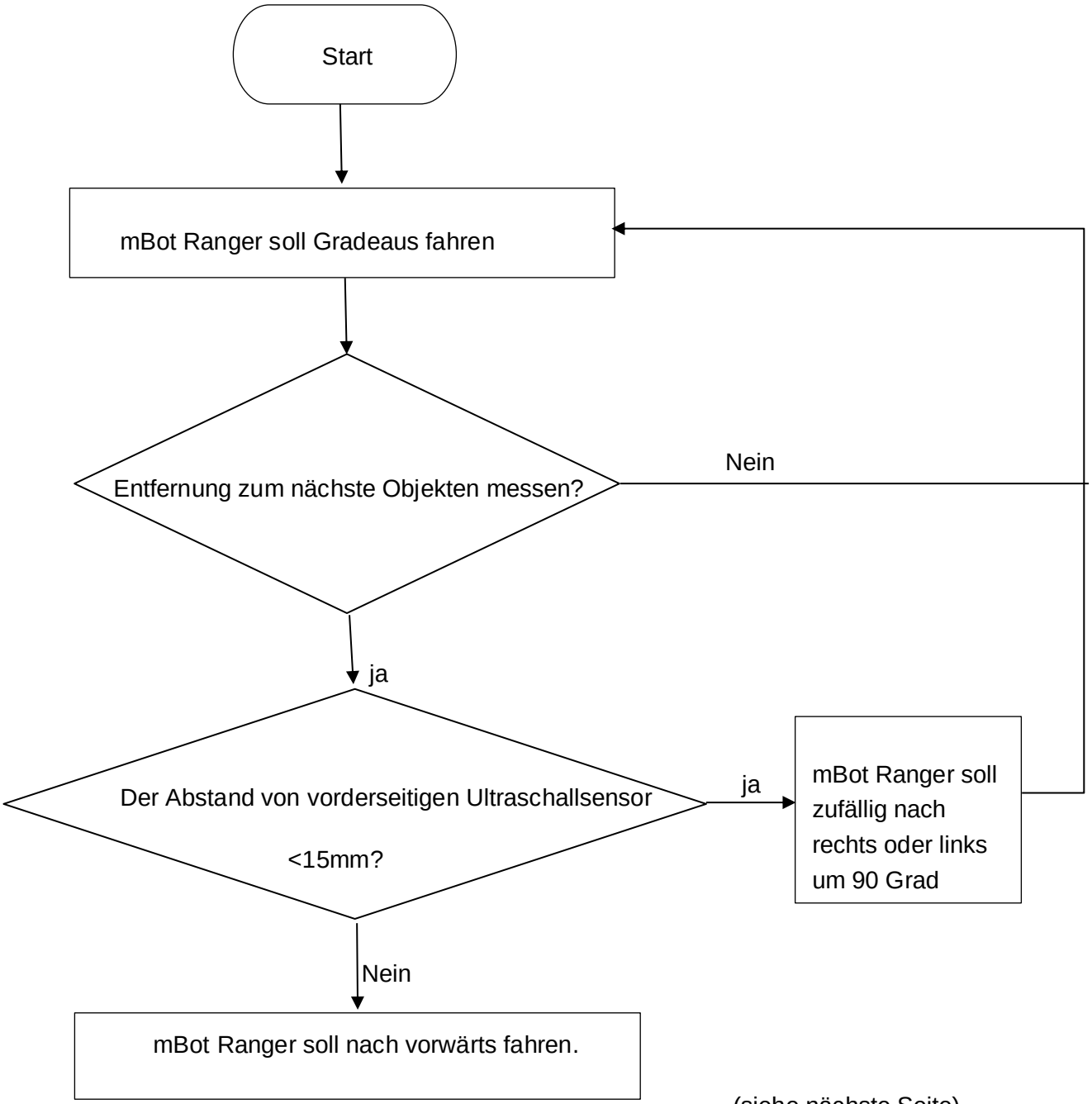


Abbildung 30: Codeblöck5(Quelle[Temperatur mit 7 Segement Anzeige,mblock5])

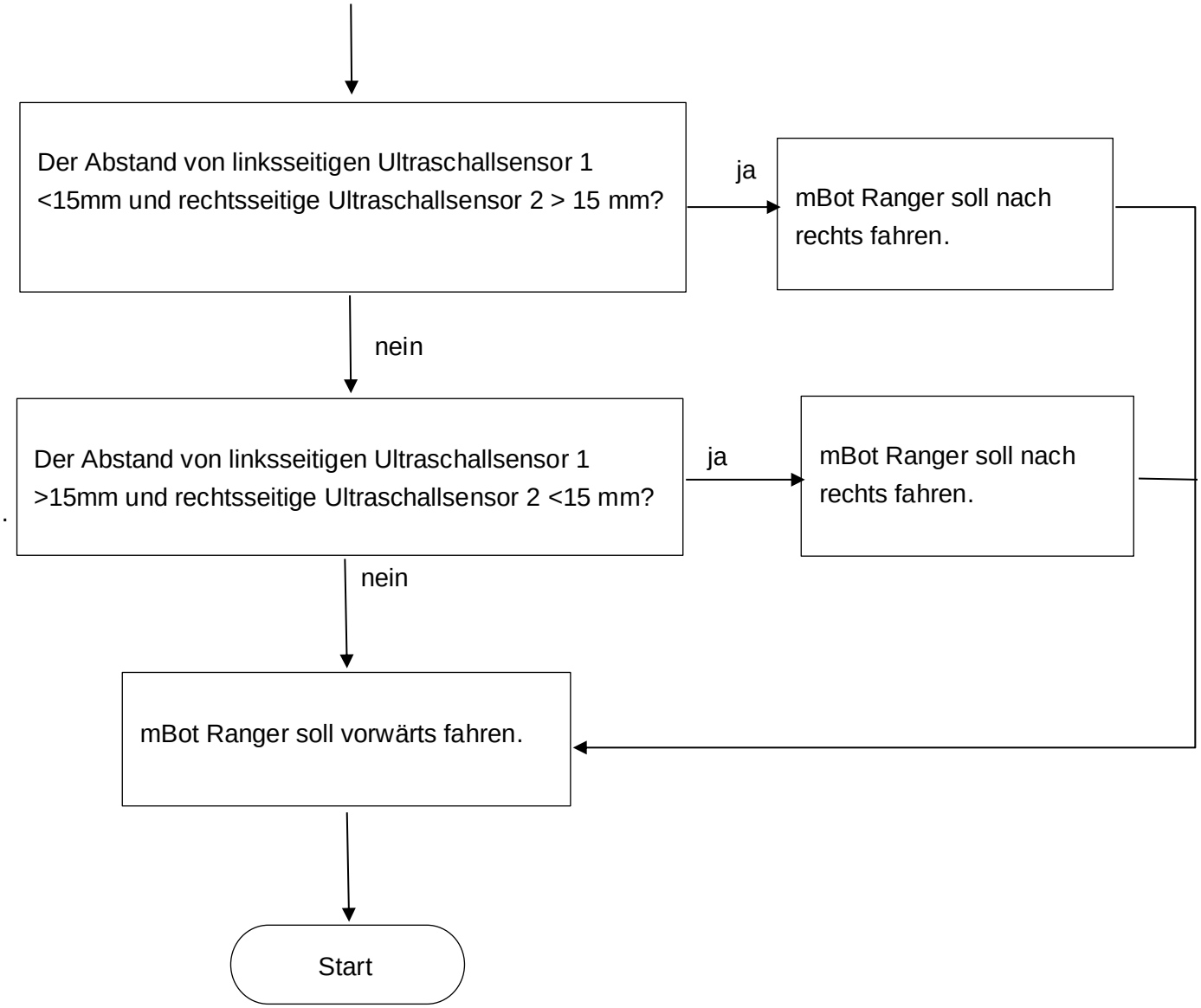
Wie aus den beiden obigen Abbildungen (Abb. 16 und Abb. 17) deutlich hervorgeht, wird durch den Anschluss einer externen Sieben-Segment-Anzeige die aktuelle Temperatur von 27,86 Grad Celsius angezeigt. Diese Temperatur entspricht der Temperatur an der Unterseite des am mBot Ranger installierten Me-Augrina-Boards.

3.6 Haltung von Mindestabstand

Der mBot Ranger soll durch zusätzliche Ultraschallsensoren an den Seiten immer einen Mindestabstand zu Objekten in der Nähe einhalten. Das bedeutet, dass der mBot Ranger auf der linken und rechten Seite mit zusätzlichen Ultraschallsensoren ausgestattet wird. Nach dem gleichen Prinzip wird zuerst der Abstand von beiden Seiten zu Objekten auf 15mm eingestellt. In dieser Applikation werden insgesamt drei Ultraschallsensoren verwendet, wobei der dritte Sensor an der Vorderseite des mBot Ranger angebracht ist. Im Folgenden wird auch ein Algorithmus vorgestellt:



(siehe nächste Seite)



Nach dem obigen Algorithmus wird der mBot Ranger im Codeblock von mBlock um 90 Grad nach links oder rechts gedreht. Dadurch tritt das Problem auf, dass der mBot Ranger den Mindestabstand zwischen den beiden Seiten nicht einhalten kann. Daher bietet es eine Lösung, die darin besteht, zunächst die Geschwindigkeit und die Zeit von Fahren von mBot Ranger zu verringern. Wie man auf den nachfolgenden Codeblöcken sehen kann, wird den mBot Ranger auf eine Geschwindigkeit von 30% eingestellt und bist 0,2 Sekunden lang nach links gefahren. Danach bewegt den mBot Ranger nach vorwärts mit 50% Geschwindigkeit für 1 Sekunde . Wenn die folgenden Codeblöcke auf dem mBot Ranger hochgeladen und die Start-Taste gedrückt werden, ist deutlich zu sehen, dass der mBot Ranger den Mindestabstand einhält. Dadurch wird auch das Problem der zu großen Drehwinkel des mBot Ranger beseitigt.

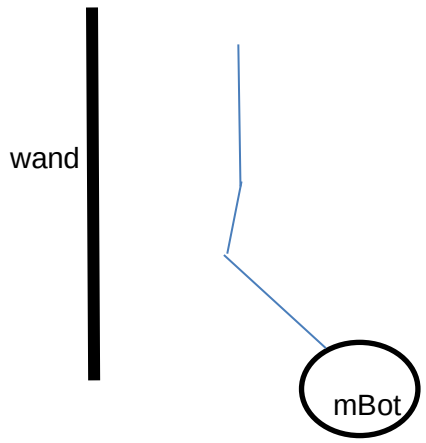
Abbildung 31: Codeblöcke 6(Quelle[Haltung von Mindestabstand,mblock5])

Um die Methode für diese Anwendung zu vereinfachen und zu optimieren, werden die erste Anwendung (Ausweichung von Hindernisse) und die sechste Anwendung (Haltung von Mindestabstand) unter Einsatz von Arduino kombiniert, und im Folgenden werden einige der wichtigsten Kerncodes gezeigt:

```
1.  if (cm1 < 15 && flag == 0) {
2.      flag = 1;
3.      angle_p = angle;
4.      rand = random(3, 5);
5.      } // Wenn vorne Ultraschallsensor1 eine Objekt detektiert, wird mBot
    zufällig nach rechts oder links um 90 Grad fahren
6.    else if (cm3 < 15 && flag == 0) {// Wenn rechtssseitig Ultraschallsensor3 die
    Objekt detektiert
7.      flag = 2; // Die Kennzeichen :  Abbiegung nach links
8.      angle_p = angle;// die Winkel von Gyroskopsensor wird bestimmt.
9.    }
10.      else if (cm2 < 15 && flag == 0) {// Wenn linksseitig Ultraschallsensor2
    die Objekt detektiert
11.       flag = 3;// Die Kennzeichen :  Abbiegung nach rechts
12.       angle_p = angle; // die Winkel von Gyroskopsensor wird bestimmt.
13.     }
14.     if (flag == 1) {// wenn der mBot eine vorne Objekt detektiert , es  wird um
    90 Grade umgedreht,
15.       flag2 = rand;
16.       move(rand, 50 / 100.0 * 255);// mBot soll nach rechts oder links fahren
17.       if (abs(angle_p - angle) >= 90) flag = 0;
18.     } // Wenn aktuelle Winkle minus vorher Winkel ist gleich 90, dann die
    Richtung wird erfolgreich verändert ,wird mBot in dieser Richtung geredeaus
    fahren.
```

```
19.     else if (flag == 2 || flag == 7) {//Wenn mBot eine rechtsseitige Objekt
    detektiert,
20.       flag2 = 3;
21.       move(3, 50 / 100.0 * 255);//mBot fährt nach links
22.       if (abs(angle_p - angle) >= 20 && flag == 2) flag = 4,angle_p =
    angle;//mBot fahren nach links um 20 Grade
23.       else if (abs(angle_p - angle) >= 10 && flag == 7) flag = 0;
24.     } //mBot fahren nach rechts um 10 Grad und danach Gradaus
25.     else if (flag == 3 || flag == 6) {// wenn mBot eine linksseitige Objekt
    detektiert,
26.       flag2 = 4;
27.       move(4, 50 / 100.0 * 255);// mBot  fährt nach rechts
28.       if (abs(angle_p - angle) >= 20 && flag == 3) flag = 5,angle_p = angle;//
    mBot fahrt nach rechts um 20 Grad
29.    else if (abs(angle_p - angle) >= 10 && flag == 6) flag = 0;// Danach wird
    mBot nach links um 90 Grad und danach Gradaus
30.     }
31.     else if (flag == 4 || flag == 5) {// mBot soll nach vorwärts fahren
32.       flag2 = 1;
33.       move(1, 50 / 100.0 * 255);
34.       //delay(300);
35.       if (flag_time == 0) flag_time = millis();
36.       {
37.         if ((millis() - flag_time) > 1000) {// Start
38.           flag += 2;
39.           flag_time = 0;// Die Lösung von Timer
40.           angle_p = angle;//Die  Bestimmung von Winkel
41.         }
42.       }
43.     } else {
44.       flag2 = 1;
45.       move(1, 50 / 100.0 * 255);// mBot soll nach vorwärts fahren
46.     }
```

Der Grundgedanke dieses Codes ist, dass das mBot Ranger, wenn es mit dem Ultraschallsensor auf der linken oder rechten Seite ein Hindernis erkennt, zunächst 20 Grad nach rechts oder links ausweicht, dann kurz Gradeaus fährt, dann 10 Grad nach links oder rechts ausweicht und wieder Gradeaus fährt, Die Bewegungsbahn dieses mBots ist in einem der folgenden Grafik dargestellt.



3.7 Die Anzeige von LED nach oben(Gyroskopsensor)

Diese Applikation beinhaltet einen eingebauten Gyroskopsensor und eine RGB-LED und basiert auf dem Prinzip eines digitalen Wasserwaagensystems, welches sicherstellt, dass sich der mBot Ranger in einem Gleichgewichtszustand befindet. Wenn der mBot Ranger auf einer unebenen Strecke fährt, leuchten die oberen LEDs je nach Neigungswinkel nach oben oder unten. Der Gyroskopsensor misst die Winkel in X, Y und Z Richtung. Jedoch kann dieses Problem nicht einfach mit dem mBlock5 gelöst werden, da man nur die 3 Achsenwinkel messen kann. Um eine effektive Lösung zu finden, ist es notwendig, die Arduino-Software zu verwenden. Bevor dies geschieht, werde die Lösung vorgestellt: die RGB-LED auf dem mBot Ranger wird auf einer Ebene platziert, da sie aus einem Kreis mit 12 LEDs besteht. Diese Ebene wird mit dem Mittelpunkt des Kreises als Ursprung definiert und die X- und Y-Achsen werden entsprechend ausgerichtet.

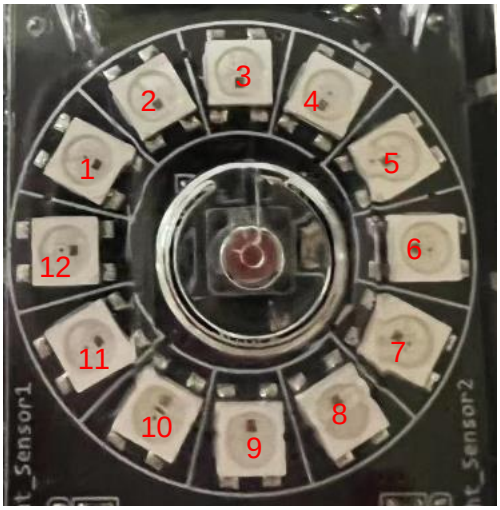
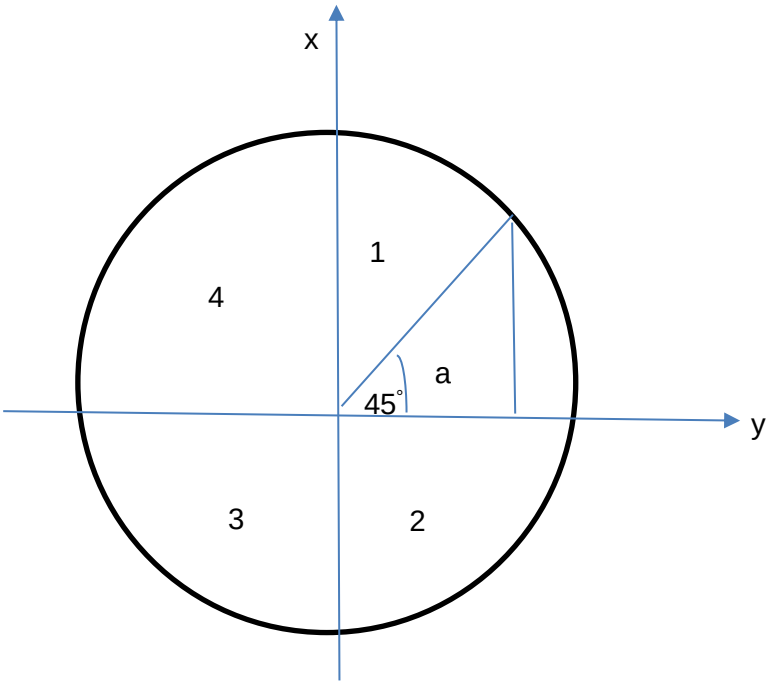


Abbildung 32:RGB-LED(Quelle[mBot Ranger, Mittweida])



Zunächst wird der Kreis in vier Quadranten verteilt und der Winkel a wird anhand des Beispiels im obigen Diagramm mit Hilfe der Funktion atan() berechnet.

Formel: $a = \arctan\left(\frac{abs(x)}{abs\ (y)}\right)$

Winkelsbereich von a : $0 < a < \frac{\pi}{2}$

Hier ist die wichtigste Code:

```
1. void redrawLeds(double x, double y, double z)
2. {
3.      x = constrain(x,-MAX_ANGLE, MAX_ANGLE);
4.      y = constrain(y,-MAX_ANGLE, MAX_ANGLE);
5.
6.      if (x < -ZERO_VAL && y < -ZERO_VAL) {
7.            lightLeds(x,y, 3, 5, 89, 0);
8.      }//Wenn der winkel a sich im ersten bereich
   befindet(90˚,0°),die Nummer von LED 3-5 beleuchtet.
9.      else if (x > ZERO_VAL && y < -ZERO_VAL) {
10.                lightLeds(x, y, 6, 8, 0, 89);
11.           }////Wenn der winkel a sich im ersten bereich
   befindet(0˚,90°),die Nummer von LED 6-8 beleuchtet
12.           else if (x > ZERO_VAL && y > ZERO_VAL) {
13.                lightLeds(x, y, 9, 11, 89, 0);
14.           }////Wenn der winkel a sich im ersten bereich
   befindet(90˚,0°),die Nummer von LED 9-11 beleuchtet
15.           else if (x < -ZERO_VAL && y> ZERO_VAL) {
16.                lightLeds(x, y, 12, 2, 0, 89);
17.           }////Wenn der winkel a sich im ersten bereich
   befindet(0˚,90°),die Nummer von LED 12-2 beleuchtet
18.           else
19.           {
20.                for (int i = 0; i < NUM_LEDS; i++)
21.                {
22.
23.                     led.setColorAt(i, 0, 0, 0);
24.                }
25.                led.show();
26.           }// Die Nummer von LED beleuchtet.
27. }
```

Das bedeutet, dass der Winkelbereich der x- und y-Achse auf einen begrenzten Bereich von π/2 bis -π/2 begrenzt ist. Wenn die Bibliothek neue Daten vom Gyroskop empfängt, ruft sie redrawLeds(x, y, z) mit drei Argumenten auf, die für yaw, pitch und roll stehen. Die beide Achsen (x, y) von -MAX_ANGLE bis +MAX_ANGLE werden beschränkt. Der maximale Winkel ist auf 45 Grad definiert und kann verändert werde. Um die Position der LEDs entsprechend der Winkelausrichtung des mBots besser bestimmen zu können, muss zunächst die Position des LEDs bestimmt werden, der den vier Ecken des mBots entspricht, die sich in Richtung 45 Grad befinden.

1. Wenn den mBot Ranger nach links oben gehobe wird, sind die Winkel auf der x- und der y-Achse beide kleiner als Null, so dass der Winkel a im ersten Quadranten liegt und die Bereich von a ist (89,0). Gleichzeitig wird die Leds von Nummer 3 bis zum Nummer 5 leuchten.

2. Der zweite Fall liegt vor, wenn der Winkel a größer als Null und kleiner als 90 Grad ist, was bedeutet, dass der Winkel a im zweiten Quadranten liegt ,ist der Winkel von x größer 0 und der Winkel y kleiner 0 ist , so dass die untere linke Ecke des mBot Ranger höher liegt als der Rest und die sechste bis achte LED aufleuchtet.
3. Wenn die untere rechte Ecke dieses mBots angehoben wird, leuchten die neunte Led und die elfte Led auf. Die Winkel von x-und y-Achse ist beiden größer als Null. Zu diesem Zeitpunkt befindet sich der Winkel a im dritten Quadranten, der von 89 Grad bis 0 Grad reicht.
4. Im letzten Fall, wenn der Winkel von x- Achse kleiner als 0 ist und der Winkel von ist und der Winkel von y-Achse größer als 0 ist, wird die obere rechte Ecke des mBots angehoben, leuchtet die von 0 auf 2 geführte LED auf. Zu diesem Zeitpunkt befindet sich der Winkel a im dritten Quadranten, der von 0 Grad bis 90 Grad reicht.

Die obige Beschreibung ist die Kernidee der Hauptlösung für dieses Problem, der nächste Schritt ist der Code für die Umrechnung des Winkels A, der dieser Position entspricht.

```
1.      double angle = (atan((double)abs(x) / (double)abs(y)) *
   4068) / 71;// Angel bestimmen
```

Zum Schluss wird noch die Messwerte angezeigt,wenn den mBot Ranger in verschiedene Richtungen gehoben wird.

```
                X       Y       a       n
31.458 -> -11   33      18      0
31.491 -> -11   31      19      0
31.534 -> -10   30      18      0
31.534 -> -10   27      20      0
31.580 -> -10   25      21      0
31.580 -> -9    23      21      0
31.626 -> -7    21      18      0
31.626 -> -5    21      13      0
31.672 -> -2    21      5       0
31.706 -> 3     21      8       10
31.748 -> 6     20      16      10
31.748 -> 8     19      22      10
31.794 -> 9     17      27      10
31.794 -> 10    15      33      10
31.839 -> 11    14      38      10
31.839 -> 12    13      42      10
31.874 -> 13    12      47      9
31.874 -> 13    11      49      9
31.914 -> 14    10      54      9
31.959 -> 15    9       59      9
31.959 -> 16    8       63      9
32.006 -> 17    8       64      9
32.006 -> 18    7       68      9
```

X-der Winkel von Gyroskopsensor nach x-Achse

Y-der Winkel von Gyroskopsensor nach y-Achse

a-der aktuelle Winkel

n-die Beleuchtung von Nummer von LED

Abbildung 33: Messwerten(Quelle[Arduino IDE])

4 Ergebnisse

In diesem Kapitel werden die Endergebnisse dieser Anwendungen beschrieben und der Status der Sensoren anhand dieser Ergebnisse bewertet.

4.1 Bewertung von unterschieden Sensoren

Verschiedene Anwendungen werden mit unterschiedlichen Sensoren ausgestattet, die jeweils eine andere Rolle bei der Produktion dieser Anwendung spielen. Da die erzielten Ergebnisse aber auch zeigen werden, dass die Genauigkeit des Sensors nicht stimmt, werde im Folgenden zeigen, was im Vorfeld getan wurde und was erwartet wird, bevor eine Bewertung des Sensors erfolgt. Dann wird auch erklärt, warum die beiden nicht übereinstimmen.

Während des Erstellung von Applikationen besitzt die Ultraschallsensor und Linien-folgers Sensor eine gute Genauigkeit und funktioniert sehr gut. Die zwei Appikationen sind Hindernisse Ausweichung und eine Folgerung von eine schwarz Linien, die für mBot Ranger sehr gut funktioniert.

Bei dritten Applikation nach Kapitel 3.3 ist der mBot Ranger mit hilfe von eine Schallsensor nach größte Lautstärke gefahren. Aber wenn die Applikation ausgeführt wird, ist es klar, dass sich das mBot Ranger manchmal nicht in Richtung des maximalen Geräusches bewegt, und manchmal in Richtung eines kleineren Geräusches. Für diese Situation gibt es zwei Gründe.Da es nur einen interne Geräuschsensor gibt, wird die Geräuschintensität vorne und hinten durch die Zeitdifferenz gemessen, wenn das Geräusch immer vorhanden ist, und dieses Mal kann der Geräuschsensor die Intensität des Geräuschs nicht schnell genug messen, so dass sich das mBot Ranger manchmal vorwärts und rückwärts bewegt. Die zweite Gründe ist dafür ,dass die Genauigkeit von diese Schallsenor nicht ausreichende ist. Um die gewünschten Ergebnisse zu erzielen, wäre es besser, einen externen Sensor zu haben, der die Schallintensität über zwei Sensoren empfängt und prüft, um die Richtung des mBot Rangers zu kontrollieren.

Die vierte Anwendung basiert auf den in Kapitel 3.4 gezeigten Ergebnissen, bei der der mBot Ranger mit einem externen Kompasssensor ausgestattet ist, durch den die LEDs über dem mBot Ranger immer in Richtung Norden leuchtet, wenn der mBot Ranger fährt. Nach den Ergebnissen des Tests funktioniert dieser Kompasssensor sehr gut, aber es ist zu empfehlen, vor jeder Installation des Kompasssensors am mBot Raner die Kalibrierung von Kompasssensor durchzuführen, um die Genauigkeit des Kompasssensors zu verbessern.

In Kapitel 3.5 wird der Temperatursensor verwendet, um die Umgebungstemperatur zu testen, aber da der Temperatursensor im Board installiert ist, wird es, wenn das mBot Ranger für eine gewisse Zeit gefahren wird, Wärme an der Unterseite des Boards geben, was zu einer Temperaturabweichung von der tatsächlichen Umgebungstemperatur führt, im Folgenden sind die Ergebnisse der Abb.33, die gemessen werden. Die erste Temperatur ist 28,91C˚, die mit 7 Segmenten angezeigt wird, die vom Temperatursensor im Bord des mBot Ranger gemessen werden, die zweite Temperatur ist 26.4 C˚ , die mit einem Thermometer gemessene Umgebungstemperatur ist. Aus dieser Abb.33 ist es auch ersichtlich, dass der im Board von mBot Ranger installierte Temperatursensor 1 bis 2 Grad Celsius höher ist als die normale Umgebungstemperatur.

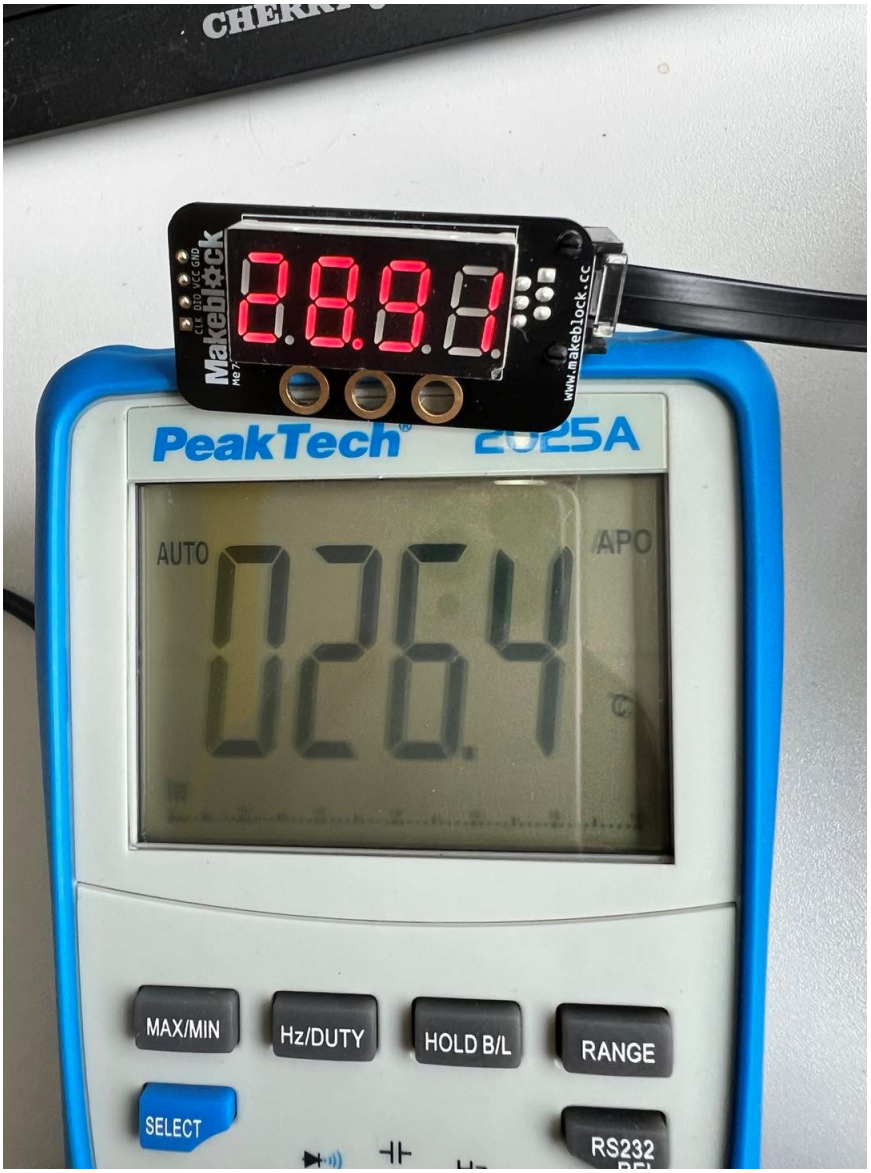


Abbildung 34: Zwei Messwerten von Temperaturen (Quelle[Labor,HS Mittweida])

Wenn der mBot Ranger in Kapitel 3.6 mit Hilfe von externen Ultraschallsensoren, die links und rechts angebracht sind, ein Hindernis innerhalb eines bestimmten Bereichs erkennt, wird der mBot Ranger dazu programmiert, in eine Richtung zu fahren, in der ein Mindestabstand eingehalten wird. Nach den Ergebnissen des Testlaufs des mBots ist der Ultraschallsensor in ausgezeichnetem Zustand.

Wie in Abschnitt 3.7 beschrieben, erkennt der mBot Rangers mit Hilfe eines internen Gyroskopsensors, ob sich der mBots im Gleichgewicht befindet, und wenn sich der Teil des mBots in der Aufwärtsposition befindet, leuchtet die diesem Teil entsprechende LED auf. Für die Programmierung dieser Anwendung, die auch ein Projekt "Gyroscope Fun with NeoPixel Ring"[2] im Internet nachahmt, wird die Arduino IDE verwendet. Anschließend diese Anwendung, die auch ein Projekt im Internet nachahmt wird, mit der Arduino-IDE programmiert und dann die Daten in den Mikrocontroller des mBot Rangers hochgeladen. Wenn das mBot Ranger in eine beliebige Richtung angehoben wird, erscheinen die folgenden Gyroskop-Testwinkel der X- und Y-Achse sowie der Winkel der angehobenen Position des Fahrzeugs a und die Nummer der leuchtenden LEDs

5 Zusammenfassung und Ausblick

5.1 Zusammenfassung und Ausblick

Die vorliegende Arbeiten befasst sich mit Erstellung von Applikationen für einen mBot Ranger.Dabei wurde die folgenden Sieben Fragenstellung untersucht:

- Ausweichung von Hindernisse
- Folgerung nach einer Linie
- Fahren zur größten Lautstärke
- Beleuchtung von Led nach Norden durch Kompass
- Anzeige von Temperatur mit 7-Segement
- Haltung von Mindestabstand
- Beleuchtung von Led nach Oben durch Gyroskopsensor

Die Methoden, die zur Beantwortung dieser Fragestellungen eingesetzt wurden, umfassen die Software mblock 5 und Arduino IDE. Mit der ersten Software mblock 5 können die meisten der oben genannten Probleme gelöst werden, und die zweite Software Arduino IDE kann die Probleme beseitigen, die die erste Software nicht lösen kann. Daraus wird deutlich, dass die Kombination der beiden Softwaretypen die besseren Lösungen für die Probleme darstellt. Diese Aufgaben werden mit einer Kombination aus Software und Hardware erledigt, d. h. werden muss zunächst die Programmanweisungen mit Hilfe von Software geschrieben, auf den mBot Ranger hochgeladen und mehrmals ausgeführt, getesten und geändert, um das gewünschte Endergebnis zu erzielen. Die Ergebnisse dieser Arbeit sind, dass der mBot Ranger in der Lage ist, Hindernissen mit den Ultraschallsensoren gut auszuweichen und einen Mindestabstand zwischen den beiden Seiten einzuhalten, indem zusätzliche Ultraschallsensoren auf beiden Seiten des Fahrzeugs installiert werden. Zweitens ermöglicht der Linien-Folgerssensor dem Roboter, der Linie zu folgen. Damit der mBot Ranger den Schallbefehlen folgen kann, werden die Schallsensoren verwendet, um den an zwei verschiedenen Orten gemessenen Schall zu vergleichen, und der mBot Rangerfährt zu dem Ort, an dem der Schall stärker ist. Allerdings waren das Ergebnis hier nicht so gut, wie es hätte sein können, da es nur einen Klangsensor an der Unterseite des mBot Rangers gab und der Klang nicht so genau war, wie er hätte sein können, und das mBot Ranger nicht in der Lage war, schnell genug in Richtung des lauten Klangs zu fahren. Als Nächstes misst der mBot Ranger die Temperatur mit Hilfe des Temperatursensors. Der Temperatursensor ist an der Unterseite der Board angebracht und misst eine höhere Temperatur als die Raumtemperatur, da der Motor Wärme abgibt, wenn der mBot eine Zeit lang gefahren wird. Um zu bestimmen, wohin der mBot Ranger in

Richtung Norden zeigt, muss man den Kompasssensor und das Leuchten der LED des RGB verwenden. Diese LED leuchtet, um als Zeigeranzeige zu dienen. Die letzte

Anwendung, die ebenfalls durch das Aufleuchten einer LED angezeigt werden muss, erkennt mit Hilfe eines Gyroskopsensors, ob der mBot Ranger im Gleichgewicht ist. Befindet sich mBot Ranger in einer aufwärts gerichteten Position, leuchtet die entsprechende LED in diese Richtung.Außerdem werden die Ergebnisse dieser Arbeitsläufe auf einem CD in Form eines Videos gespeichert.

Die vorliegende Arbeit bietet Möglichkeiten für weitere Forschung auf dem Gebiet von Robotik. Es gibt noch viele offene Fragen, die in Zukunft untersucht werden könnten, zum Beispiel werden Ultraschall- und Schallsensoren in der Realität verwendet, um Autos zu steuern, die während der Fahrt automatisch Hindernissen ausweichen. Die Ergebnisse dieser Arbeit könnten auch dazu beitragen, Robotik in Zukunft zu verbessern. Insgesamt trägt die Arbeit dazu bei, die Forschung in Erstellung von Applikationen für einen mBot Ranger voranzutreiben und neue Erkenntnisse zu gewinnen.

Literatur

[1] Erik Bartmann, Jörn Donges : Programmierspaß und smarte Elektronik mit Makeblock, München, Carl Hanser Verla, 2018

[2] https://projecthub.arduino.cc/danionescu/811f19ea-44ac-4c3b-afbe-f42090b948c8 , verfügbar am Dec 12, 2017

[3] https://www.bedienungsanleitu.ng/makeblock/mBot-ranger/anleitung, verfügbar am März 12,2013

[4] https://www.waycon.de/produkte/ultraschallsensoren/messprinzip-ultraschallsensoren ,verfügbar am Jan.19,2019

[5] https://www.arduino.cc/en/Main/Software ,verfügbar am March 17,2021

[6] https://github.com/Makeblock-official/Makeblock-Libraries/archive/master.zip , verfügbar am Okt.10,2020

[7] http://docs.makeblock.com ,verfügbar am Juli.16, 2020

[8] https://shop.technik-lpe.de/sensoren/884-temperatursensor-wasserdicht-6928819500624.html ,verfügbar am Feb 11. 2012

Anlagen

Anlagen, Teil 1:Quellcode

1.Hindernisse Ausweichung

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>

MeEncoderOnBoard Encoder_1(SLOT1);//Dateioutput von Motor1 mit Slot 1
MeEncoderOnBoard Encoder_2(SLOT2);//Dateioutput von Motor2 mit Slot 2
MeLineFollower linefollower_9(9);//Dateioutput von Linienfolgerssenor mit Anschluss 9
MeLightSensor lightsensor_12(12);//Dateioutput von Lichtssensor mit Anschluss 12

void isr_process_encoder1(void)
{
  if(digitalRead(Encoder_1.getPortB()) == 0){
    Encoder_1.pulsePosMinus();
  }else{
    Encoder_1.pulsePosPlus();
  }
} // Verarbeitung der Interrupt des Encoder1 Drvicer auf der Leiterplatte.um die Anzahl der Impulse zu berechnen.
void isr_process_encoder2(void)
{
  if(digitalRead(Encoder_2.getPortB()) == 0){
    Encoder_2.pulsePosMinus();
  }else{
    Encoder_2.pulsePosPlus();
  }
} // Verarbeitung der Interrupt des Encoder2 Drvicer auf der Leiterplatte.um die Anzahl der Impulse zu berechnen.
void move(int direction, int speed)
{
  int leftSpeed = 0;
  int rightSpeed = 0;//intialisierung von Speed
  if(direction == 1)
  {
    leftSpeed = -speed;
    rightSpeed = speed;
  }// Die Richtung is Vorwarts
  else if(direction == 2)
  {
    leftSpeed = speed;
    rightSpeed = -speed;
  }// Die Richtung is Rückwärts
  else if(direction == 3)
  {
    leftSpeed = -speed;
    rightSpeed = -speed;
  }// Die Richtung is nach links
  else if(direction == 4)
  {
    leftSpeed = speed;
    rightSpeed = speed;
  }// Die Richtung is nach rechts
  Encoder_1.setTarPWM(leftSpeed);
  Encoder_2.setTarPWM(rightSpeed);
}  // Das Auto fährt in welcher vier Richtung

void _delay(float seconds) {
  if(seconds < 0.0){
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
}

void setup() {
  TCCR1A = _BV(WGM10);
  TCCR1B = _BV(CS11) | _BV(WGM12);
  TCCR2A = _BV(WGM21) | _BV(WGM20);
  TCCR2B = _BV(CS21);
  // Einstellung PWM KHz
  attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1, RISING);
  attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2, RISING);
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));

  move(1, 50 / 100.0 * 255);//Das Auto fährt vorwärts mit Geschwindigkeit 50
  while(1)
  {
      if((0?(3==0?linefollower_9.readSensors()==0:(linefollower_9.readSensors() & 3)==3):(3==0?linefollower_9.readSensors()==3:(linefollower_9.readSensors() & 3)==0))){

          move(1, 50 / 100.0 * 255);

      } //Die linienfolgerssensor wird beiden Seiten auf Schwartz gesetzt? ja,das Auto fährt Vorwärts mit Geschwindigkeit 50
      else{
          if((1?(2==0?linefollower_9.readSensors()==0:(linefollower_9.readSensors() & 2)==2):(2==0?linefollower_9.readSensors()==3:(linefollower_9.readSensors() & 2)==0))){

              move(4, 50 / 100.0 * 255);

          } //Die linienfolgerssensor wird lins Seiten auf Schwartz gesetzt? ja,das Auto dreht rechts mit Geschwindigkeit 50
          else{
              if((1?(1==0?linefollower_9.readSensors()==0:(linefollower_9.readSensors() & 1)==1):(1==0?linefollower_9.readSensors()==3:(linefollower_9.readSensors() & 1)==0))){

                  move(3, 50 / 100.0 * 255);

              }//Die linienfolgerssensor wird rechts Seiten auf Schwartz gesetzt? ja,das Auto dreht links mit Geschwindigkeit 50
              else{

              }//Das Auto fährt vorwärts mit Geschwindigkeit 50

          }

      }

      _loop();
  }

}

void _loop() {
  Encoder_1.loop();
  Encoder_2.loop();
}

void loop() {
  _loop();
}
```

2.Linien-folger

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>
// Definieren Die Bibliotheke

MeUltrasonicSensor ultrasonic_10(10);//Dateioutput von Unltraschallsensor mit Anschluss 10
MeEncoderOnBoard Encoder_1(SLOT1);// Dateioutput von Motor 1 mit Slot 1
MeEncoderOnBoard Encoder_2(SLOT2);//Dateioutput von Motor mit Slot 2
MeLightSensor lightsensor_12(12);// Dateioutput von Lichtsensor mit Anscluss 12

void isr_process_encoder1(void)
{
  if(digitalRead(Encoder_1.getPortB()) == 0)
  {
    Encoder_1.pulsePosMinus();
  }
  else
  {
    Encoder_1.pulsePosPlus();
  }
}
// Verarbeitung der Interrupt des Encoder1 Drvicer auf der Leiterplatte.um die Anzahl der Impulse zu berechnen.
void isr_process_encoder2(void)
{
  if(digitalRead(Encoder_2.getPortB()) == 0)
  {
    Encoder_2.pulsePosMinus();
  }
  else
  {
    Encoder_2.pulsePosPlus();
  }
}
// Verarbeitung der Interrupt des Encoder2 Drvicer auf der Leiterplatte.um die Anzahl der Impulse zu berechnen.
void move(int direction, int speed)
{
  int leftSpeed = 0;
  int rightSpeed = 0; //intialisierung von Speed
  if(direction == 1)
  {
    leftSpeed = -speed;
    rightSpeed = speed;
  } // Die Richtung is Vorwarts
  else if(direction == 2)
  {
    leftSpeed = speed;
    rightSpeed = -speed;

  } //Die Richtung is Rückwärts
  else if(direction == 3)
  {
    leftSpeed = -speed;
    rightSpeed = -speed;
  }//Die Richtung is nach Links
  else if(direction == 4)
  {
    leftSpeed = speed;
    rightSpeed = speed;
  } //Die Richtung is nach Rechts
  Encoder_1.setTarPWM(leftSpeed);
  Encoder_2.setTarPWM(rightSpeed);
  // Das Auto fährt in welcher vier Richtung
}
void _delay(float seconds)
{
  if(seconds < 0.0)
  {
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
} // warten

void setup()
{
  TCCR1A = _BV(WGM10);
  TCCR1B = _BV(CS11) | _BV(WGM12);
  TCCR2A = _BV(WGM21) | _BV(WGM20);
  TCCR2B = _BV(CS21);
  //Einstellung PMW 8KHz
  attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1, RISING);
  attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2, RISING);
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));
  while(1)
  {
    if(ultrasonic_10.distanceCm() < 15)//der Abstand ist kleiner als 15?
    {

      move(3, 50 / 100.0 * 255);// ja, nach links mit 50

    }
    else
    {

      move(1, 50 / 100.0 * 255);// nein,nach rechts mit 50

    }

    _loop();
  }

}

void _loop() {
  Encoder_1.loop();
  Encoder_2.loop();
}
void loop() {
  _loop();
}//schleifen
```

3.Anzeige nach Norden

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>

MeRGBLed rgbled_0(0, 12);
MeCompass compass_7(7);//Kompass uploard
MeLightSensor lightsensor_12(12);// mit lightsensor Verbinden

float currentAngle = 0;
float Requested_angle = 0;
float schhwellenswert = 0;//Variablen einstellung

void _delay(float seconds) {
  if(seconds < 0.0){
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
}

void setup() {
  rgbled_0.setpin(44);
  rgbled_0.fillPixelsBak(0, 2, 1);
  compass_7.begin();
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));
  while(1) {

    rgbled_0.setColor(0,0,0,0);
    rgbled_0.show();
    currentAngle = floor(compass_7.getAngle() / 30);
    if(currentAngle == 0.000000){

      rgbled_0.setColor((currentAngle + 3),255,0,0);
      rgbled_0.show();

    }
    if(currentAngle == 1.000000){

      rgbled_0.setColor(2,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 1, dann leuchtet 2.Led
    if(currentAngle == 2.000000){

      rgbled_0.setColor(1,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 2, dann leuchtet 1.Led
    if(currentAngle == 3.000000){

      rgbled_0.setColor(12,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 3, dann leuchtet 12.Led
    if(currentAngle == 4.000000){

      rgbled_0.setColor(11,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 4, dann leuchtet 11.Led
    if(currentAngle == 5.000000){

      rgbled_0.setColor(10,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 5, dann leuchtet 10.Led
    if(currentAngle == 6.000000){

      rgbled_0.setColor(9,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 6, dann leuchtet 9.Led
    if(currentAngle == 7.000000){

      rgbled_0.setColor(8,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 7, dann leuchtet 8.Led
    if(currentAngle == 8.000000){

      rgbled_0.setColor(7,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 8, dann leuchtet 7.Led
    if(currentAngle == 9.000000){

      rgbled_0.setColor(6,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 9, dann leuchtet 6.Led
    if(currentAngle == 10.000000){

      rgbled_0.setColor(5,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 10, dann leuchtet 5.Led
```

```
    if(currentAngle == 11.000000){

      rgbled_0.setColor(4,255,0,0);
      rgbled_0.show();

    }// Wenn akuelle Bereiche ist 11, dann leuchtet 4.Led

    _loop();
  }

}

void _loop() {
}

void loop() {
  _loop();
}
```

4.Fahren zur größte Lautstärke

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>

MeEncoderOnBoard Encoder_1(SLOT1);// Dateioutput von Motor 1 mit Slot 1
MeEncoderOnBoard Encoder_2(SLOT2);// Dateioutput von Motor 2 mit Slot 2
MeLightSensor lightsensor_12(12);//Dateioutput von lichtsensor mit Anschluss 12
MeSoundSensor soundsensor_14(14);//Dateioutput von lauterstärkersensor mit Anschluss 6

float Sound = 0;
float Sound_2 = 0;
float Summe_1 = 0;
float Summe_2 = 0;
//Variablen definieren

void isr_process_encoder1(void)
{
  if(digitalRead(Encoder_1.getPortB()) == 0){
    Encoder_1.pulsePosMinus();
  }else{
    Encoder_1.pulsePosPlus();
  }
}// Verarbeitung der Interrupt des Encoder1 Drvicer auf der Leiterplatte.um die Anzahl der Impulse zu berechnen
void isr_process_encoder2(void)
{
  if(digitalRead(Encoder_2.getPortB()) == 0){
    Encoder_2.pulsePosMinus();
  }else{
    Encoder_2.pulsePosPlus();
  }
}// Verarbeitung der Interrupt des Encoder2 Drvicer auf der Leiterplatte.um die Anzahl der Impulse zu berechnen
void move(int direction, int speed)
{
  int leftSpeed = 0;
  int rightSpeed = 0;//initialisierung von Speed
  if(direction == 1)
  {
    leftSpeed = -speed;
    rightSpeed = speed;
  }// Die Richtung is Vorwärts
  else if(direction == 2)
  {
    leftSpeed = speed;
    rightSpeed = -speed;
  }// Die Richtung is Rückwarts
  else if(direction == 3)
  {
    leftSpeed = -speed;
    rightSpeed = -speed;
   }// Die Richtung is nach links
   else if(direction == 4)
   {
    leftSpeed = speed;
    rightSpeed = speed;
   }// Die Richtung is nach rechts
   Encoder_1.setTarPWM(leftSpeed);
   Encoder_2.setTarPWM(rightSpeed);
}
void L_C3_A4uterst_C3_A4rker (){
    Sound = 0;
    Sound_2 = 0;
    Summe_1 = 0;
    Summe_2 = 0; //initialisierung von Sound

    move(1, 50 / 100.0 * 255);
    _delay(1);
    move(1, 0);
    for(int count=0;count<10;count++){
        Sound = soundsensor_14.strength();
        Summe_1 += Sound;
    }
    Sound = Summe_1 / 10;

    move(1, 50 / 100.0 * 255);
    _delay(1);
    move(1, 0);
    for(int count2=0;count2<10;count2++){
        Sound_2 = soundsensor_14.strength();
        Summe_2 += Sound_2;
    }
    Sound_2 = Summe_2 / 10;
    _delay(1);//warten 1 sec

}

void _delay(float seconds) {
  if(seconds < 0.0){
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
}
```

```
}

void setup() {
  TCCR1A = _BV(WGM10);
  TCCR1B = _BV(CS11) | _BV(WGM12);
  TCCR2A = _BV(WGM21) | _BV(WGM20);
  TCCR2B = _BV(CS21);
  //Einstellung PMW 8KHz
  attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1, RISING);
  attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2, RISING);
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));
  L_C3_A4uterst_C3_A4rker();
  if(Sound > Sound_2)// wen die Größe von Sound groß als Die Größe von Lauterstärke 2
  {

      move(2, 50 / 100.0 * 255);// der Roborter fährt ruckwärts  mit 50
      _delay(5);// fährt bis 5 sec
      move(2, 0);//stoppen

  }else{
      Encoder_1.setTarPWM(0);
      Encoder_2.setTarPWM(0);
      _delay(0.5); // stoppen

  }
}

void _loop() {
  Encoder_1.loop();
  Encoder_2.loop();
}

void loop() {
  _loop();
}//schleifen
```

5.Anzeige von Temperatur

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>

MeOnBoardTemp temperature_onboard(PORT_13);// Dateioutputvon Temperaturssensor mit Anschluss 13
Me7SegmentDisplay seg7_8(8);// Dateioutput von 7 SegementDisplaymit Anschluss 8
MeLightSensor lightsensor_12(12);// Dateioutput von Lichtsensor mit Anscluss 12

float max = 0;
float min = 0;
float Lauterst_C3_A4rke = 0;
float Temp = 0;
//Variablen Definieren
void _delay(float seconds) {
  if(seconds < 0.0){
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
} // Starte der Roboter

void setup()
{
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));
  while(1)
  {

      seg7_8.display(float(temperature_onboard.readValue()));

      _loop();
  }//  Temperatur lesen und 7Segement anzeigen

}

void _loop() {
}

void loop() {
  _loop();
}
```

6.1 Mindestabstand halten

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>

MeUltrasonicSensor ultrasonic_6(6);
MeUltrasonicSensor ultrasonic_7(7);
MeEncoderOnBoard Encoder_1(SLOT1);
MeEncoderOnBoard Encoder_2(SLOT2);
MeLightSensor lightsensor_12(12);
MeUltrasonicSensor ultrasonic_10(10);

float ult1 = 0;
float ult2 = 0;
float ult_3 = 0;

void isr_process_encoder1(void)
{
  if(digitalRead(Encoder_1.getPortB()) == 0){
    Encoder_1.pulsePosMinus();
  }else{
    Encoder_1.pulsePosPlus();
  }
}
void isr_process_encoder2(void)
{
  if(digitalRead(Encoder_2.getPortB()) == 0){
    Encoder_2.pulsePosMinus();
  }else{
    Encoder_2.pulsePosPlus();
  }
}
void move(int direction, int speed)
{
  int leftSpeed = 0;
  int rightSpeed = 0;
  if(direction == 1){
    leftSpeed = -speed;
    rightSpeed = speed;
  }else if(direction == 2){
    leftSpeed = speed;
    rightSpeed = -speed;
  }else if(direction == 3){
    leftSpeed = -speed;
    rightSpeed = -speed;
  }else if(direction == 4){
    leftSpeed = speed;
    rightSpeed = speed;
  }
  Encoder_1.setTarPWM(leftSpeed);
  Encoder_2.setTarPWM(rightSpeed);
}
void ult3 (){
    while(1) {
        ult_3 = ultrasonic_10.distanceCm();
        if(ult_3 < 15){

            move(3, 30 / 100.0 * 255);

        }else{

            move(1, 30 / 100.0 * 255);

        }

        _loop();
    }

}

void _delay(float seconds) {
  if(seconds < 0.0){
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
}

void setup() {
  TCCR1A = _BV(WGM10);
  TCCR1B = _BV(CS11) | _BV(WGM12);
  TCCR2A = _BV(WGM21) | _BV(WGM20);
  TCCR2B = _BV(CS21);
  attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1, RISING);
  attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2, RISING);
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));
  ult3();
  while(1) {
      ult1 = ultrasonic_6.distanceCm();
      ult2 = ultrasonic_7.distanceCm();
      if((ult1 < 15)  &&  (ult2 > 15)){

          move(3, 30 / 100.0 * 255);
          _delay(0.2);
          move(3, 0);
```

```
      move(3, 0);

      move(1, 50 / 100.0 * 255);
      _delay(1);
      move(1, 0);

    }else{
      if((ult2 < 15)  &&  (ult1 > 15)){

        move(4, 30 / 100.0 * 255);
        _delay(0.2);
        move(4, 0);

        move(1, 50 / 100.0 * 255);
        _delay(1);
        move(1, 0);

      }else{

        move(1, 50 / 100.0 * 255);

      }
    }

    _loop();
  }

}

void _loop() {
  Encoder_1.loop();
  Encoder_2.loop();
}

void loop() {
  _loop();
}
```

6.2 Mindestabstand halten und die Hindernisse Ausweichung

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <MeAuriga.h>
MeGyro gyro(0, 0x69);
MeUltrasonicSensor ultrasonic_10(10);
MeUltrasonicSensor ultrasonic_7(7);
MeUltrasonicSensor ultrasonic_6(6);
MeEncoderOnBoard Encoder_1(SLOT1);
MeEncoderOnBoard Encoder_2(SLOT2);
MeLightSensor lightsensor_12(12);

void isr_process_encoder1(void) {
  if (digitalRead(Encoder_1.getPortB()) == 0) {
    Encoder_1.pulsePosMinus();
  } else {
    Encoder_1.pulsePosPlus();
  }
}
void isr_process_encoder2(void) {
  if (digitalRead(Encoder_2.getPortB()) == 0) {
    Encoder_2.pulsePosMinus();
  } else {
    Encoder_2.pulsePosPlus();
  }
}
void move(int direction, int speed) {
  int leftSpeed = 0;
  int rightSpeed = 0;
  if (direction == 1) {
    leftSpeed = -speed;
    rightSpeed = speed;
  } else if (direction == 2) {
    leftSpeed = speed;
    rightSpeed = -speed;
  } else if (direction == 3) {
    leftSpeed = -speed;
    rightSpeed = -speed;
  } else if (direction == 4) {
    leftSpeed = speed;
    rightSpeed = speed;
  }
  Encoder_1.setTarPWM(leftSpeed);
  Encoder_2.setTarPWM(rightSpeed);
}

void _delay(float seconds) {
  if (seconds < 0.0) {
    seconds = 0.0;
  }
  long endTime = millis() + seconds * 1000;
  while (millis() < endTime) _loop();
}
void setup() {
  TCCR1A = _BV(WGM10);
  TCCR1B = _BV(CS11) | _BV(WGM12);
  TCCR2A = _BV(WGM21) | _BV(WGM20);
  TCCR2B = _BV(CS21);
  attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1, RISING);
  attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2, RISING);
  randomSeed((unsigned long)(lightsensor_12.read() * 123456));
  Serial.begin(9600);
  randomSeed(analogRead(A0));
  gyro.begin();
  int angle = 0, angle_p = 0;
  int flag = 0,flag2=1;
  long flag_time = 0;
  int rand;                       //Zufällig Richtung
  while (1) {


    int cm1 = ultrasonic_10.distanceCm();
    int cm2 = ultrasonic_7.distanceCm();
    int cm3 = ultrasonic_6.distanceCm();
    double x, y, z;
    gyro.update();
    x = gyro.getAngleX();
    y = gyro.getAngleY();
    z = gyro.getAngleZ();
    angle = z;
/*   Serial.print(cm1);
    Serial.print(' ');
    Serial.print(cm2);
    Serial.print(' ');
    Serial.print(cm3);
    Serial.print(' ');
    Serial.print(z);
    Serial.print(' ');
    Serial.println();
*/

    if (cm1 < 15 && flag == 0) {
      flag = 1;
      angle_p = angle;
      rand = random(3, 5);

    } // Wenn vorne Ultraschallsensor1 eine Objekt detektiert, wird mBot zufällig nach rechts oder links um 90 Grad fahren
    else if (cm3 < 15 && flag == 0) {{// Wenn rechtssseitig Ultraschallsensor3 die Objekt detektiert
      flag = 2;// Die Kennzeichen :  Abbiegung nach links
      angle_p = angle;//die Winkel von Gyroskopsensor wird bestimmt.
    }
    else if (cm2 < 15 && flag == 0) {// wenn ult1 vorne eine Objekt detektieren, werde der Roboter zufällig nach rechts oder Links um 90 grad fahren.
      flag = 3;//Die Kennzeichen :  Abbiegung nach rechts
      angle_p = angle;//die Winkel von Gyroskopsensor wird bestimmt.
    }
    if (flag == 1) { // wenn der mBot eine vorne Objekt detektiert , es  wird um 90 Grade umgedreht,
      flag2 = rand;
      move(rand, 50 / 100.0 * 255);//mBot soll nach rechts oder links fahren
      if (abs(angle_p - angle) >= 90) flag = 0;
```

```
    } //Wenn aktuelle Winkle minus vorher Winkel ist gleich 90, dann die Richtung wird erfolgreich verändert ,wird mBot in dieser Richtung geredeaus fahren.
    else if (flag == 2 || flag == 7) {//Wenn mBot eine rechtsseitige Objekt detektiert,
      flag2 = 3;
      move(3, 50 / 100.0 * 255);//mBot fährt nach links
      if (abs(angle_p - angle) >= 20 && flag == 2) flag = 4,angle_p = angle;//;//mBot fahren nach links um 20 Grade
      else if (abs(angle_p - angle) >= 10 && flag == 7) flag = 0;//mBot fahren nach rechts um 10 Grad und danach Gradaus
    }
    else if (flag == 3 || flag == 6) {//wenn mBot eine linksseitige Objekt detektiert,
      flag2 = 4;
      move(4, 50 / 100.0 * 255);//mBot  fährt nach rechts
      if (abs(angle_p - angle) >= 20 && flag == 3) flag = 5,angle_p = angle;//mBot fahrt nach rechts um 20 Grad
      else if (abs(angle_p - angle) >= 10 && flag == 6) flag = 0;// Danach wird mBot nach links um 90 Grad und danach Gradaus
    }
    else if (flag == 4 || flag == 5) {
      flag2 = 1;
      move(1, 50 / 100.0 * 255);//mBot soll nach vorwärts fahren
      //delay(300);
      if (flag_time == 0) flag_time = millis();//start
      else
      {
        if ((millis() - flag_time) > 1000) {
          flag += 2;
          flag_time = 0;//Die Lösung von Timer
          angle_p = angle;//Die  Bestimmung von Winkel
        }
      }
    } else {
      flag2 = 1;
      move(1, 50 / 100.0 * 255);//mBot soll nach vorwärts fahren
    }
    Serial.print('A');
    Serial.print(flag);
   // move(flag2, 100 / 100.0 * 255);
    _loop();
   /* Serial.print(cm1);
    Serial.print(' ');
    Serial.print(cm2);
    Serial.print(' ');
    Serial.print(cm3);
    Serial.print(' ');
    Serial.print(z);
    Serial.print(' ');
    Serial.print(flag);*/
    Serial.println();
  }
}

void _loop() {
  Encoder_1.loop();
  Encoder_2.loop();
}

void loop() {
  _loop();
  Serial.println(1);
}
```

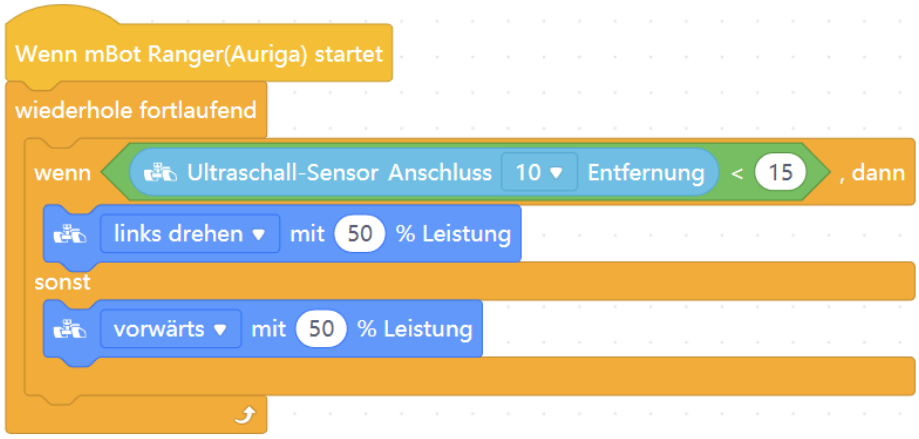
7.Gyroskopsensor

```
#include <Arduino.h>
#include "MeOrion.h"
#include <Wire.h>
//#include <MeAuriga.h>

#define NUM_LEDS 12
#define ZERO_VAL 0.5
MeGyro gyro(0,0x69);
MeRGBLed led(0, 12);//led initialisierung

const int MAX_ANGLE = 45; // MAximal Winkel bestimmen
const int LED_OFFSET = 0;//Intinialisierung von LED

unsigned long lastPrintTime = 0;

void setup()
{
  Serial.begin(115200);
  gyro.begin();
  led.setpin(44);
  Serial.println("x\ty\ta\tn");
} // set up
void loop()
{
  double x, y, z;
  gyro.update();
  x = gyro.getAngleX();
  y = gyro.getAngleY();
  z = gyro.getAngleZ();
  redrawLeds(x, y, z);
  delay(20);
} //  Ausgabe von winkel von gyroskope

void redrawLeds(double x, double y, double z)
{
  x = constrain(x,-MAX_ANGLE, MAX_ANGLE);
  y = constrain(y,-MAX_ANGLE, MAX_ANGLE);

  if (x < -ZERO_VAL && y < -ZERO_VAL) {
    lightLeds(x,y, 3, 5, 89, 0);
  }// die winkel a in den 1.Quadrate, die von 3-5 led leutchen
  else if (x > ZERO_VAL && y < -ZERO_VAL) {
    lightLeds(x, y, 6, 8, 0, 89);
  }// die winkel a in den 2 .Quadrate, die von 6-8 led leutchen
  else if (x > ZERO_VAL && y > ZERO_VAL) {
    lightLeds(x, y, 9, 11, 89, 0);
  }//die winkel a in den 3.Quadrate, die von 9-11 led leutchen
  else if (x < -ZERO_VAL && y> ZERO_VAL) {
    lightLeds(x, y, 0, 2, 0, 89);
  }//die winkel a in den 4.Quadrate, die von 0-2 led leutchen
  else
  {
    for (int i = 0; i < NUM_LEDS; i++)
    {

      led.setColorAt(i, 0, 0, 0);
    }
```

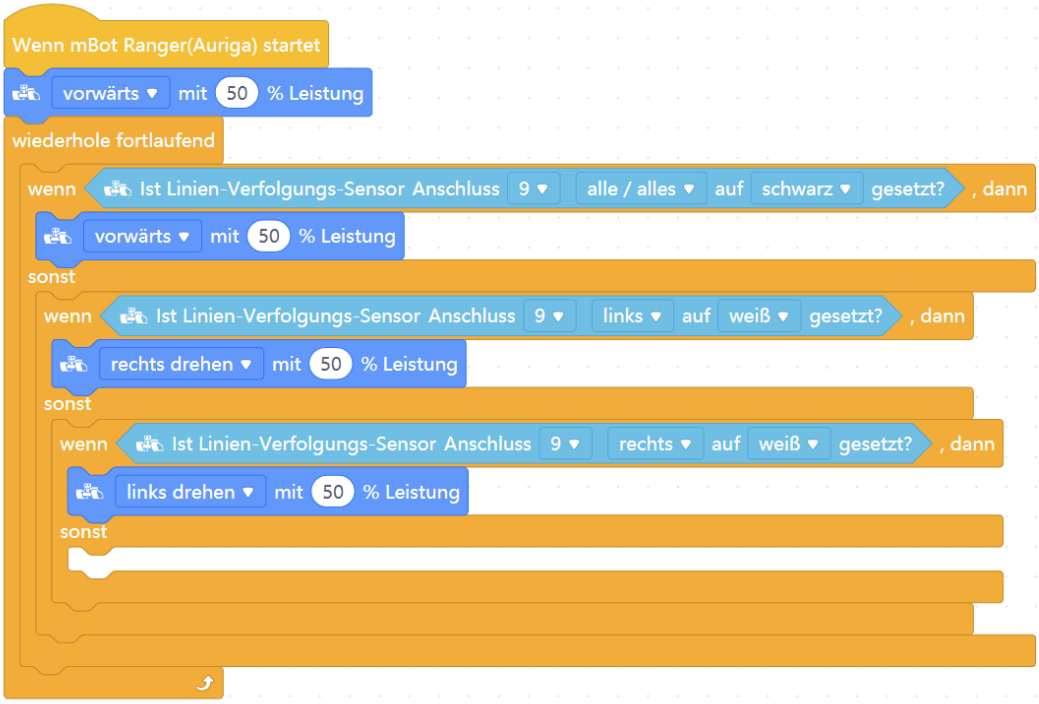
```
    }
    led.show();
  }// die Anzahl von led leutechen
}
void lightLeds(double x, double y, int fromLedPosition, int toLedPosition, int fromAngle, int toAngle)
{
  double angle = (atan((double)abs(x) / (double)abs(y)) * 4068) / 71;// angel bestimmen
  int ledNr = map(angle, fromAngle, toAngle, fromLedPosition, toLedPosition);
  printDebug(x, y, ledNr, angle);

  for (int i = 0; i < NUM_LEDS; i++)
  {
    uint8_t red = 0;
    uint8_t green = 0;
    uint8_t blue = 0;
    if (i == ledNr)
    {
      red = 180;
    }
    led.setColorAt(i, red, green, blue);
  }
  led.show();
}  //Farbe und Nummer von LEDs bestimmen

void printDebug(int x, int y, int lightLed, int angle)
{
  Serial.print(x);
  Serial.print('\t');
  Serial.print(y);
  Serial.print('\t');
  Serial.print(angle);
  Serial.print('\t');
  Serial.println(lightLed);
}// Ausgabe von x,y a, anzahl von led
```

Anlagen, Teil 2:Coldblöcke

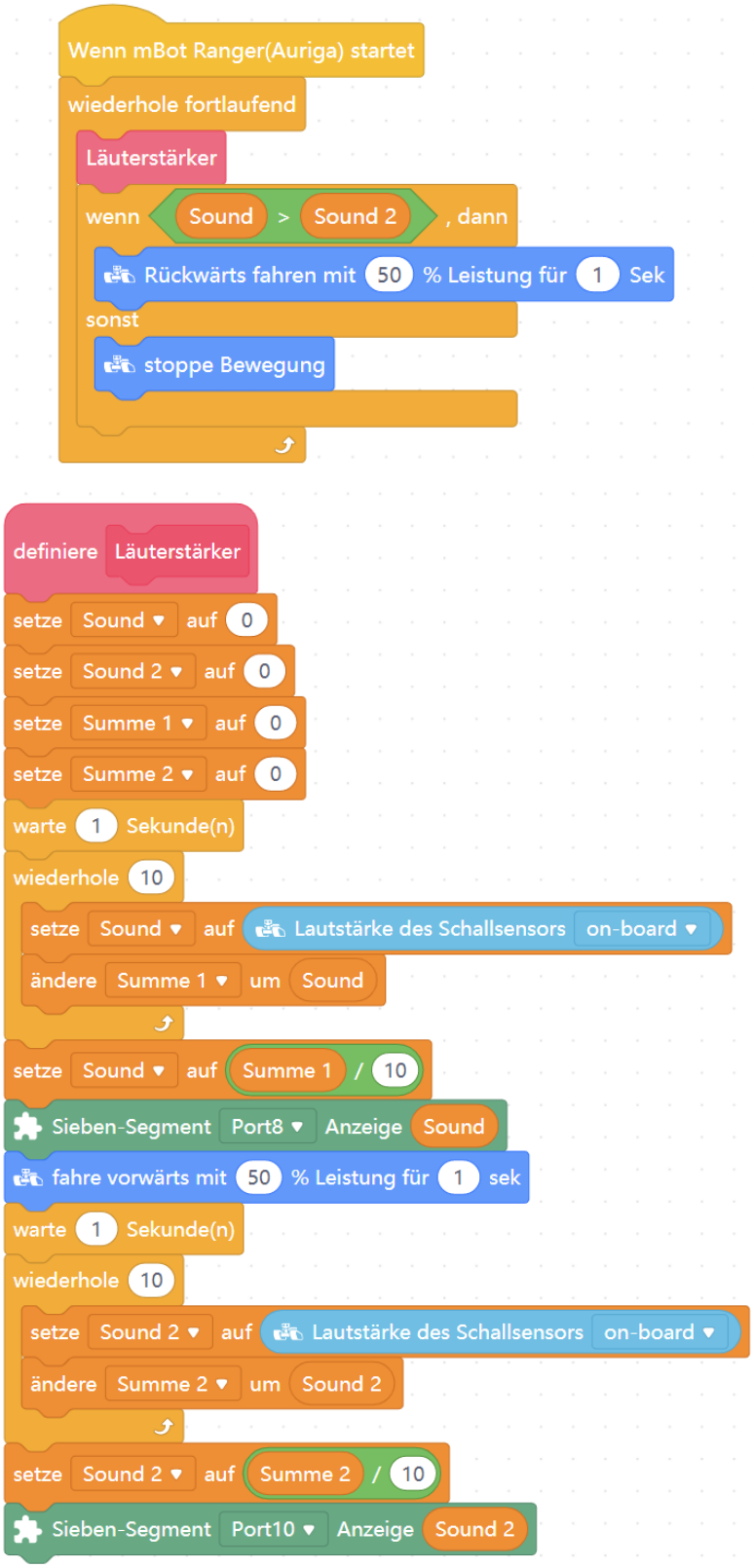
1,Hindernisse ausweichung



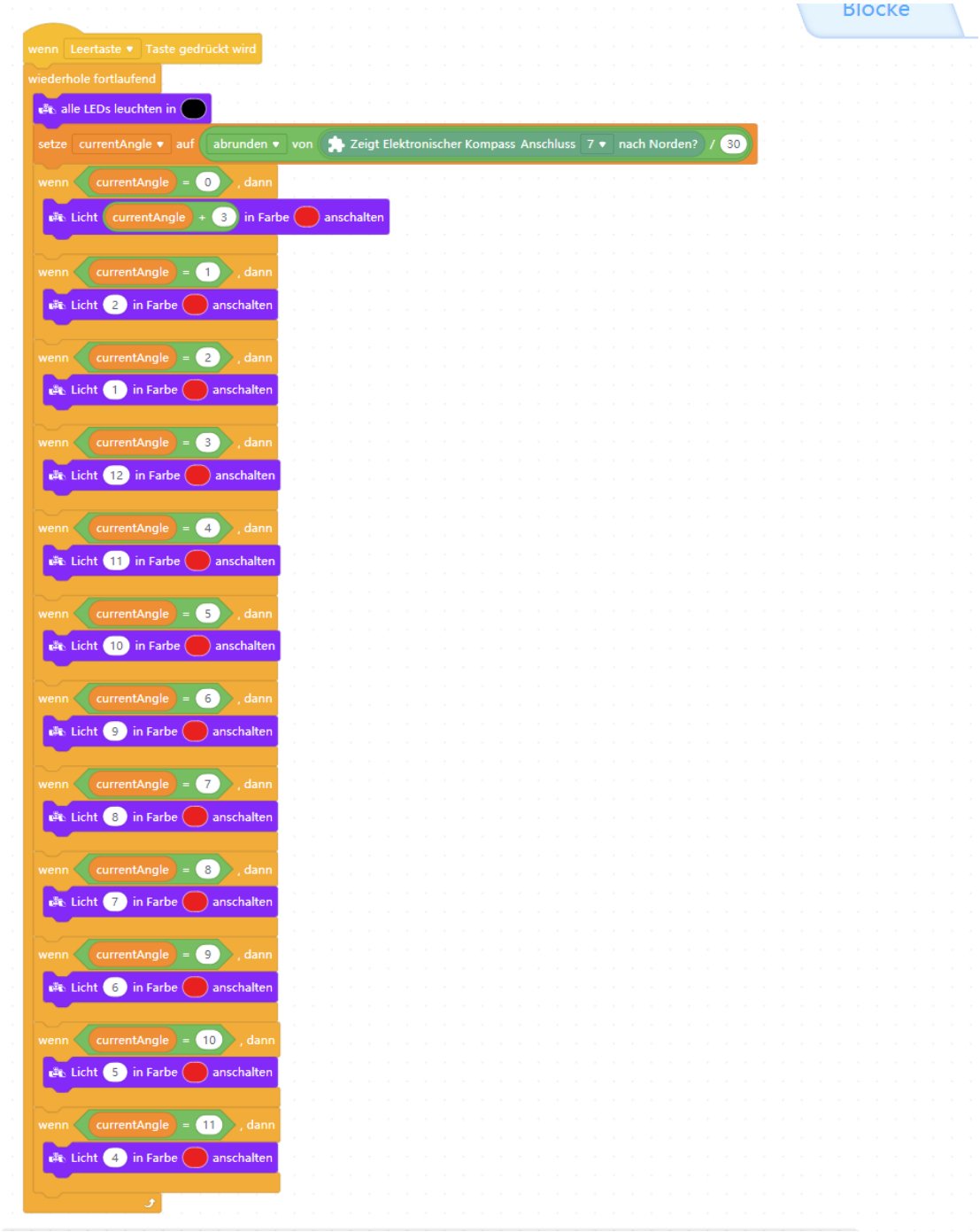
2.Linien-folger



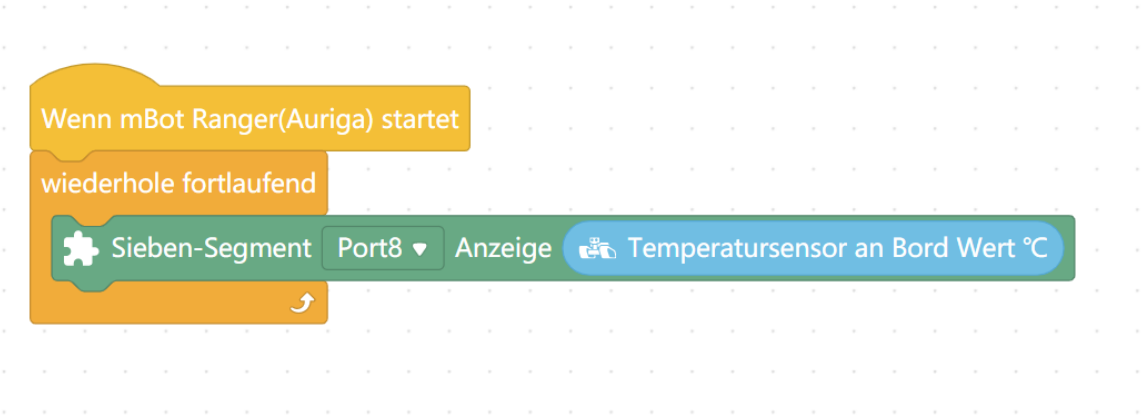
3.Fahren zur größte Lautstärke



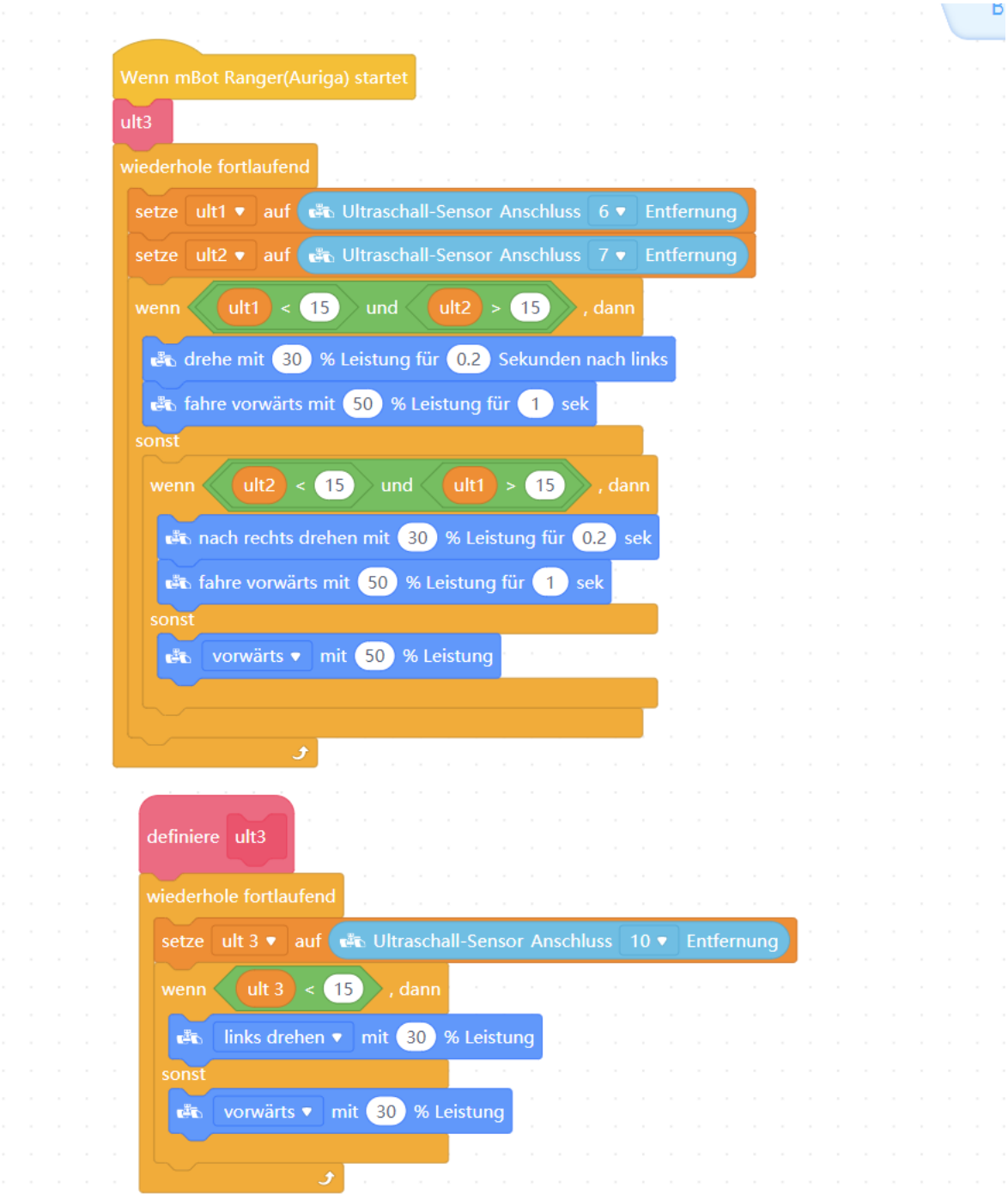
4.Anzeigen nach Norden



5.Temperatur Anzeige



6.Mindestabstand halten



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.

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

Mittweida, den 03.April 2023

Yulu Zhu