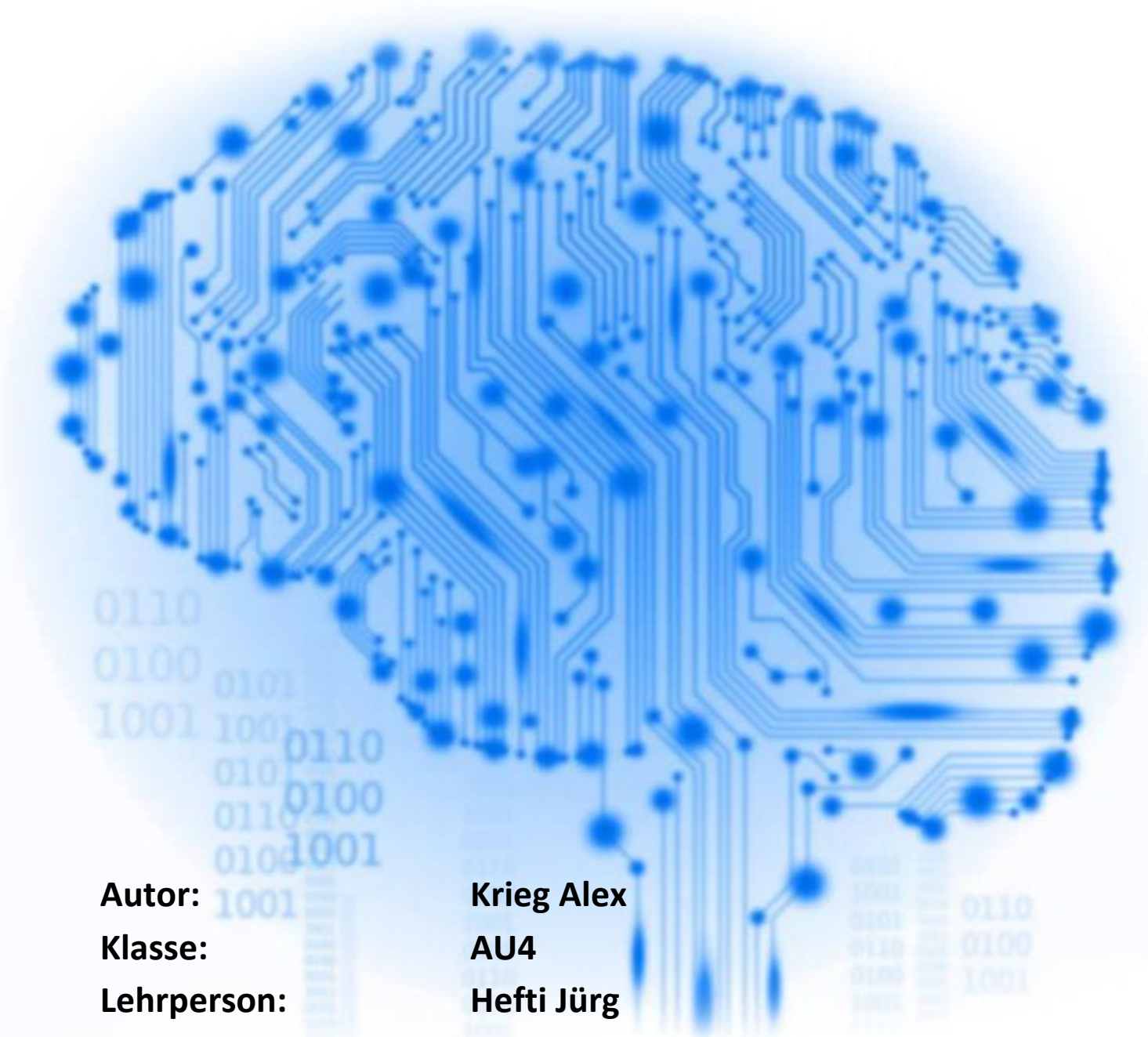
Die künstliche Intelligenz



Autor:	**Krieg Alex**
Klasse:	**AU4**
Lehrperson:	**Hefti Jürg**
Abgabedatum:	**14. Dezember 2017**
Schule:	**Gewerblich- Industrielle Berufsfachschule Ziegelbrücke**

Inhaltsverzeichnis

1 Vorwort

Es ist faszinierend zu sehen wie Lernen funktioniert und man die Lernmethode mathematisch mit Algorithmen nachbauen kann. Die Forschung ist schon sehr fortgeschritten in diesem Bereich und dieses Wissen wird schon heute angewendet. YouTube lernt darüber, was der Nutzer anschaut und gibt dementsprechend auch Suchvorschläge an. Facebook weiss ganz genau wer du bist und was auf jedem Bild das du hochlädst zu sehen ist. Ich war erstaunt als ich sah, dass ein Bild auf Facebook noch nicht geladen wurde, aber stattdessen stand ein kleiner Text z.B.: “Könnte Person mit Bart enthalten“.

Ich habe mich schon länger für die künstliche Intelligenz interessiert. Dieser Bereich der Wissenschaft fasziniert mich sehr. Dieses Wissen hilft dabei das Lernverhalten eines Lebewesens besser zu verstehen. Mir macht es Spass einen Roboter zu bauen, der wenn alles gut läuft auch noch selber lernt sich zu bewegen und mit der Umwelt interagiert. Ich werde einen Roboter bauen, der sich in einer Umgebung selbständig orientieren kann und Hindernissen ausweicht. Ich werde den Aufbau eines neuronalen Netzes erklären und den Lernalgorithmus, den ich für den Roboter verwende erklären. Es gibt mehrere Herausforderungen. Zum Einen muss der Roboter genügend früh funktionstüchtig sein, damit er ausreichend Zeit hat zu lernen. Das technische Equipment benötigt hierzu ausreichend Rechenleistung um das Lernen zu bewältigen.

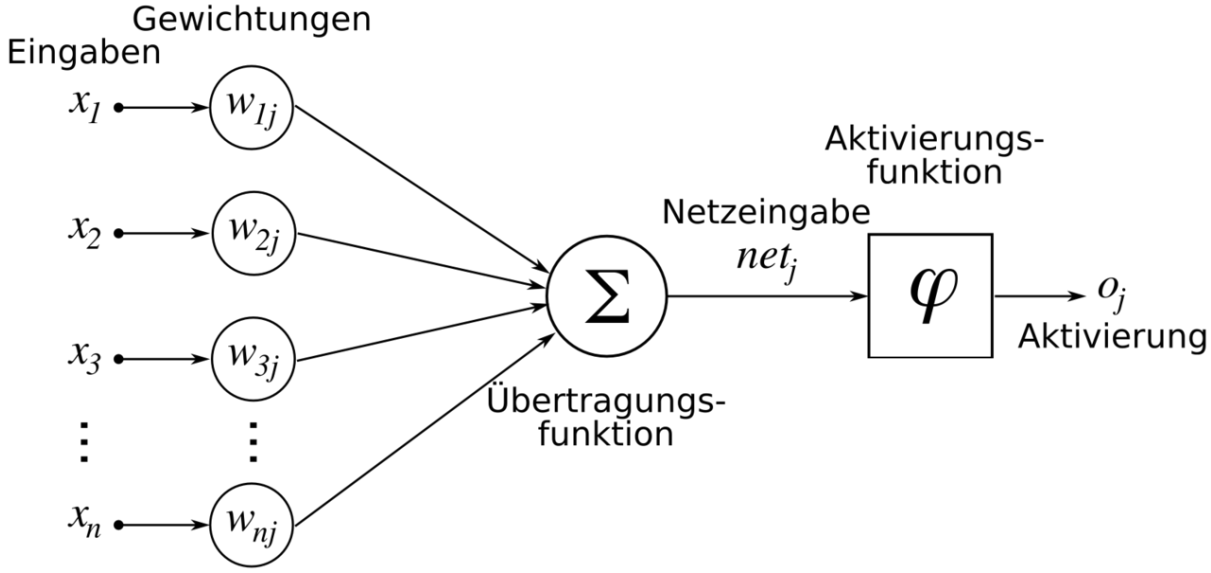
2 Zielsetzung

- Effizientes Rechnen sicherstellen, wenn möglich auf der GPU (Graphical Processing Unit) für mehr Rechenleistung.
- Ein übersichtliches und strukturiertes Programm schreiben.
- Eine Visualisierung des neuronalen Netzes entwickeln.
- Ein funktionstüchtigen Roboter bauen.
- Das neuronale Netz gut trainieren.
- Ein Interview mit Guido Schuster durchführen.

3 Grundstruktur eines neuronalen Netzes

3.1 Das Neuron

3.1.1 Formel



Dieses Bild stellt den Aufbau eines Neurons dar. Das Neuron ist der wichtigste Teil einer KI (Künstliche Intelligenz). In einem Neuron werden mehrere Input Signale zu einem Output Signal verarbeitet. Dies wird wie folgt gemacht:

$$O_j = \varphi\left(\sum_i x_i w_{ij}\right)$$

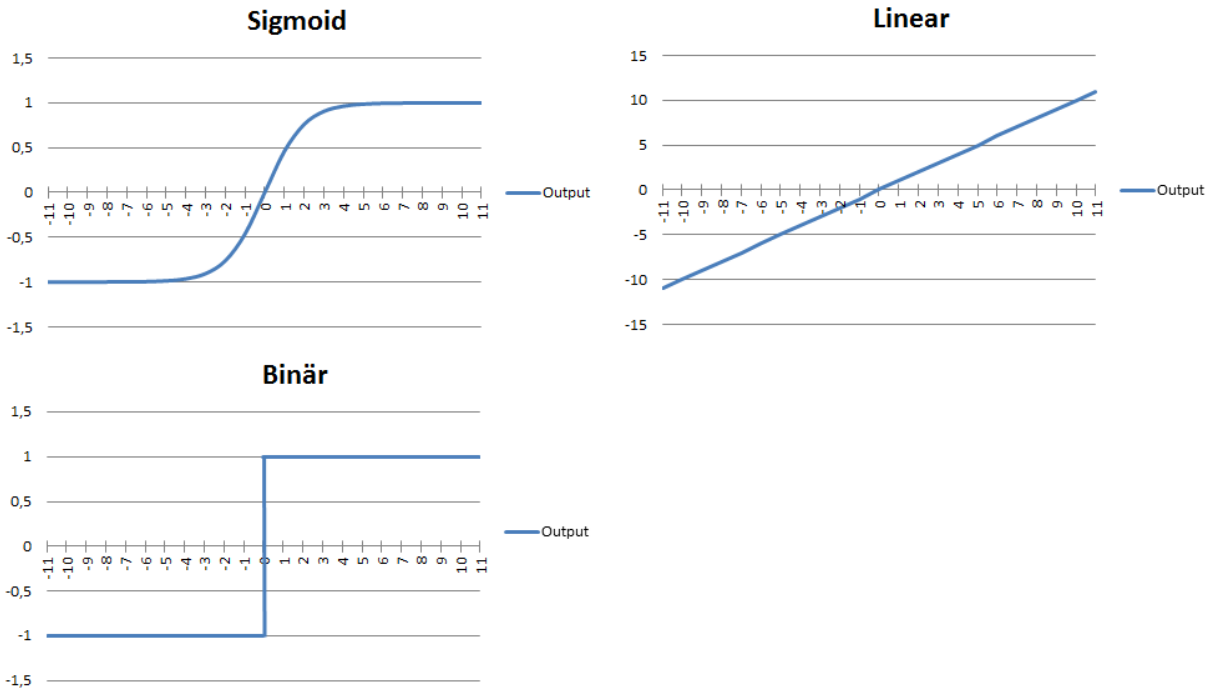
O_j	-->	Output vom Neuron j.
φ	-->	Aktivierungsfunktion, siehe Kapitel Aktivierungsfunktion
X_i	-->	Eingangs Signal i.
W_{ij}	-->	Gewichtung der Verbindung vom Eingangs Signal i vom Neuron j.

Erklärung:

Oj als Ergebnis des Neurons j berechnet sich wie folgt:
Man nimmt jedes Eingangs Signal und multipliziert dieses mit dessen Gewichtung (Gewichtung ist die Stärke der Verbindung).
Wenn man alle Multiplikationen ausgeführt hat, werden alle miteinander addiert.
Das erhaltene Ergebnis nennt man Netzinput.
Der Netzinput wird der Aktivierungsfunktion übergeben, diese Funktion berechnet den Output des Neurons.

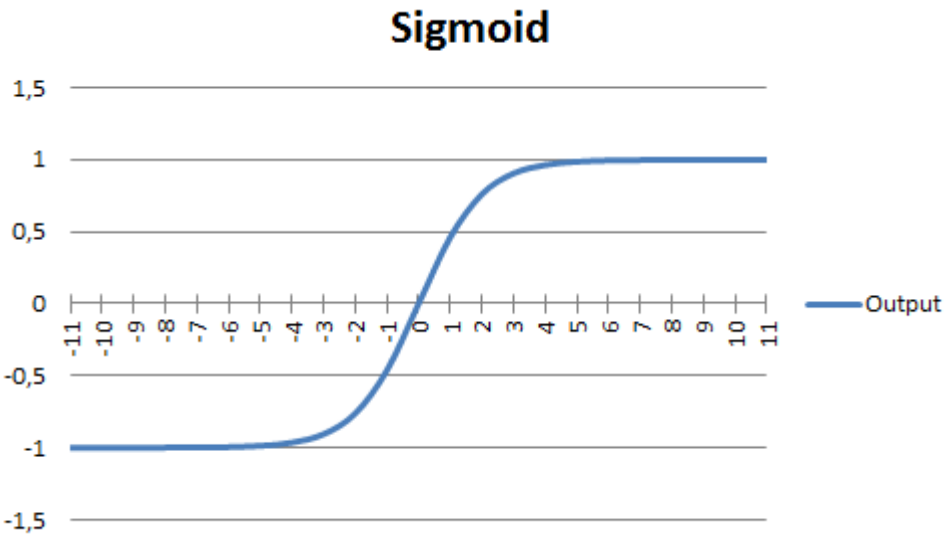
3.1.2 Aktivierungsfunktion

Es gibt verschiedene Aktivierungsfunktionen, hier einige Möglichkeiten:



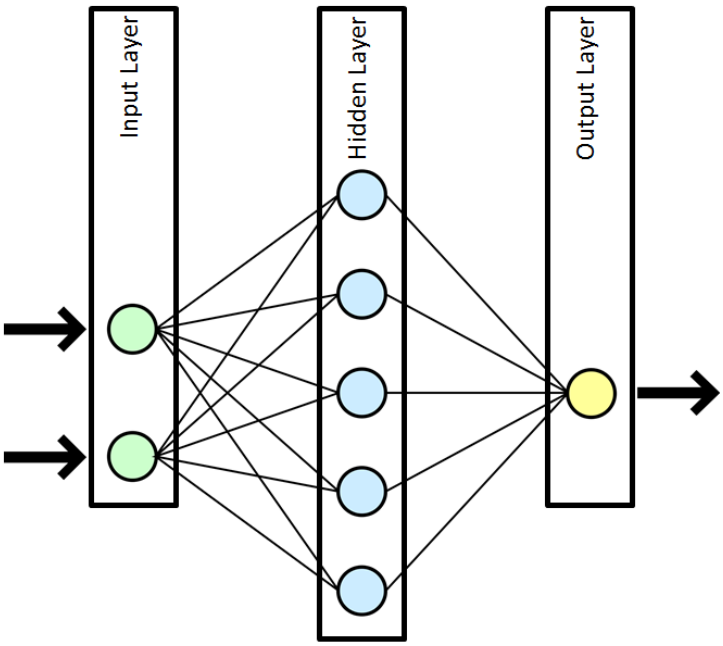
Die X-Achse (Horizontale) stellt den Netzinput dar. Die Y-Achse (Vertikale) stellt den Output dar.

In meiner Arbeit wird die Sigmoid Funktion verwendet. Im Bild unten kann man diese Funktion sehen. Wenn der Netzinput 1 beträgt, ist der Output 0.5 und wenn der Netzinput -10 ist liegt am Output -1 an. Diese Funktion ermöglicht die Begrenzung der Signale zwischen -1 und +1, da ansonsten die Zahlen zu gross werden könnten. Ein weiterer Grund für diese Funktion ist, dass sie keine abrupten Änderungen aufweist.



3.2 Die Vernetzung

Das unten gezeigte Bild stellt ein neuronales Netz dar. Es besteht aus vernetzten Neuronen, die miteinander verbunden sind. Es gibt 3 Ebenen (Layer). Der erste Layer ist der Input Layer, die darin enthaltenen Neuronen sind keine richtigen Neuronen. Das liegt daran, weil sie nur die Eingangssignale speichern und nichts rechnen. Wenn man das Netz ansteuert bekommt man direkten Zugriff auf diese Neuronen um den Übergabewert darin zu speichern. Im Input Layer kann es y beliebige Neuronen haben. Der zweite Layer ist der Hidden Layer. Seine Eigenheit liegt darin, dass er nicht nur aus einem Layer bestehen muss, wie im Bild, sondern aus x beliebigen mit y beliebigen Neuronen. Der letzte Layer ist der Output Layer, und dieser gibt es nur in einfacher Ausführung. Der Output Layer muss genau so viele Neuronen enthalten, die zur Lösung der Problematik erforderlich sind. Am Ausgang der Outputneuronen kann man die vom Netz errechneten Werte ablesen.



Als Formel ausgedrückt sieht das Netz so aus:

$$output_y = \varphi(\sum_{H_y} \varphi(\sum_{i_y} i_y * w_{I_y H_y})_y * w_{H_y O_y})$$

Iy:	Jedes Neuron vom Input Layer in y Richtung.
Hy:	Jedes Neuron vom Hidden Layer in y Richtung.
Oy:	Jedes Neuron vom Output Layer.
W_{IyHy}:	Das Gewicht zwischen den Neuronen Iy und Hy.
W_{HyOy}:	Das Gewicht zwischen den Neuronen Hy und Oy.
φ:	Aktivierungsfunktion.
$output_y$:	Output von jedem Output Neuron Oy.

4 Lernalgorithmus

In meiner Arbeit habe ich den genetischen Lernalgorithmus verwendet, weil er den grossen Vorteil hat, dass es keine Lernparameter mit den entsprechenden richtigen Resultaten benötigt. Diese bräuchte man bei der Backpropagation*. Was heisst das jetzt? Im Fall der an das Netz gestellten Aufgabe kann man nicht eindeutig sagen was richtig und was falsch ist. Daher ist es praktisch unmöglich eine richtige Lösung bereit zu stellen. Mit dem genetischen Lernalgorithmus muss man dem neuronalen Netz nur kommunizieren wie gut es die Aufgabe gelöst hat. Dieser Art von Lernen sagt man auch unsupervised learning oder unüberwachtes Lernen.

*Im Glossar beschrieben

4.1 Definition Lernen

„Aus lernpsychologischer Sicht wird Lernen als ein Prozess der relativ stabilen Veränderung des Verhaltens, Denkens oder Fühlens aufgrund von Erfahrung oder neu gewonnenen Einsichten und des Verständnisses (verarbeiteter Wahrnehmung der Umwelt oder Bewusstwerdung eigener Regungen) aufgefasst.
Die Fähigkeit zu lernen ist für Mensch und Tier eine Grundvoraussetzung dafür, sich den Gegebenheiten des Lebens und der Umwelt anpassen zu können, darin sinnvoll zu agieren und sie gegebenenfalls im eigenen Interesse zu verändern. So ist für den Menschen die Fähigkeit zu lernen auch eine Voraussetzung für ein reflektiertes Verhältnis zu sich, zu den anderen und zur Welt. Die Resultate des Lernprozesses sind nicht immer von den Lernenden in Worte fassbar (implizites Wissen) oder eindeutig messbar.“ Zitat: https://de.wikipedia.org/wiki/Lernen

Die Aufgabe von einem künstlichen neuronalen Netz ist dieselbe wie oben beschrieben. Es lernt indem es bestimmte Muster erkennt und anhand dieser Muster reagiert. Das Wissen wird weitergegeben und verbessert. So kann sich eine KI auch in einer noch unbekannten Umgebung zurechtfinden.

Verwirrt und hilflos ist eine KI erst dann, wenn die neue Umgebung nicht den Regeln der vorherigen Umgebung entspricht. Einfach ausgedrückt ist es schwerer sich in einer stark veränderten oder ändernden Umwelt zurechtzufinden. Wie würden Sie reagieren, wenn Sie morgens aufwachen und sehen, dass alle Bäume vom Himmel herunter wachsen und nicht aus dem Boden wie gewohnt. Sie müssen nun auch lernen, wie man die Äpfel von solchen Bäumen pflücken kann.

4.2 Schwierigkeit

Um sicherzustellen, dass das Netz auch richtig bewertet wird muss man eine Formel entwickeln. Das schwierige daran ist es alle relevanten Faktoren einzubeziehen. Es gibt keine richtige oder falsche Bewertungsformel, aber je besser sie ist desto genauere Lernergebnisse bekommt man. Als Beispiel kann angeführt werden, wenn z.B. ein vom Netz gesteuertes Fahrzeug eine möglichst grosse Strecke zurücklegen soll, kann es gut sein, dass das Fahrzeug immer im Kreis fährt. Der Anwender möchte aber von der KI, dass sie von A nach B kommt oder sich möglichst weit weg von A entfernt. Dies bedeutet, dass nicht einfach die Strecke zur Bewertung herangezogen werden kann. Man kann auch Regeln und Bestrafungen definieren, indem man dem Fahrzeug eine vorgegebene Zeit gibt, in der es so weit wie möglich kommen soll. Wenn das Fahrzeug die zu erreichende Strecke rückwärts fährt, kann man den Test auch abbrechen und dem Netz eine schlechte Bewertung geben. Grundsätzlich lernt das neuronale Netz nur die Mindestanforderungen die durch die Bewertungsformel bestimmt sind.

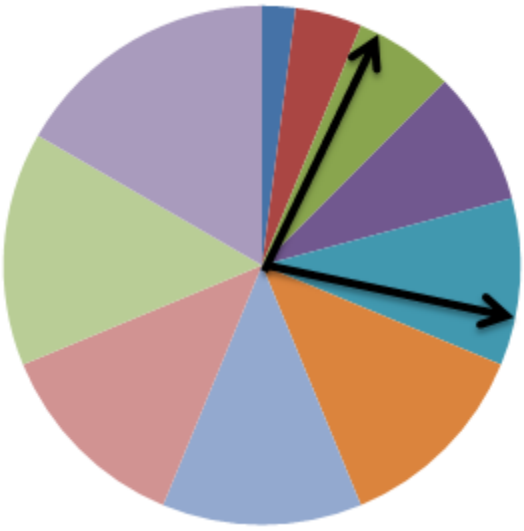
4.3 Ablauf

Um den Lernalgorithmus anwenden zu können benötigt man mehrere Instanzen von neuronalen Netzen. Der Grund dafür ist einfach, denn ein Lebewesen* benötigt Artgenossen um sich fortpflanzen und entwickeln zu können. Man kann selber entscheiden wie viele Lebewesen erstellt werden sollen. Es sollte bedacht werden, dass wenn es zu wenige sind (<30), auch kein grosser Lernfortschritt beobachtet werden kann. In der Regel erstellt man 100 Lebewesen. Je mehr es sind desto rechenintensiver wird der gesamte Prozess.

Die Lebewesen werden nun getestet, indem sie ihr Problem lösen sollen. Dies steht in direkter Abhängigkeit der gestellten Aufgabe. Der Anwender muss der Aufgabe entsprechend eine Formel bereitstellen, die während dem Testlauf für jedes Lebewesen einzeln eine Bewertung abgibt, wie gut sie sind.

*Im Glossar beschrieben

4.3.1 Selektion



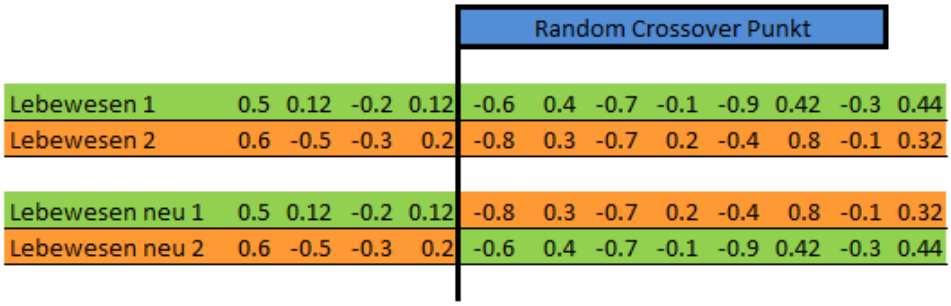
Nach dem Testlauf hat jedes Lebewesen seine Bewertung und mit dieser werden nun zwei Lebewesen selektiert.

Dies funktioniert so: Zuerst wird jede Bewertung zusammengezählt, das Ergebnis ist der Umfang des Kreises (Rechts). Das entspricht 100% Jetzt weiss man auch wie gross der Anteil jeder Bewertung ist. Anschliessend wird eine zufällige Zahl zwischen 0 und der Summe aller Bewertungen generiert (Pfeil1). Die Spitze des Pfeils entspricht der generierten Zahl. Diese befindet sich auf einem Kreisausschnitt einer Bewertung eines Lebewesens. Das Programm ermittelt nun welcher Bewertung dies entspricht. Wenn dies erledigt ist, wird ein zweiter Pfeil generiert. Hier wird noch zusätzlich dafür gesorgt, dass automatisch ein neuer Wert für den zweiten Pfeil generiert wird, wenn beide Pfeile auf denselben Kreisausschnitt zeigen. Das passiert so lange bis beide Pfeile nicht auf denselben Ausschnitt zeigen. Warum das Ganze? Nun ja man weiss doch das es keine gute Kombination ist sich mit sich selber zu Paaren.

Mit dieser Selektion wird sichergestellt, dass diejenigen, die eine hohe Bewertung haben auch eine höhere Chance haben ausgesucht zu werden. So bleiben die guten, starken Lebewesen und die schwachen sterben aus.

4.3.2 Crossover

Das Crossover ist die Paarung der Lebewesen. Diese findet nicht ganz gleich wie in der Natur statt aber das Prinzip ist dasselbe.

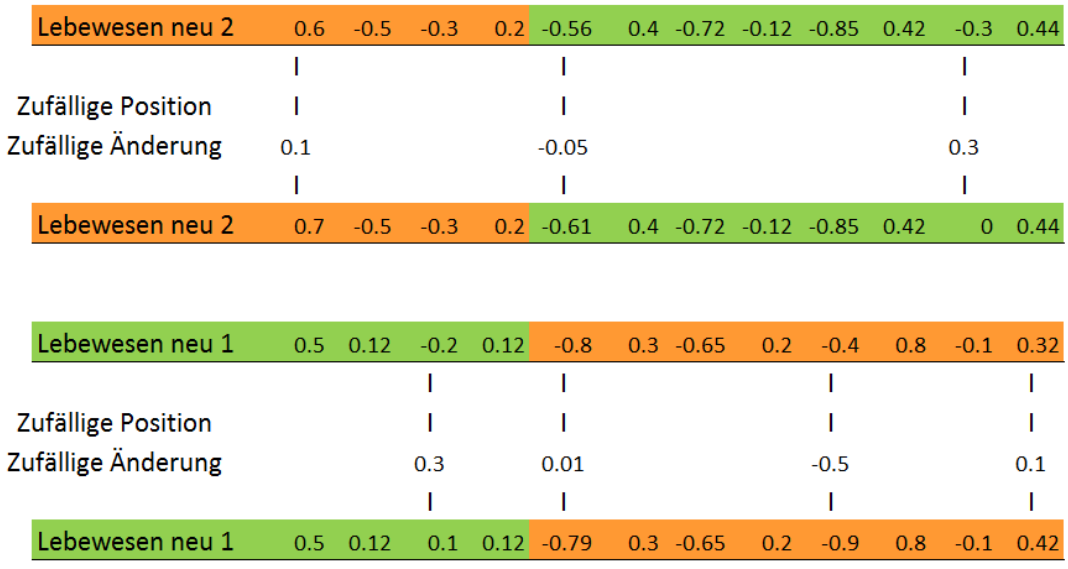


Für das Crossover werden die Genome (alle Gewichtungen/Stärken der Verbindungen zwischen den Neuronen) der selektierten Lebewesen miteinander gekreuzt. Das geschieht wie folgt:

Als erstes wird eine zufällige Zahl zwischen 1 und der Genomgrösse-1 generiert. Diese Zahl entspricht dem Random Crossover Punkt. Wie im Bild zu sehen werden die Genome so untereinander getauscht. Die neuen Genome werden zwischengespeichert, weil dieser ganze Prozess mit „Selektion“, „Crossover“ und „Mutation“ so oft durchgeführt werde muss wie die halbe Anzahl der Lebewesen ist. Es kann sein, dass ein Lebewesen mehrfach selektioniert wird. Man will nicht mit den geänderten Genomen weiter paaren sondern mit den originalen.

4.3.3 Mutation

Die Mutation sorgt dafür, dass die Lebewesen variieren können. Sie ändern sich schon durch das Crossover aber die Zahlen an sich bleiben immer die gleichen und darum wird die Mutation vorgenommen.

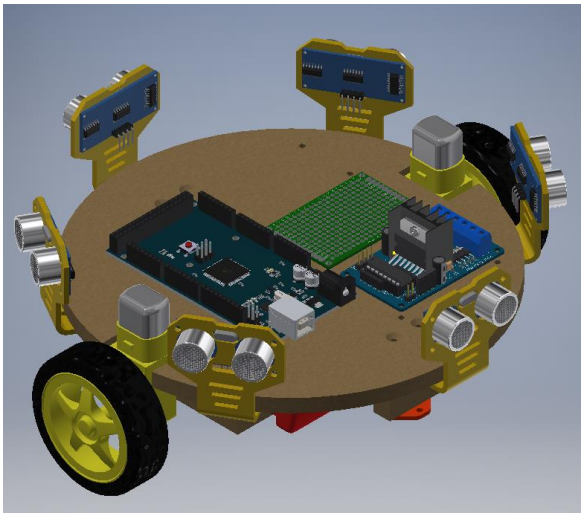


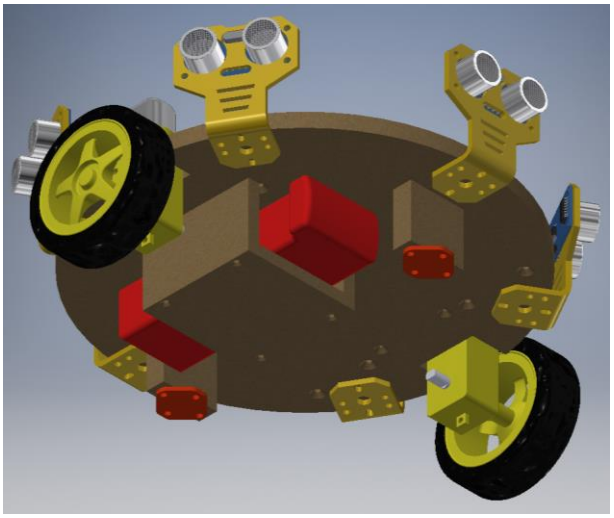
Für die Mutation werden zwei Parameter benötigt. Zum einen benötigt man eine Zahl die sagt wie gross die Chance ist, dass eine Zahl des Genoms verändert wird (Zufällige Position). Wenn diese Zahl gross ist, dann werden wenige Werte verändert. Der zweite Parameter sagt wie stark die Modifikation sein soll. Wenn diese Zahl gross ist, dann wird sich die Änderung nur schwach auswirken.

5 Der Roboter

5.1 Planung

Im ersten Schritt habe ich mir überlegt welche Aufgabe ich mir selber zumuten kann, und in dieser Zeit bewältigbar und vom Roboter schlussendlich auch beherrscht werden kann. Es war nicht einfach, weil auch einfachen Aufgaben ziemlich aufwendig werden können. Daher bin ich zur Überzeugung gekommen einen Roboter zu bauen, der sich mit Hilfe von 2 Rädern bewegen kann. Weiter soll er seine Umwelt wahrnehmen können, dafür soll er Sensoren bekommen. Die Aufgabe soll sein: Manövrieren, auf Hindernisse reagieren, Ausweichen und eine grosse Strecke fahren. Da ich die Aufgabe jetzt definiert habe geht es an die Planung für die Hardware. Um mir einen Überblick zu verschaffen, habe ich mit Autodesk Inventor mehrere Modelle gezeichnet und mich schlussendlich für das einfachste Modell entschieden.





5.2 Aufbau

5.2.1 Komponente

5.2.1.1 Stückliste

Name	Beschreibung	Stk.
Holz R=200*8mm	Roboter Körper	1
Holz 30*30*10mm	Stützen für Kugel Rad	2
Kunststoff Adapter	Adapter für Kugel Radmontage	2
Holz 60*30*6mm	Akku Halterung Wand	2
Holz 60*60*6mm	Akku Halterung Boden	1
Akku	LiPo 3S 2700mAh	1
Antrieb	Motor + Getriebe mit Rad	2
Distanz Sensor	HC-SR04 Ultrasonic mit Montagewinkel	6
Arduino	Arduino Mega 2650	1
Motor Driver	YFRobot L298N Motor Driver	1
Stromversorgung	LM2596 DC-DC-Abwertswandler	1
Print 40*60mm	Print mit Bestückung für die Signal Verteilung	1
WLAN-Modul	ESP8266 ESP-01 WLAN Modul	1

5.2.1.2 Sensoren

Es werden 6 Ultraschall Sensoren gleichmässig in einem Kreis auf dem Roboter angebracht.

Mit Hilfe dieser Sensoren erhält der Roboter einen Rundumblick. Die Sensoren sind 60° zu einander versetzt.

Der HC-SR04 Ultraschall Sensor kann eine Distanz von 2cm bis zu 4m mit einem Erfassungskegel von 15° messen. Die Ansteuerung erfolgt durch 2 Pins(Trigger, Echo).

5.2.1.3 Controller

Als Controller wird ein Arduino Mega verwendet. Der Arduino ist für das Lesen der Sensoren zuständig, zudem steuert er die Motoren an. Über die Serial Schnittstelle werden die Sensor-daten dem WLAN-Modul übergeben. Das WLAN-Modul sendet dem Arduino über die serielle Schnittstelle die Motordaten.

5.2.1.4 Aktoren

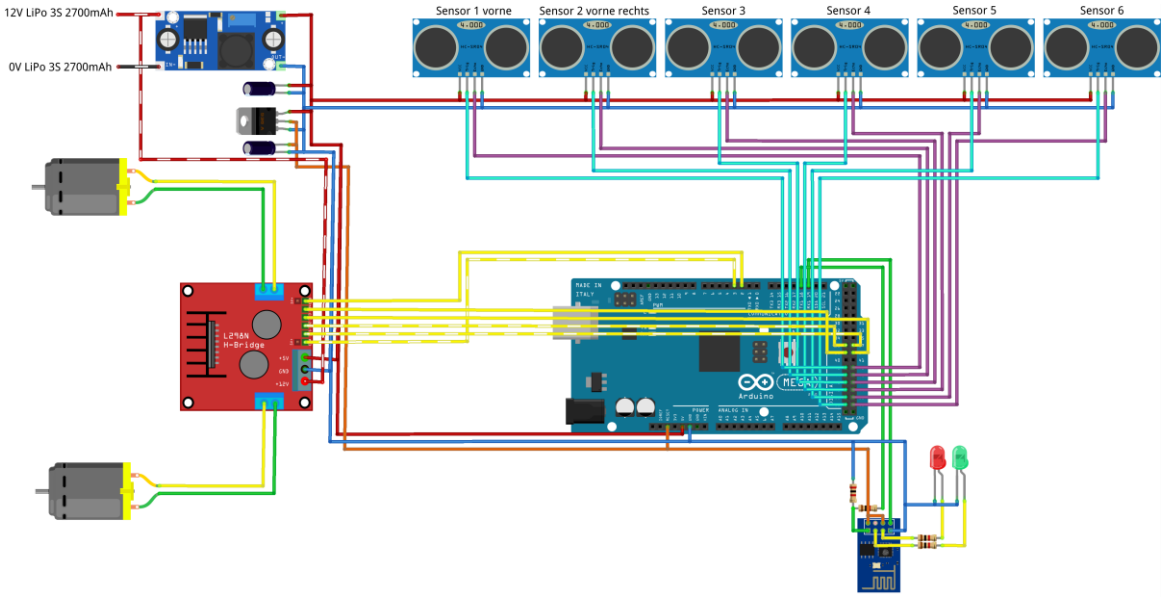
Es werden zwei DC Motoren angesteuert. Diese Motoren sind an einem Getriebe montiert. Dabei kann mit kleinerer Drehzahl aber dafür mit einer höheren Kraft auf die Räder gearbeitet werden. Die Motoren werden von dem YFRobot L298N Motor Driver Board angesteuert.

5.2.1.5 WLAN

Das WLAN-Modul ermöglicht die Verbindung zwischen Roboter und PC. Der Roboter hört nur auf die Befehle des Steuer-Pc's. Das Modul kommuniziert über Serial mit dem Arduino. Das auf dem Signal Verteiler Print aufgebrachte WLAN-Modul signalisiert mit Hilfe der zwei LED's, ob eine Verbindung besteht (Grün) oder nicht (Rot).

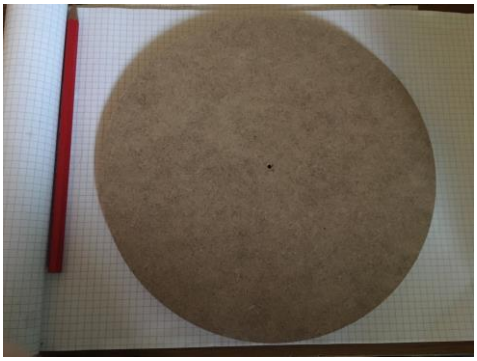
5.2.2 Verdrahtung

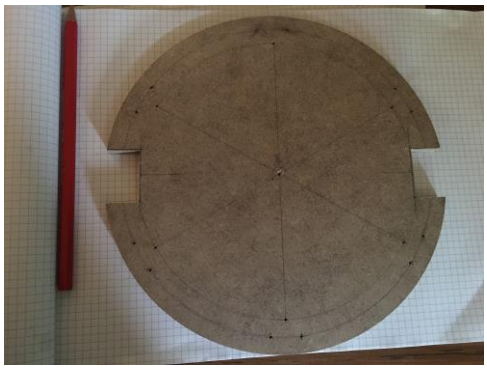
Alle Sensoren werden vom Arduino angesteuert und überwacht. Die Motoren werden über das Treiber Board vom Arduino angesteuert. Das WLAN-Modul ist über Serial Bus mit dem Arduino verbunden. Einige Bauteile auf dem Schema sind nicht dieselben wie auf dem fertigen Roboter. Der Grund dafür ist, dass nicht alle Bauteile erhältlich waren die aufgrund des Programmes (Frizing) notwendig gewesen wären.



5.3 Zusammenbau

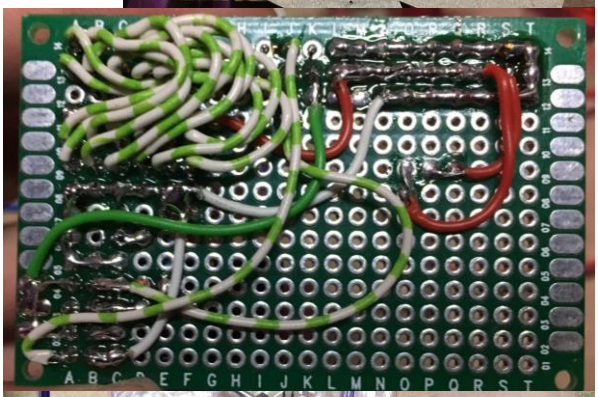
Für den Bau des Roboters habe ich zuerst die Bohrungen und Ausschnitte der Bauteile markiert. Danach ging es auch schon ans Handwerkliche. Die Teile, die ich verwendet habe wurden mir zugeschnitten von meiner Schwester Corine Krieg (Schreinerin) bereitgestellt. Danke dafür. Weiter waren noch die Bohrungen und Ausschnitte zu erstellen.

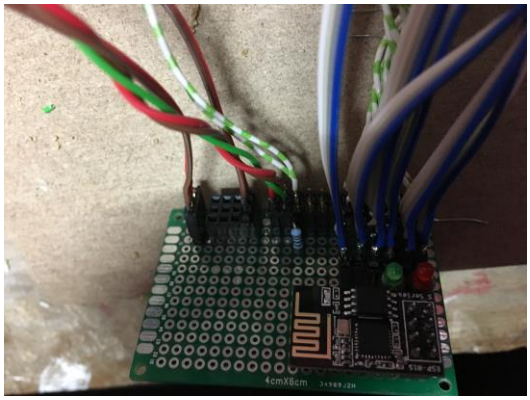


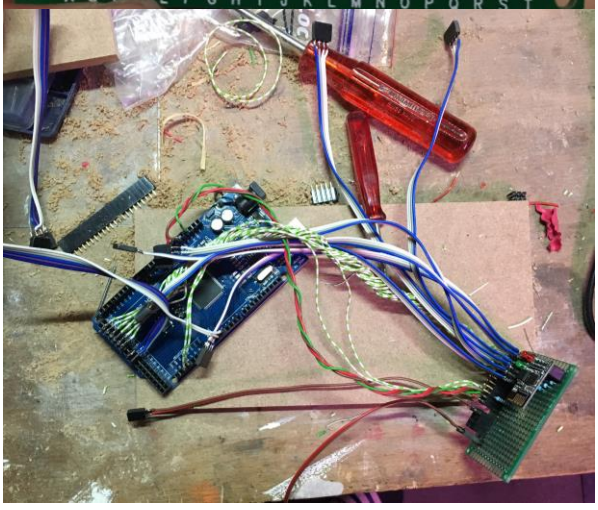


Dank dem CAD-Modell konnte ich alle Masse 1:1 auf das Holz übertragen. Ich habe viel Wert darauf gelegt, dass das CAD-Modell exakt ist. Da ich wusste, dass es immer die kleinen Details sind die man im CAD vergisst und in Realität dann fatale Folgen haben können.

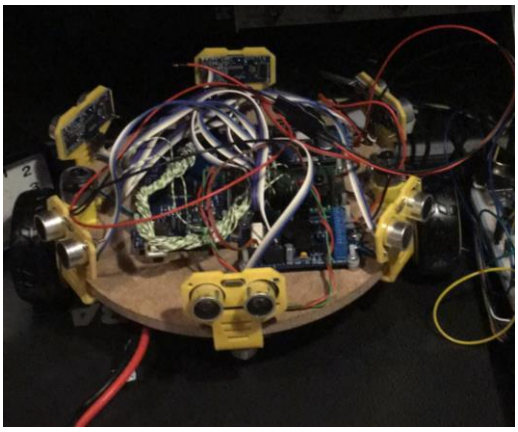
Das Gerüst war fertig nun musste ich nur noch die Elektronik bauen. Ich habe einen kleinen Print verwendet auf dem alle Sensoren eingesteckt werden. Auch das WLAN-Modul hat hier seinen Platz gefunden.





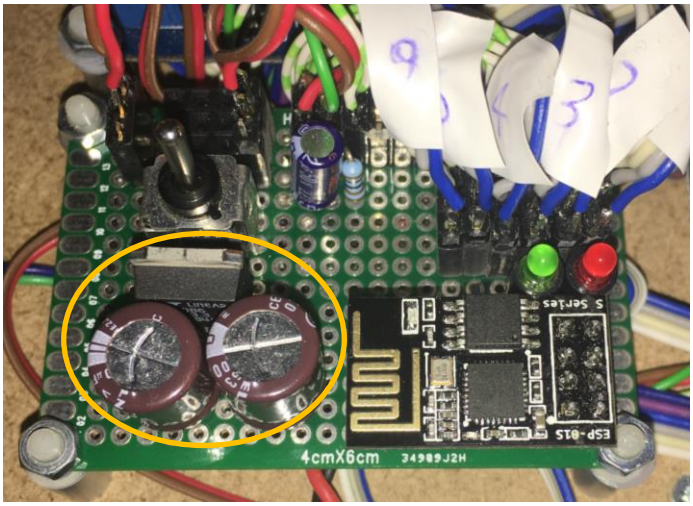


Für jeden Sensor musste ich ein Kabel herstellen um die Sensoren mit dem Print verbinden zu können. Als das ausgeführt war konnte ich den Roboter zusammensetzen. Dank der guten Vorarbeit mit dem CAD passten alle Masse von Anfang an.

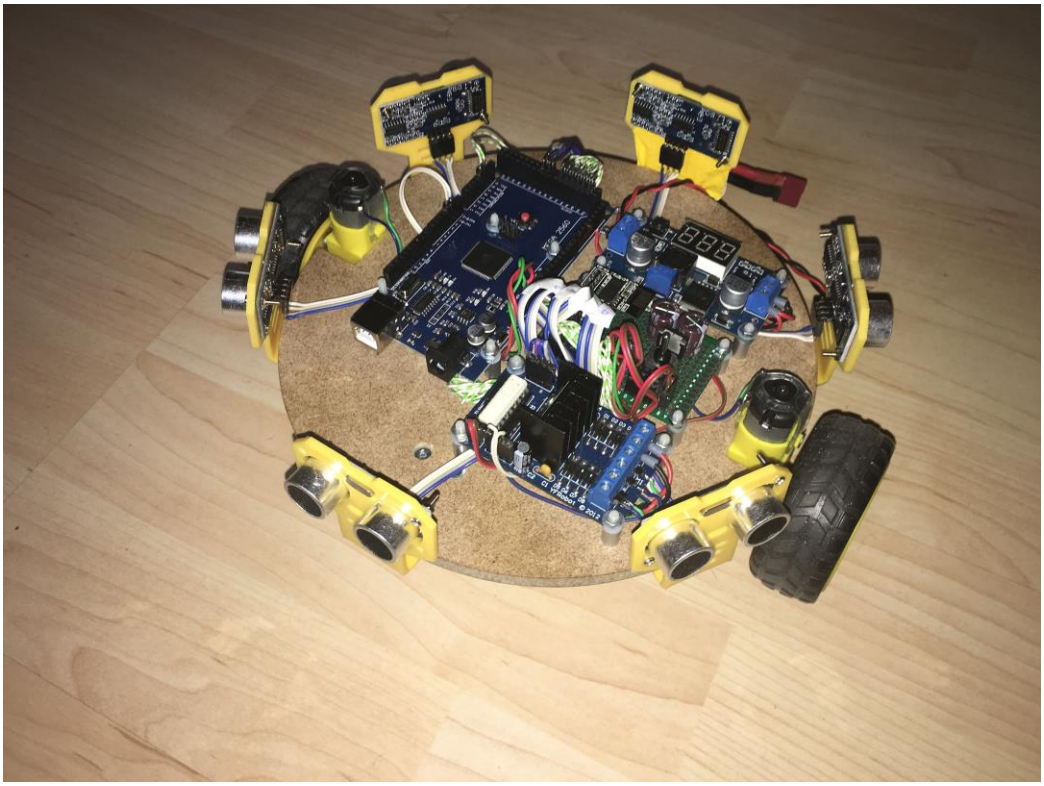


Der Roboter sah nicht schlecht aus aber zufrieden war ich noch nicht. Aus Zeitgründen beliess ich ihn in diesem Zustand.

Als die ersten Tests durchgelaufen waren hatte ich mit Kommunikationsproblemen zu kämpfen. Ich wusste nicht woran es lag, denn aus irgendwelchen Gründen hat sich das WLAN-Modul ab und zu von selbst neugestartet. Meine Vermutung lag bei der Stromversorgung. Das Modul hat durch das Senden von WLAN Signalen ab und zu einen zu hohen Stromverbrauch, den mein Arduino nicht liefern kann. Ich entschied mich den Print zu bearbeiten und eine separate Energieversorgung für das WLAN-Modul zu entwickeln. Ich habe beim Bau der Platine extra dafür gesorgt, dass alles möglichst platzsparend ist um weitere Ausweiterungen möglich zu machen. So gab es genug Platz um den 3.3V Spannungsregler (LT1086) mit Stützkondensatoren einzubauen.



Da ich schon dabei war den Roboter zu bearbeiten habe ich auch gleich noch die Verkabelung überarbeitet. Nun bin ich mit dem Roboter zufrieden wie er aussieht. Es war nicht einfach jedes Kabel unterhalb der Bauteile durchzuführen. Es hat mich einige Stunden gekostet die ganze Kabelführung neu zu machen aber mit Blick auf das Resultat hat es sich gelohnt.



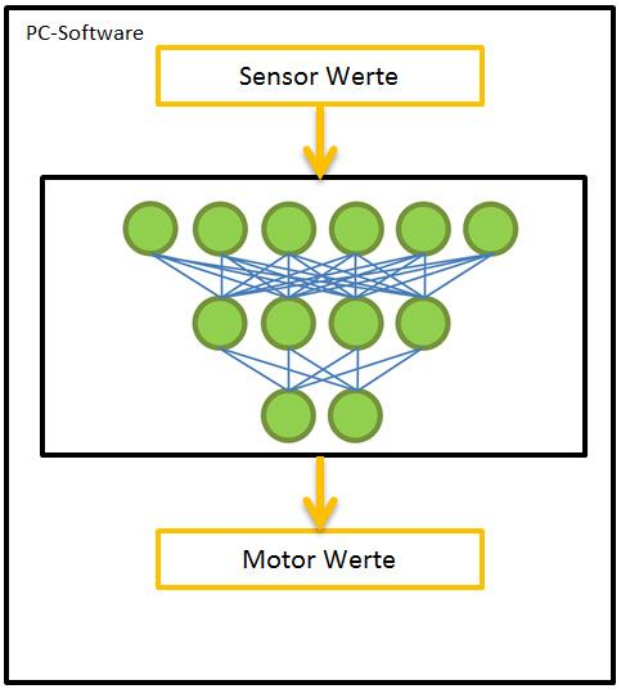
5.4 Software

5.4.1 Struktur

5.4.1.1 PC-Seitig

Der PC ist die Hauptzentrale und sorgt dafür, dass sich der Roboter überhaupt bewegt. Um dies erreichen zu können werden die Werte von den Sensoren des Roboters benötigt. Anschliessend können die Motorwerte durch das neuronale Netz errechnet und dem Roboter übergeben werden.

Die Software (PC) ist so aufgebaut, dass zuerst die Sensorwerte vom Roboter empfangen werden. Wie diese Werte empfangen werden kann man im Kapitel Kommunikation entnehmen. Hat das Programm die benötigten Daten, werden diese dem neuronalen Netz übergeben. Das Resultat bzw. die Ausgabe des Netzes können ausgelesen werden und dem Roboter zugesendet werden (siehe Kommunikation).



5.4.1.2 Roboter-Seitig

Der Roboter muss sich bewegen und auf sein Umfeld achten. Dazu besitzt der Roboter 6 Sensoren die ihm in bestimmte Richtungen die Entfernung zum nächsten Hindernis angeben.

Die Sensoren werden nacheinander angesteuert und die Entfernung in einem pro Sensor 15° Kegel gemessen. Sie werden nacheinander abgefragt, da es ansonsten zu Überschneidung der von den Sensoren gesendeten Ultraschallsignalen kommen kann. Wenn diese Werte empfangen sind, werden sie an den PC übermittelt (siehe Kommunikation).

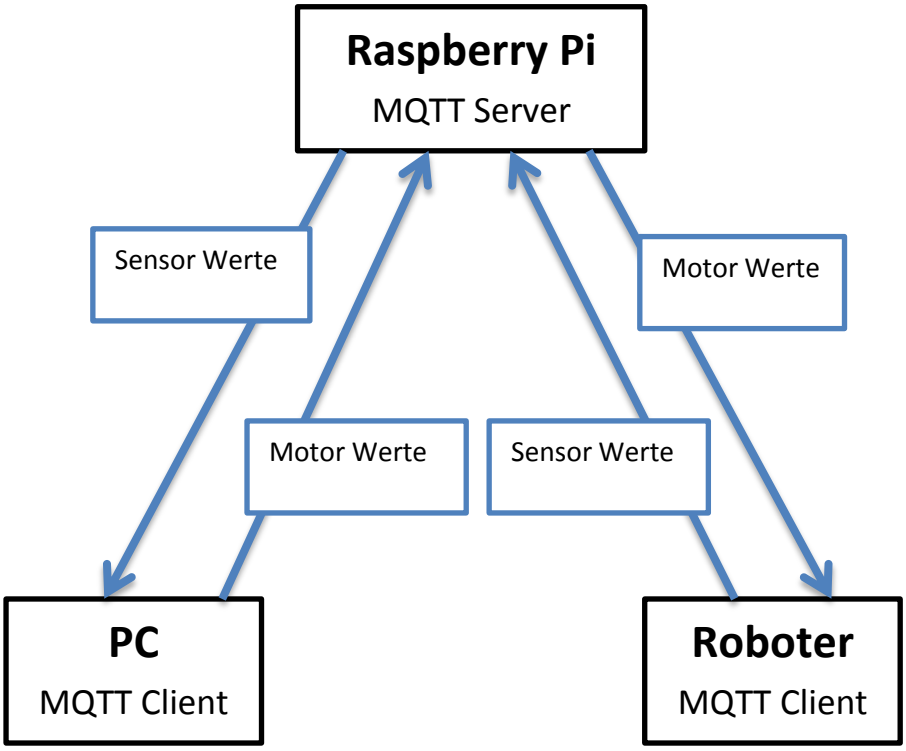
Nach der Übermittlung wird die Zeit abgewartet bis der PC seine Motor Daten dem Roboter zusendet (siehe Kommunikation). Sind die Daten empfangen, werden sie geprüft und anschliessend werden die Motoren entsprechend angesteuert.

5.4.2 Kommunikation

5.4.2.1 Überblick

PC und Roboter müssen miteinander kommunizieren können. Beide Parteien benötigen voneinander gewisse Parameter um als Ganzes arbeiten und funktionieren zu können. Durch dieses Bedürfnis entsteht das Problem, dass man nicht einfach eine Kabelverbindung zwischen Roboter und PC aufbauen kann, weil sich der Roboter frei bewegen soll und nicht durch ein Kabel gestört werden darf.

Deshalb wird eine WLAN-verbindung aufgebaut. Als Hauptknoten (Router) wird ein D-Link DIR-505 eingesetzt. In dessen Netzwerk können sich alle Teilnehmer anmelden und anschliessend miteinander kommunizieren.



5.4.2.2 Protokoll

Für eine Datenübertragung wird MQTT verwendet. Dies ermöglicht einen einfachen Datenaustausch zwischen den Teilnehmern.

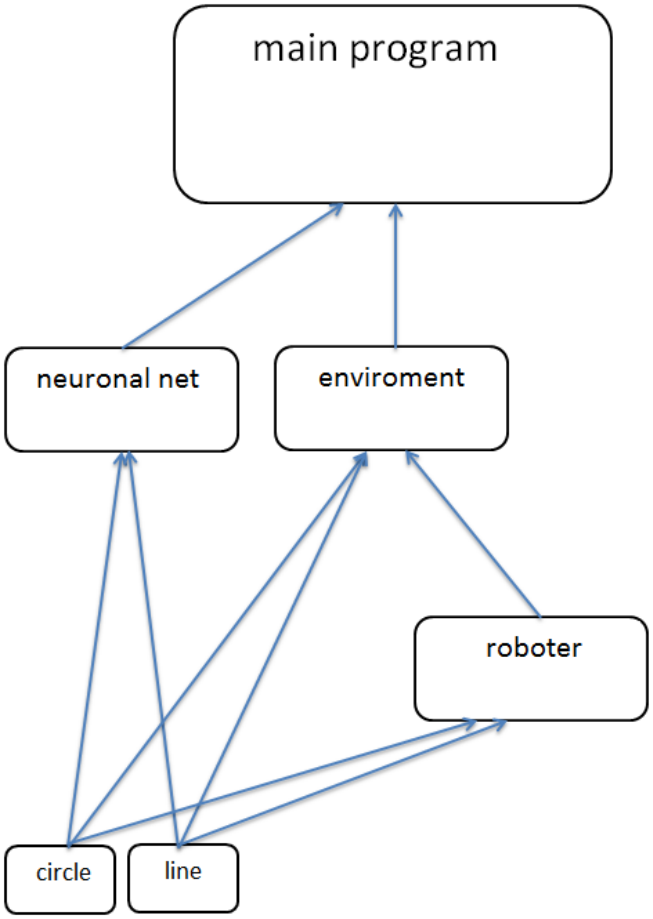
Dazu wird ein Raspberry Pi verwendet. Dieser hat den benötigten MQTT Server am Laufen. Der Roboter sendet also seine Daten nicht an den PC sondern zum Raspberry Pi. Der PC kann sich die gewünschten Daten dann beim Raspberry Pi abholen. Der Raspberry Pi ist ein kleiner Computer mit einem einfachen Linux drauf.

6 Simulation

Da Lernen ein zeitaufwändiger Prozess ist, habe ich mich dafür entschieden eine Simulation zu programmieren. Diese Software soll so realistisch wie möglich sein.

6.1 Aufbau

Das Programm ist so aufgebaut, dass es mehrere Klassen* gibt. Es gibt eine Klasse für den Roboter und zum andern noch eine für das Umfeld. Es sind noch weitere Klassen aktiv, aber diese spielen nur eine untergeordnete Rolle.



Rechts im Bild sieht man die Klassenübersicht. Sie zeigt welche Klassen einander kennen und miteinander verbunden sind.

Der Roboter kennt „circle“ und „line“, weil der Körper des Roboters aus einem Kreis besteht und die Sensoren als Linien dargestellt werden. „circle“ und „line“ beherbergen Formeln, die die Kollisionspunkte untereinander berechnen können. „enviroment“ kennt den Roboter, da er diesen auf dem Spielfeld benötigt. „Circle“ und „line“ werden von „enviroment“ ebenfalls benötigt, da die Hindernisse aus Kreisen und die Spielfeldabgrenzung aus Linien bestehen. Das „main program“ kennt „neuronal net“ und „enviroment“.

Das „main program“ sorgt für die grafische Oberfläche. Einerseits für die Verbindung zwischen Roboter und neuronalem Netz aber auch dafür, dass jeder Roboter seine Bewertung bekommt.

Das „enviroment“ übernimmt alle physikalischen Berechnungen für die Kollisionen. Im Vergleich zur echten Welt hat das „enviroment“ die Rolle der Physik und Umwelt. Wir Menschen (hier der Roboter) müssen uns nicht darum kümmern, dass wir nicht durch die Wand (hier die Hindernisse) gehen können oder wir müssen auch nicht berechnen, wie weit ein Objekt von uns entfernt ist.

Das „main program“ beinhaltet die Intelligenz und sorgt dafür, dass der Roboter seinen Bestimmungszweck mit dem Ausführen von kollisionsfreiem Bewegen erfüllen kann.

*Im Glossar beschrieben

6.2 Ablauf

Die Schritte einer KI sind Eingabe, Verarbeitung, Ausgabe. Dies geschieht in einer endlosen Schleife, aber dazwischen werden noch andere Aufgaben erledigt.

6.2.1 Ermittlung der Netzeingaben

Bevor man dem Netz Daten übermitteln kann muss man diese bestimmen. Dazu errechnet das „enviroment“ jegliche Kollisionspunkte der Sensoren des Roboters gegenüber den Hindernissen. So kann das „main program“ den Roboter fragen wie weit er mit jedem Sensor bis zum nächsten Hindernis sehen kann. Das sind dann die Inputs oder Eingaben für das Netz. Die Inputs werden aber vor der Übergabe noch bearbeitet. Die vom Roboter kommenden Daten sind Strecken im cm-Bereich und müssen umgewandelt werden, sodass sie zwischen -1 und +1 liegen und somit für das Netz brauchbar sind:

$$input_i = 1 - \left(\left(\frac{rb_i}{1000} \right) + rand \right)$$

$input_i$ → Input für das Netz. i für jeden Input.
rb_i → Der Sensorwert vom Sensor i vom Roboter.
rand → Eine zufällige Zahl zwischen -0.05 und +0.05

Es wird noch geprüft ob der Wert $input_i$ kleiner als 0 ist. Wenn ja, dann wird er auf 0 gesetzt. Das ergibt schlussendlich eine Range von 0 bis 1 zu den Inputs des neuronalen Netzes. Der Wert rand wird dafür verwendet um ein kleines Rauschen einzubauen.

6.2.2 Netz Ausgaben aufbereiten

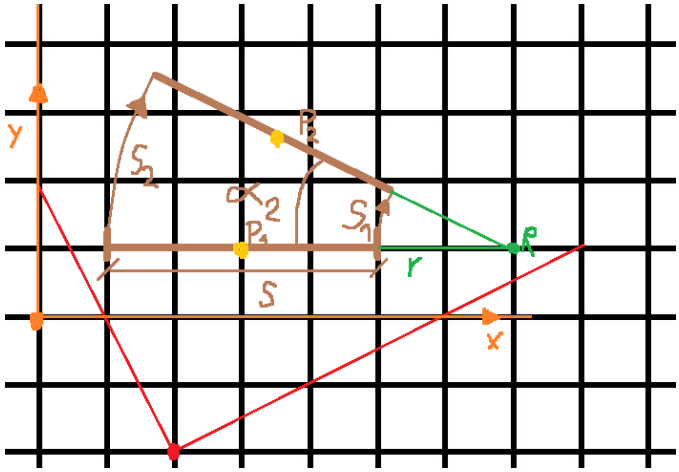
Sind die Eingänge für jedes Lebewesen gesetzt, errechnet jedes individuelle neuronale Netz die entsprechenden Outputsignale. Diese werden vom „main program“ abgefragt. Die Werte liegen zwischen -1 und 1 und müssen nun in eine für die Motoren entsprechende Form umgewandelt werden. In der Simulation wird dies nicht gemacht. Hier wurde definiert, dass die Signale gleich der zu fahrenden Strecke entsprechen. Warum erfolgt in der Simulation nicht dieselbe Betrachtungsweise der Strecke? Das Programm müsste mit dem Signal die Umfangsgeschwindigkeit der einzelnen Räder errechnen um sagen zu können wie gross die zu fahrende Strecke sein soll. Für eine Geschwindigkeit braucht man aber eine Strecke + eine Zeit und im Programm gibt es den Parameter Zeit nicht. Man kann also nicht sagen wie schnell der Roboter in der Simulation fährt. Darum werden die vom Netz errechneten Signale direkt als Strecke für jedes Rad genommen. Wenn das Signal negativ ist, dann fährt das entsprechende Rad rückwärts.

6.2.3 Ansteuerung der Räder

Für die Berechnung der neuen Position und der neuen Rotation des Roboters sind zwei Schritte notwendig. Zuerst wird mit den Strecken S1 (Motor Wert links) und S2 (Motor Wert rechts) der Winkel (α2) berechnet, wie stark sich der Roboter dreht. Dazu wird die bereits bestehende Rotation des Roboters ausser Acht gelassen und so berechnet als ob die Drehung des Roboters beim Winkel 0° beginnt.

$$\propto_2 = \frac{180(S2 - S1)}{S\pi} \qquad r = \frac{180S1}{\pi \propto}$$

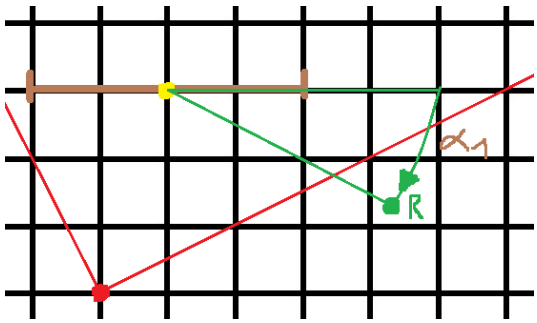
$$R = \begin{pmatrix} P_{1x} + \frac{S}{2} + r \\ P_{1y} \end{pmatrix}$$



P1: Momentane Position des Roboters.
x und y stehen für die Achsen.

Sind α2 und R berechnet, wird der Punkt R um den Punkt P1 (momentane Roboter Pos.) um den Winkel α1 (globale Rotation des Roboters) gedreht. Dies geschieht mit Hilfe einer Drehmatrix.

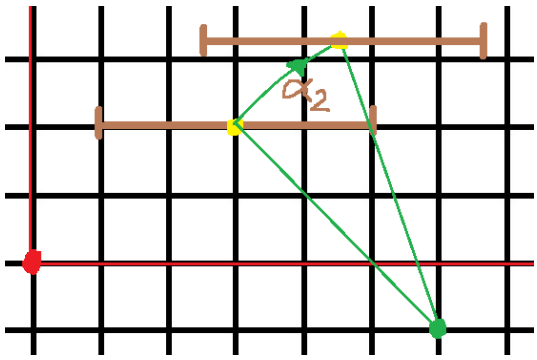
$$R_2 = \begin{pmatrix} \cos(\alpha 1)\, R_{1x} - \sin(\alpha 1)\, R_{1y} + P1_x \\ \sin(\alpha 1)\, R_{1x} + \cos(\alpha 1)\, R_{1y} + P1_y \end{pmatrix}$$



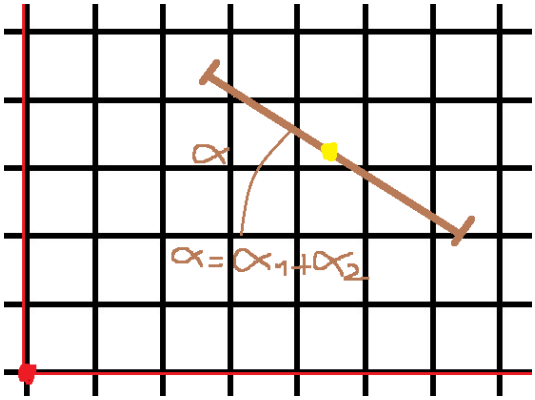
R1: Alte Position des Rotationspunktes R.
R2: Neue Position des Rotationspunktes R.

Nun kann wieder mit Hilfe der Drehmatrix die neue Position des Roboters bestimmt werden.

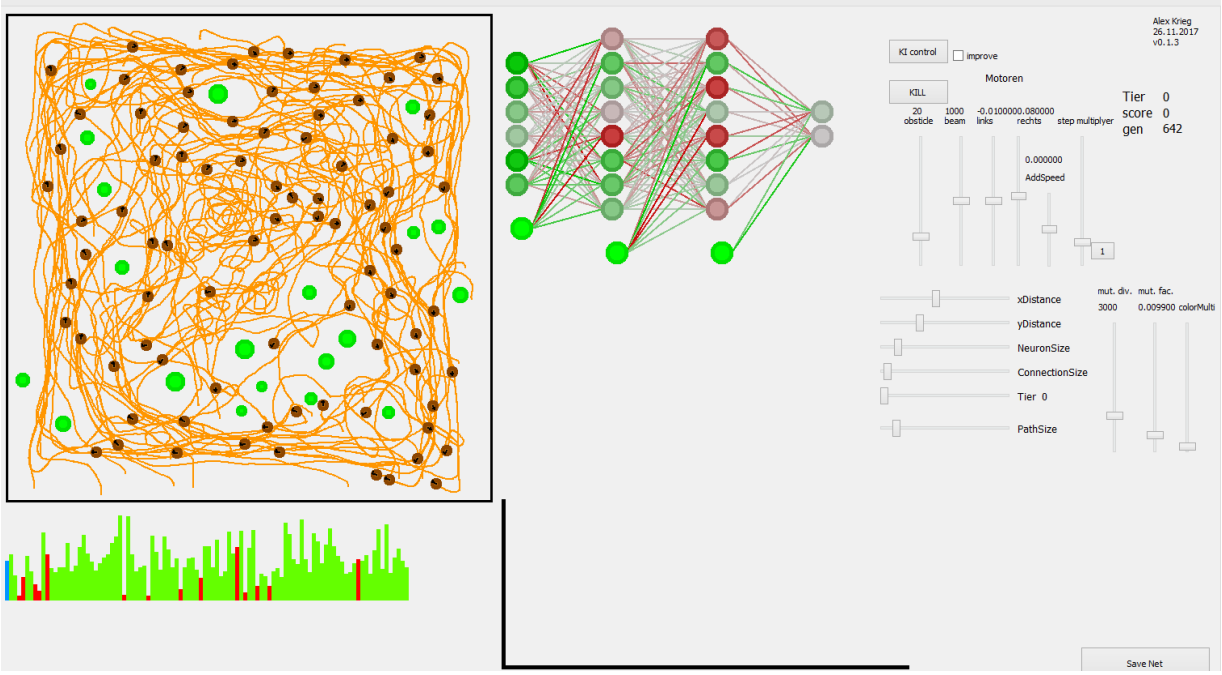
P1 wird um R um den Winkel α2 gedreht.



Die endgültige Rotation ist dann nur noch die Summe von α1 + α2. Dieser Winkel kann direkt als Rotation angewendet werden. Somit ist die Neu-Positionierung abgeschlossen.



6.3 Tests



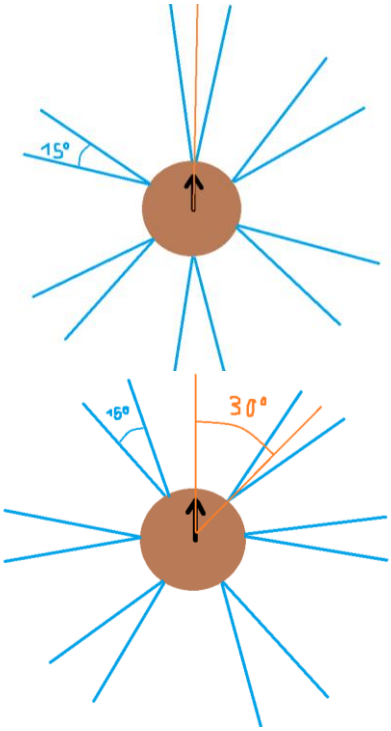
In der Simulation habe ich verschiedene Einstellungen getestet um die Beste zu finden. Es gibt viele Parameter, die zum Erfolg oder Misserfolg der Resultate führen können. Man kann z.B. mit der Dimension des neuronalen Netzes spielen. Im Moment kann ich noch nicht sagen, welche Einstellung die Beste ist. Die Bewertungsformel spielt auch eine grosse Rolle. Die habe ich in verschiedenen Varianten getestet. Die Formel, die aufgrund der guten Praktizierbarkeit zur Anwendung kommt sieht folgendermassen aus:

$$score += \left(\frac{\sum_i input}{10} + distance\right) * \frac{0.01}{1 + \text{diffAngle}}$$

score += → += bedeutet jede Runde wird etwas dazugezählt, also der score ist die schlussendliche Bewertung.
Input → jeder Input der dem Netz gegeben wird bzw. die Sensorwerte in der Form die das Netz sie sieht.
distance → die in diesem Schritt zurückgelegte Strecke (beim Rückwärtsfahren ist die Strecke negativ).
diffAngle → die Änderung des Drehwinkels, dieser Wert ist immer positiv.

6.4 Resultate

Im Laufe der Tests habe ich herausgefunden, dass die Netze mit der normalen Ausrichtung der Sensoren nicht so gute Ergebnisse lieferten wie diejenigen, deren Sensoren um 30° gedreht waren. Das bedeutet, bei meinem Roboter sind 6 Sensoren gleichmässig auf seinem runden Körper angeordnet und der erste Sensor schaut nach vorne. In der Simulation kann man die Drehung anpassen. Ich habe getestet ob der Anordnungswinkel eine Rolle spielt und er hat einen erheblichen Einfluss. Da ich den Roboter zu diesem Zeitpunkt schon gebaut hatte, konnte ich daran nichts mehr ändern. Darum versuche ich jetzt das Netz auch mit der 0° Drehung gut zu trainieren.



Ich konnte beobachten, dass sich die Roboter zu Beginn der Simulation alle im Kreis drehten. Das liegt daran, da zu Beginn alle Gewichte der neuronalen Netze zufällig gewählt werden und darum auch noch kein Wissen vorhanden ist. Um aus dieser Situation schnell raus zu kommen, habe ich die Mutationsrate und die Stärke der Änderung so eingestellt, dass sich viele Gewichtungen stark ändern. Nach ca. 10 Generationen habe ich diese Werte runtergeschraubt um den Lernprozess zu verfeinern. Nach einigen Generationen sah man erste Fortschritte. Nach ca. 100 Generationen konnte man bei allen Robotern ein typisches Verhalten feststellen. Es war die ganze Simulation davon betroffen. Die Rede ist vom Abbiegen, also dem Ändern der Richtung. Es war so, dass wenn ein Roboter 90° nach rechts abbiegen möchte konnte er das nur indem er 270° nach links abgebogen ist. Er konnte also nur links abbiegen. Das ist nicht unbedingt schlecht, es ist einfach die Art die gelernt wurde um Hindernissen auszuweichen.

7 Interview

Nach einem kurzen Telefonat mit Guido Schuster hat er sich für ein Interview über künstliche neuronale Netze bereitgestellt. Guido Schuster ist ein Dozent an der HSR und seine fachlichen Schwerpunkte sind Digitale Signal- und Bildverarbeitung. Ich fuhr am 24.November.2017 zur HSR in Rapperswil. Abgemacht war auf 9:45 Uhr. Ich konnte ihm einige Fragen stellen.

Frage 1: **Wie haben Sie sich für künstliche Intelligenz interessiert?**

Die erste Berührung mit künstlicher Intelligenz war im Jahr 1990. Ich habe mich für ein bestimmtes Gebiet der künstlichen Intelligenz spezialisiert. Mein Fachgebiet ist die Bildverarbeitung mithilfe von CNN's (Convolutional Neural Network).

Frage 2: **Was für Vorteile bietet eine künstliche Intelligenz?**

Durch eine trainierte KI gibt es weniger Aufwand für die Menschen. Viele Prozesse können einfacher und besser gesteuert werden.

Frage 3: **Wo könnte sie eingesetzt werden?**

Eine KI kann überall eingesetzt werden wo Sensoren sind. Sie kann auch zur Textverarbeitung eingesetzt werden.

Frage 4: **Gibt es Risiken einer KI?**

Risiken gibt es überall, es kommt immer auf die Art der Verwendung an. Wie bei allem muss man eine KI richtig anwenden. Beispiel: Wenn ein Terrorist ein Lastwagen (=KI) dazu verwendet Menschen umzubringen, ist der Lastwagen deshalb nicht schlecht.

Frage 5: **Kann sich eine KI verselbständigen?**

Ja, wenn man der KI die Möglichkeit dazu gibt. Gibt man einem Roboter Beine so kann er diese natürlich auch einsetzten um weg zu laufen.

Frage 6: **Was denken Sie über Vorurteile?**

Die Menschen haben Angst vor dem Unbekannten. Vorurteile werden oft auch dazu verwendet um etwas schlecht zu reden oder um über falsche Probleme zu sprechen um dann von den wahren Fakten abzulenken.

Frage 7: **Gibt es Gesetze für eine KI oder deren Forschung?**

Mir sind keine Gesetze für KI bekannt.

Ich habe das Interview mit Guido Schuster sehr spannend gefunden. Es war toll mit jemandem zu sprechen der sich in diesem Gebiet bestens auskennt und Vertiefungsmodule an der Fachhochschule zu diesem Thema hält.

8 Zusammenfassung

Eine KI (künstliche Intelligenz) besteht aus vernetzten Neuronen. Ein Neuron ist ein Knotenpunkt dem Signale zulaufen. Diese Signale erreichen das Neuron durch eine Verbindung. Diese hat eine gewisse Stärke. Je stärker eine Verbindung ist, desto stärker ist das Signal das beim Neuron ankommt. Das Signal ergibt sich aus dem Produkt der Stärke der Gewichtung und dem Signal, das zum Neuron gelangen soll. Das Neuron wiederum sendet selbst ein Signal ab. Dieses Signal ergibt sich aus der φ(Summe aller Eingangssignale * deren Verbindungsgewichte).
φ ist die Aktivierungsfunktion. Dieses Ausgangssignal gelangt über andere Verbindungen zu den nächsten Neuronen. Ich habe einen Roboter gebaut, der sich selbstständig bewegen und Hindernissen ausweichen soll. Der Roboter hat 6 Ultraschall Distanz-Sensoren und zwei Motoren. Für die Steuerung des Roboters habe ich eine KI verwendet. Das heisst, ich muss die KI nur trainieren und danach kann der Roboter die Aufgabe lösen. Um die KI trainieren zu können habe ich eine Software entwickelt, die den Roboter und sein Umfeld simulieren. Dank dieser Simulation kann der Roboter ganz bequem an einem Computer lernen. In der Simulation werden alle Berechnungen durchgeführt, die in der echten Welt wegfallen, wie z.B. die Berechnung um herauszufinden wie weit ein Sensor bis zum nächsten Hindernis sehen kann. Oder auch wie sich die Räder verhalten wenn sie vom Motor angesteuert werden.
Wenn eine trainierte KI bereit für den echten Roboter ist und das ist der Fall, wenn sich die KI in der Simulation bewährt, dann wird sie auf dem Roboter getestet. Dazu wird der Roboter auf dem Boden eingeschaltet. Er verbindet sich anschliessend automatisch mit dem vorgesehenen WLAN-Netzwerk und wartet auf seine Befehle. An einem PC starte ich eine Software, die die trainierte KI lädt. Sobald ich dem Roboter das OK gebe, sendet er über einen Server seine Sensorwerte an den PC, wobei die Werte an die KI übermittelt werden. Die von der KI errechneten Ausgaben werden über den Server an den Roboter gesendet und dieser führt dann die Signale als Fahrbefehl aus.

Am 24. November 2017 habe ich ein Interview mit Guido Schuster geführt. Es war sehr interessant, er hat mir erzählt, welchen Themen der KI er sich widmet. Ich habe ihm meine Arbeit ein wenig näher gebracht und die Lernsoftware gezeigt. Das Gespräch hat sich für mich gelohnt.

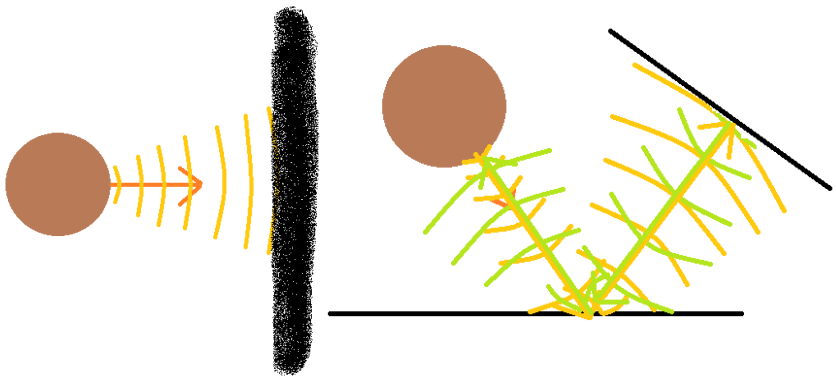
9 Fazit

Ich fand es sehr spannend dieses Projekt durchführen zu können. Ich wollte schon lange einen autonomen Roboter bauen. Ich hätte das ganze Vorhaben heute nicht meistern können, wenn ich mich nicht schon länger dafür interessiert und damit auseinandergesetzt hätte.

Es gab eine Vielzahl Probleme, die ich im Verlauf des Projektes bewältigen musste. Ich war einige Male am Boden Zerstört, aber ich habe für fast alle Hindernisse eine Lösung gefunden.

Für mich war es neu ein GUI (Graphical User Interface) basiertes Programm zu schreiben, ich musste mich einarbeiten, aber es hat sich gelohnt. Durch die Komplexität eines GUI basierten Programmes habe ich das Ziel: „Effizientes Rechnen sicherstellen, wenn möglich auf der GPU für mehr Rechenleistung“ nicht erreicht. Es ist aber nicht so schlimm, denn ich kann in der Simulation auch schon nach kurzer Zeit einen Fortschritt sehen.

Ich war erstaunt als ich den Roboter zum ersten Mal mit einem trainierten neuronalen Netz gesteuert hatte. Der Roboter hat sein Bestes gegeben und konnte grossen Hindernissen ausweichen. Leider nicht allen Hindernissen, aber da kann die KI nichts dafür. Auf dem Roboter sind Ultraschall Sensoren positioniert. Diese senden Schallwellen aus und werden von flachen Oberflächen wie mit einem Spiegel reflektiert. Bei stoffigen Hindernissen ist es so, dass gar kein Echo entsteht.



Ich hätte auch Laser-Sensoren verwenden können aber die sind teurer und trotzdem nicht immer zuverlässig. Mich stört es aber nicht so, denn es geht darum, dass diese KI in der Lage ist zu lernen und das Gelernte umzusetzen. In der Simulation kann man gut sehen, dass es funktioniert, denn da werden keine Reflektionen berechnet, sondern die direkten Kollisionspunkte.

Zum Schluss möchte ich noch den Leuten danken, die mir bei Problemen geholfen haben.

Martin Fatzer: Berechnung der Kollisionspunkte für die Software.
David Feldmann und Patrik Rothenberger: Probleme mit der seriellen Kommunikation zwischen Arduino und WLAN-Board. Auch wenn das Problem bzw. die Lösung bis zu diesem Zeitpunkt noch nicht gefunden wurde.

10 Arbeitsjournal

13.11.2017	10:15:00	12:00:00	1.75	Roboter bau
13.11.2017	13:30:00	14:00:00	0.50	Simulations Software
13.11.2017	14:30:00	17:45:00	3.25	Simulations Software
14.11.2017	07:30:00	12:00:00	4.50	MQTT Kommunikation versucht
14.11.2017	13:30:00	16:30:00	3.00	MQTT Kommunikation versucht
15.11.2017	07:30:00	12:00:00	4.50	Simulations Software
15.11.2017	13:30:00	18:10:00	4.67	Simulations Software
15.11.2017	19:00:00	23:00:00	4.00	Simulations Software
16.11.2017	07:15:00	08:45:00	1.50	Doku schreiben
16.11.2017	16:30:00	23:40:00	7.17	Simulations Software
17.11.2017	08:30:00	12:15:00	3.75	MQTT Kommunikation versucht
17.11.2017	13:45:00	18:00:00	4.25	Simulations Software
17.11.2017	19:00:00	02:30:00	7.50	Simulations Software
18.11.2017	12:00:00	00:00:00	12.00	MQTT Kommunikation und opc ua versucht
19.11.2017	13:00:00	01:30:00	12.50	Simulations Software
20.11.2017	08:00:00	12:00:00	4.00	Simulations Software
20.11.2017	13:30:00	18:00:00	4.50	Simulations Software
21.11.2017	08:00:00	12:15:00	4.25	Simulations Software
21.11.2017	13:30:00	18:00:00	4.50	Simulations Software
21.11.2017	19:00:00	02:30:00	7.50	Simulations Software
22.11.2017	19:00:00	01:00:00	6.00	Simulations Software
23.11.2017	07:15:00	08:45:00	1.50	Doku schreiben
23.11.2017	17:00:00	00:00:00	7.00	Simulations Software
24.11.2017	09:45:00	11:00:00	1.25	Interview mit Gido Schuster
25.11.2017	13:00:00	00:00:00	11.00	Simulations Software Tests
26.11.2017	12:45:00	21:00:00	8.25	Doku schreiben / Simulations Software Tests
27.11.2017	07:45:00	12:00:00	4.25	Roboter Software / Simulations Software Tests
27.11.2017	13:30:00	17:00:00	3.50	Roboter Software / Simulations Software Tests
27.11.2017	19:00:00	00:00:00	5.00	Roboter Software / Simulations Software Tests
28.11.2017	07:45:00	12:10:00	4.42	Roboter Software / Simulations Software Tests
28.11.2017	13:30:00	18:00:00	4.50	Roboter Software / Simulations Software Tests
28.11.2017	18:00:00	22:20:00	4.33	Roboter Software -Erste Fahrt / Simulations Software Tests
29.11.2017	08:20:00	12:00:00	3.67	Roboter Software -Erste Fahrt mit KI / Simulations Software Tests
29.11.2017	16:00:00	18:15:00	2.25	Roboter Software -SerialProbleme/ Simulations Software Tests
29.11.2017	19:00:00	22:00:00	3.00	Roboter Software -SerialProbleme/ Simulations Software Tests
30.11.2017	07:15:00	08:45:00	1.50	Doku schreiben
30.11.2017	17:00:00	00:00:00	7.00	Roborer Hardware verbessern / Simulations Software Tests / Fahrtests mit KI
01.12.2017	07:30:00	12:00:00	4.50	Roboter Tests
01.12.2017	13:30:00	17:00:00	3.50	Komunikationsproblem-lösungs suche
02.12.2017	14:00:00	20:30:00	6.50	Doku schreiben
07.12.2017	17:00:00	20:00:00	3.00	Doku überarbeiten
08.12.2017	08:00:00	12:00:00	4.00	Simulations Software Verbesserungen
		Gesamt	**238.33**	**h**

11 Glossar

Arduino	Ein Arduino ist ein Microcontroller, den man mit dem PC programmieren kann. Er hat Ein/Ausgänge die angesteuert werden können.
Backpropagation	Ist ein Lernalgorithmus bei dem die Lösungen für die Trainingssituationen bestehen und dem Netz übergeben werden muss. Mit der Lösung wird der Fehler berechnet und dieser rückwärts im Netz weitergegeben um die Gewichtungen anzupassen.
CNN	„Convolutional Neural Network“ Das ist ein neuronales Netz, bei dem nicht alle Neuronen mit allen im nächsten Layer verbunden sind, sondern immer in Gruppen miteinander verbunden sind. Ein CNN ist viel komplexer als normale Netze und wird zur Bilderkennung eingesetzt.
Genom	In dem Genom sind alle Informationen eines neuronalen Netzes enthalten. Mit Informationen sind die jeweiligen Gewichtungen der Verbindungen zwischen den Neuronen gemeint. Das ganze Wissen des Netzes steckt nur in diesen Zahlen.
GPU	GPU steht für Graphics Processing Unit und wird benötigt um ein Bild bei einem Computer darstellen zu können. Eine GPU ist anders aufgebaut als eine CPU (Central Processing Unit) die den Hauptprozessor eines Computers ausmacht. Die GPU ist besser für diese Art von Berechnungen gemacht.
GUI	Ein GUI ist ein „Graphical User Interface“, dies ist ein Programm, bei dem man etwas mit Bildern und Symbolen sehen kann, anstatt nur wie bei einer Konsolenanwendung ein schwarzes Fenster mit Textausgaben zu haben.
KI	KI ist eine Abkürzung und steht für “Künstliche Intelligenz“.
Klasse	C++ wird als Programmiersprache verwendet und diese ist objektorientiert. Eine Klasse ist ein Objekt und dieses kann man sich wie ein Objekt in der Realität vorstellen.
Lebewesen	Mit Lebewesen ist hier eine Instanz des neuronalen Netzes gemeint.
MQTT	MQTT ist eine Abkürzung für Message Queuing Telemetry Transport. Es ist ein Nachrichtenprotokoll.
Raspberry Pi	Ein Raspberry Pi ist ein kleiner Computer mit einer vereinfachten Form von Linux als Betriebssystem.

12 Quellenangabe

12.1 Bilder

https://de.wikipedia.org/wiki/K%C3%BCnstliches_neuronales_Netz
Titelbild (bearbeitet): Martin Defuns
Eigene

12.2 Texte

https://de.wikipedia.org/wiki/Lernen

12.3 Wissen

https://de.wikipedia.org/wiki/K%C3%BCnstliches_neuronales_Netz
Selbststudie 3/4 Jahre lang