Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

Geräuschquellenlokalisierung mit einem humanoidem Roboter

Bachelorarbeit
im Arbeitsbereich Knowledge Technology, WTM
Prof. Dr. Stefan Wermter

Department Informatik
MIN-Fakultät
Universität Hamburg

vorgelegt von
Robert Keßler
am
31. Januar 2012

Gutachter: Prof. Dr. Stefan Wermter
Dr. Cornelius Weber

Robert Keßler
Matrikelnummer: 6053843
Wurmsweg 1
20535 Hamburg

Zusammenfassung

Um eine solide Interaktion mit einem menschlichen Benutzer möglich zu machen, muss jeder Roboter in Zukunft ein umfangreiches auditorisches System bereitstellen. Neben der Spracherkennung ist dabei auch die Geräuschquellenlokalisation ein notwendiger Bestandteil. Diese Arbeit zeigt die Realisierung eines Lokalisierungsverfahrens in der horizontalen Ebene, mit einem humanoidem Roboter unter Nutzung von Cross Correlation und Künstlichen Neuronalen Netzen.
Der Roboter soll 24 verschiedene Richtungen am vollen 360° Kreis, mit einem jeweiligen Abstand von 15°, unterscheiden. Dies gelingt dem vorgestellten Verfahren mit einer Genauigkeit von bis zu 92%. Dabei treten die fehlerhaften Klassifikationen hauptsächlich zwischen benachbarten Richtungen auf.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Aus vielen Bereichen des heutigen Lebens sind technische Gerätschaften nicht mehr wegzudenken. Sei es die Waschmaschine, die Mikrowelle oder der Kühlschrank. Diese Systeme sollen vor allem eines erreichen - dem Menschen Arbeit abnehmen bzw. diese erleichtern.
Dieser Trend ist auch bei der Entwicklung von Servicerobotern zu beobachten. Ein solides Beispiel dafür findet sich im PERSES Projekt [6]. Dieses Robotersystem hat explizit die Aufgabe, in einem einfachen Einkaufsladen dem Kunden für Fragen zu Verfügung zu stehen. Auch als unterstützende Einheit in der Chirurgie [17] können Roboter ihren Einsatzort finden.
Von einem gewöhnlichen Anwender kann jedoch nicht erwartet werden, dass dieser sich mit speziellen Eingabemethoden eines Serviceroboters auskennt. Es ist beispielsweise bei Servicerobotern zur Unterstützung von alten Menschen auch schlichtweg unrealistisch, dass diese zuerst ein 400-seitiges Handbuch mit allen Kommandos lesen und vor allem verstehen sollen. Es ist allgemein unpraktisch, wenn zur Interaktion mit dem System Vorwissen oder umfangreiche Einarbeitung notwendig sind.
Da bei zwischenmenschlicher Kommunikation Sprache als Hauptmedium genutzt wird, liegt es auch bei der Mensch-Roboter-Interaktion (MRI) nahe, natürliche Sprache als Medium zu nutzen. Die Verarbeitung von natürlichsprachlichen Eingaben würde es jedem Nutzer ermöglichen, einfach mit dem Roboter zu interagieren. Gerade wenn es zum Beispiel um die Rettung von Menschen in gefährlichen Gebieten geht, muss der Roboter auch in der Lage sein, sofort mit den Verletzten zu kommunizieren. Neben der zuverlässigen Spracherkennung muss der Roboter jedoch auch eine Möglichkeit besitzen, zu bestimmen woher ein Geräusch oder ein Hilferuf kommt. Ein Beispiel für die Notwendigkeit eines solches Systems wäre ein verrauchter Raum, eines brennenden Gebäudes, in dem der Roboter mit einfachen Kamerabildern kaum weiterführende Informationen bekommen kann.
Eine *einfache* Aufgabenstellung für einen Serviceroboter im Haushalt ist in Abbildung 1.1 dargestellt. Der Besitzer des Serviceroboters ruft diesem zu, dass er herkommen soll. Die Aufgabe des Roboters ist nun herauszufinden, wo dieses *hier* überhaupt liegt. Ein anderer Anwendungsfall wäre ein Robotersystem als Touristenführer. Sobald eine Frage gestellt wird, z.B. *"Was ist das dort?"*, muss der

Roboter zuerst einmal feststellen, woher die Frage kommt, da er aus der Semantik der Frage schließen sollte, dass eine Person der Reisegruppe gerade auf irgendetwas zeigt. Nachdem er die Person gefunden hat, kann er womöglich ein Bild analysieren oder anderweitig Informationen sammeln und auswerten. Der Roboter muss somit ein solide gestaltetes auditorisches System besitzen, um den *normalen* Umgang mit dem Anwender zu gewährleisten.

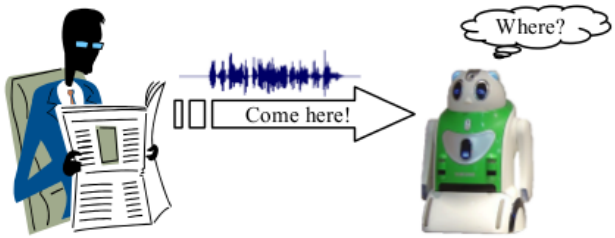


Abbildung 1.1: Aufgabenstellung eines Serviceroboters

Durch welche Methoden eine Geräuschquelle im Raum lokalisiert werden kann, ist hierbei die Hauptfragestellung, die bearbeitet werden soll.
Es gibt zwar Ansätze, die schon in den neunziger Jahren entwickelt wurden, es fehlte den Robotern jedoch an der notwendigen Rechenleistung [7]. Heutzutage ist die Technik soweit, dass auch autonome Roboter in der Lage sind, die relativ intensiven Berechnungen durchzuführen. Dies spannt ein attraktives Anwendungsgebiet auf.

Leitfaden

Die Arbeit ist nun wie folgt gegliedert. Im nächsten Kapitel finden sich einige Grundlagen, die zur Aneignung von eventuell fehlendem Vorwissen dienen sollen. Im Kapitel 3 wird die Fragestellung aufgeführt und verwandte Arbeiten herangezogen. Darauf folgend wird die Zielstellung und Umsetzung meines eigenen Verfahrens beschrieben. Im Abschnitt 5 sind zwei Versuche dargestellt, mit denen mein Verfahren getestet wurde. Abschließend findet sich eine Zusammenfassung der Ergebnisse und ein Ausblick auf weitere denkbare Arbeitsansätze.

Kapitel 2

Grundlagen

Im folgenden Abschnitt wird auf einige Grundlagen eingegangen, die zum Verständnis der später vorgestellten Arbeiten beitragen sollen. Weiterhin wird der humanoide Roboter beschrieben, mit dem mein späteres Verfahren realisiert wird.

2.1 Nao Roboter

Abbildung 2.1: Nao Roboter

Mein gesamtes Verfahren wird mithilfe eines humanoidem Roboters realisiert [4]. Der Nao Roboter ist 53 Zentimeter groß und besitzt komplexe Bewegungsmöglichkeiten mit 5 Freiheitsgraden pro Arm und Bein, sowie 2 Freiheitsgraden für den Kopf und jede Hand. Er bewegt sich auf zwei Beinen fort und besitzt 4 Mikrophone an seinem Kopf. Dabei befinden sich das linke und das rechte Mikrophon

in der gleichen X-Z Ebene und sind an der Y Achse gespiegelt. Das vordere und das hintere Mikrophon befinden sich beide auf der Y-Achse, unterscheiden sich jedoch in Höhe und Entfernung zum Mittelpunkt. Die Positionen der Mikrophone am Roboter sind in Abbildung 2.2 zu erkennen. Die Mikrophone besitzen eine maximale Sampling Rate von 48 kHz und einem Frequenzbereich für die Aufnahme von 20 Hz - 20 kHz.

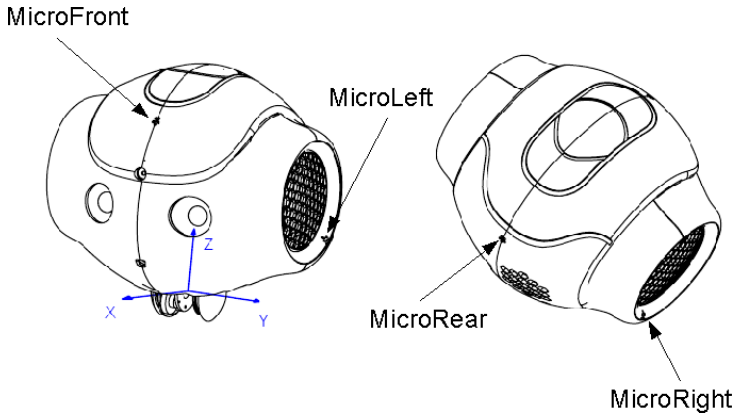


Abbildung 2.2: Mikrophonpositionen

Der Kopf des Roboters lässt sich dabei aus der Ausgangslage (0°) um jeweils 119.5° nach links bzw. nach rechts in der horizontalen Ebene drehen. Daraus ergibt sich ein Bereich von 239°, der ohne Bewegung des Roboters zugänglich ist. Um die verbleibenden 121° zu erreichen, ist eine Drehung des ganzen Roboters notwendig. Wie in Abbildung 2.3 erkennbar ist, lässt sich der Kopf auch nach vorne bzw. nach hinten neigen. In meinem Verfahren möchte ich jedoch die Neigung des Kopfes stets auf 0 Grad belassen und mich nur auf die horizontale Ebene konzentrieren.

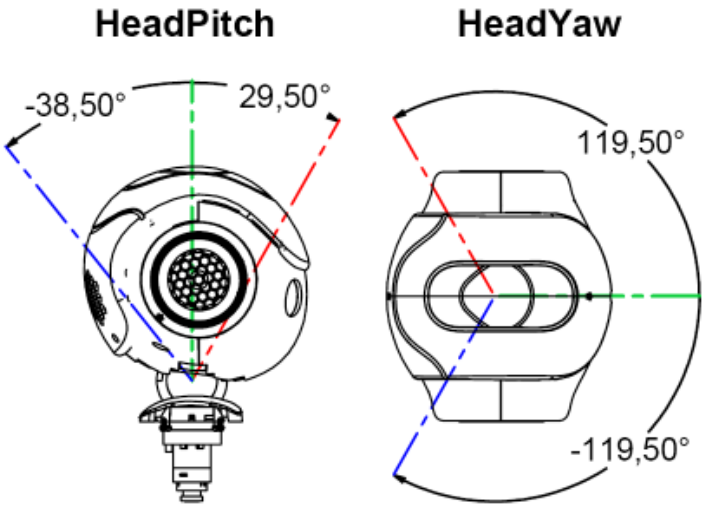


Abbildung 2.3: Bewegungsfreiheit des Kopfes

2.2 Fourier Transformation

Die Fourier Transformation ist eine grundlegende Methodik der Mathematik [23]. Sie dient dazu, ein Signal, welches über den zeitlichen Verlauf abgebildet ist, im Verlauf über den Frequenzraum darzustellen. Da in dieser Arbeit nur diskretisierte Signalfolgen auftauchen, wird auch nur die Diskrete Fourier Transformation (DFT) erläutert.
Angenommen es liegt ein zeitlich diskretisiertes Signal $x(n)$ mit $0 \leq n \leq N$ endlicher Länge vor. Gleichung 2.1 zeigt dabei die Abbildungsvorschrift der Fouriertransformation, wobei $e^{j\Omega}$ eine komplexe Zahl und Ω selbst den Winkel der komplexen Zahl in der Polarkoordinatendarstellung beschreiben.
Die Frequenzdomäne, in die wir das Signal transformieren wollen, wird nun in N gleiche Teile aufgeteilt. Da eine komplette Kreisfrequenz 2π sind, werden die Punkte also gleichmäßig im Intervall von 0 bis 2π verteilt. Ω kann nun durch Ω_k (Gleichung 2.2) ersetzt werden. Es ergibt sich daraus die Gleichung 2.3, die einen Wert für die Fouriertransformation für den Parameter k beschreibt

$$X(e^{j\Omega}) = \sum_{n=0}^{N} x(n) \cdot e^{-jn\Omega} \tag{2.1}$$

$$\Omega \to \Omega_k = \frac{2\pi}{N} \cdot k \tag{2.2}$$

$$X(e^{j\Omega_k}) = \sum_{n=0}^{N} x(n) \cdot e^{-jn\Omega_k} \tag{2.3}$$

In der Literatur ist es typischerweise so, dass Ω_k durch den Frequenzindex k ersetzt wird. Die Gleichung ändert sich demzufolge in der Schreibweise zu:

$$X(k) = \sum_{n=0}^{N} x(n) \cdot e^{-jn\frac{kn}{N}}, k = 0, 1, 2, \cdots, N-1 \tag{2.4}$$

Das Ergebnis der Diskreten Fourier Transformation ist eine Abbildung in den Raum der Komplexen Zahlen. Die Komplexen Zahlen sind dabei als Polarkoordinaten dargestellt, wobei der Winkel im Bereich von 360° gleichmäßig verteilt ist. Der Betrag der Komplexen Zahl gibt dabei an, wie häufig die zugrunde liegende Frequenz auftritt.

Abbildung 2.4 veranschaulicht, wie die Fourier Transformation arbeitet. Angenommen wir haben einen Sinuston mit 2 Hertz (Hz) und ein Mikrophon, was diesen Ton (rote Linie) eine Sekunde lang mit einer Samplingrate von 32 Hz abgetastet hat (blaue Punkte). Das Spektrum wird mit der grünen Linie gekennzeichnet. Dort ist auch der Höchstwert für 2 Hz zu erkennen.

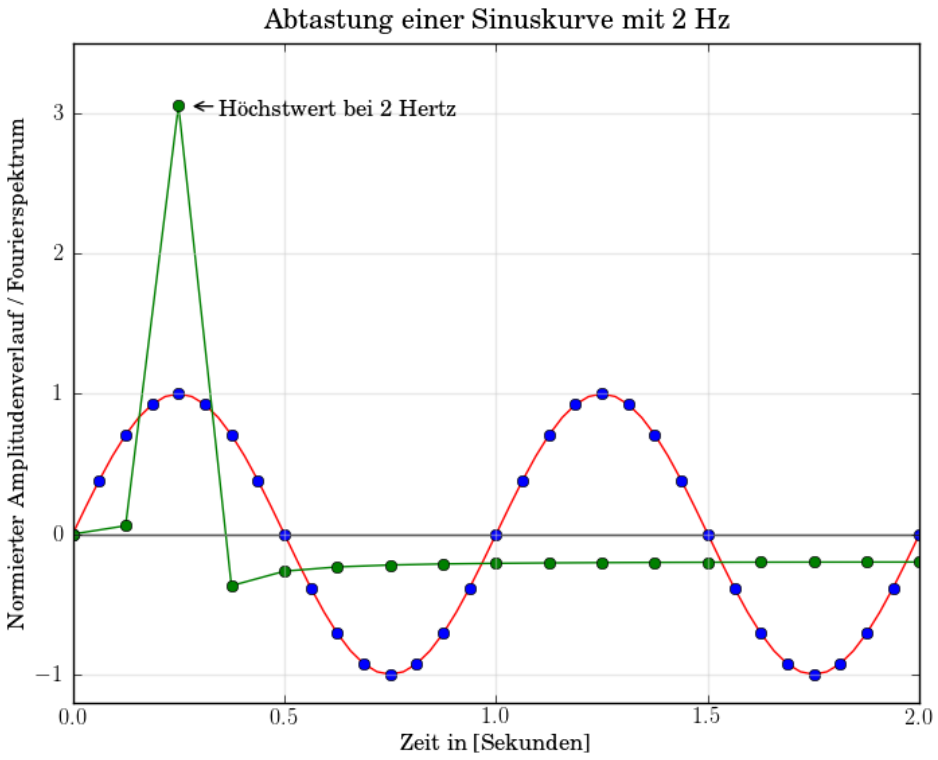


Abbildung 2.4: Beispiel einer Fouriertransformation

2.3 Audiodaten

Abtastung

Wenn Audiodaten aus der realen Welt mittels Mikrophonen aufgenommen werden, müssen diese diskretisiert werden. Diesen Vorgang nennt man Abtastung (engl. Sampling). Ein Sample ist dabei der diskretisierte Wert der Amplitude eines analogen Audiosignals zu einem bestimmten (diskreten) Zeitpunkt. Die Abtastrate beschreibt, wie oft in einer Sekunde eine Abtastung des analogen Signals vorgenommen wird.

Eigenschaften des aufgenommen Soundfiles

Alle in dieser Arbeit verwendeten Audiodaten werden mit einer Abtastrate von 48 kHz aufgenommen. Mithilfe dieser Abtastrate f lässt sich bestimmen, wie viel Zeit zwischen zwei Samples vergangen ist.
Unter Zuhilfenahme der Gleichung 2.5 lässt sich die vergangene Zeit zwischen zwei Abtastungen mit $22.8\overline{3}\ \mu s$ (Gleichung 2.6) bestimmen [21].

$$\Delta t = \frac{1}{f} \tag{2.5}$$

$$\Delta t = \frac{1}{48000\ s^{-1}} \tag{2.6}$$

2.4 Korrelationsanalyse

Unter Korrelationsanalyse (engl. Correlation Analysis) versteht man eine Abbildung von zwei Signalen über Zeit, f(t) und g(t), auf einen Wert im Intervall [-1,1]. Diese Abbildung beschreibt, wie sehr die beiden über die Zeit verlaufenden Signale sich ähneln.

Cross Correlation

Um den Zeitversatz zwischen zwei aufgenommenen Audiosignalen zu bestimmen, eignet sich ein Verfahren namens Cross Correlation (CC) [15]. Damit ist es möglich den Zeitunterschied (engl. Time Delay of Arrival, TDOA) zwischen zwei Mikrophonen zu bestimmen. Bei dem Verfahren wird eines der diskretisierten Signale jeweils um einen Schritt in der Zeit (also um ein Sample) verschoben und der Gesamtfehler zwischen den Amplitudenverläufen bestimmt. Ziel des Verfahrens ist es, den Wert für die Verschiebung zu bestimmen, an dem die Fehler minimal und damit die Correlation maximal wird.

Nehmen wir an, dass die Signale f(n) und g(n), mit der Länge N, diskretisiert vorliegen. Dabei ist N gleich der Dimension der beiden Vektoren. Zunächst werden beide Vektoren mittels 2.7 normiert. Darauf folgend wird die Correlation über den Zeitversatz T mittels 2.8 bestimmt.

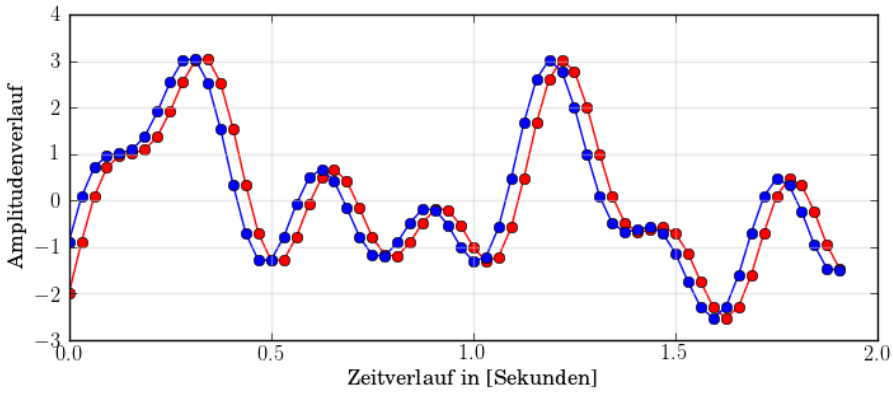
$$f_{norm}(n) = \frac{f(n)}{\sum_{n=0}^{N} f\,(n)^2} \tag{2.7}$$

$$Corr(f,g)_T = \sum_{n=0}^{N} f(n) \cdot g(n+T) \tag{2.8}$$

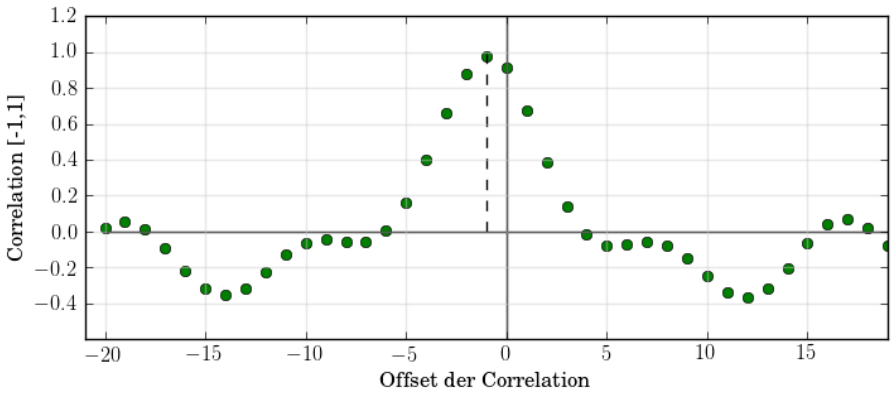
Der Verlauf der Correlation über den Zeitversatz T ist in Abbildung 2.5 dargestellt. Man erkennt dort auch den maximalen Punkt der Correlation bei T = -1.

Generalized Cross Correlation

Das Problem in der Zeitdomäne ist jedoch, dass der optimale Zeitunterschied zwischen zwei Sample liegen könnte. Um dies mittels Cross Correlation zu lösen, ist dann eine Interpolation zwischen den Samples nötig, welche die Algorithmen wesentlich komplexer werden lassen.
Um dies zu bewältigen, lässt sich Generalized Cross Correlation (GCC) verwenden. Dieses Verfahren nimmt für die Signale eine Gewichtungsfunktion hinzu. Je nachdem, wie diese Funktion gewählt ist, sind enorme Verbesserungen im Bestimmen des TDOAs möglich.



(a) Signalverlauf f(t) und g(t)



(b) Verlauf der Correlation

Abbildung 2.5: Cross Correlation

2.5 Künstliche Neuronale Netze

Um ein System zu entwerfen, was für eine gewisse Eingabe eine Klassifikation (Einordnung in eine Kategorie) trifft, eignen sich Künstliche Neuronale Netze (KNN) besonders gut [20]. Künstliche Neuronale Netze bilden dabei, wie auch ihr biologisches Vorbild, ein Modell, welches auf der Verknüpfung von vielen kleinen, sehr simpel gehaltenen Einheiten basiert. Die künstlichen Neuronen können dabei in verschiedensten Strukturen angeordnet werden. Oft genutzte Strukturen sind Feed-Forward-Strukturen, die die Neuronen in einfachen Schichten anordnen und somit eine Eingabe auf eine Ausgabe abbilden. Rekurrente Neuronale Netze (RNN) haben rückführende Strukturen, die manche Ausgaben von Neuronen wieder als Eingabe an vorhergehende Neuronen zurückgeben. Diese Strukturen sind besonders dafür geeignet, wenn Neuronale Netze einen vorherigen *Zustand* bei der nächsten Berechnung mit in Betracht ziehen sollen.

2.6 "How we localize Sound"

Wie der Mensch überhaupt in der Lage ist, die Richtung einer Soundquelle zu bestimmen, liefert oftmals die Grundlage für die verschieden technischen Lösungsansätze. William M. Hartman beschreibt dabei 1999 zwei grundlegende Prinzipien, die es dem Menschen ermöglichen, eine Geräuschquelle zu lokalisieren [7]. Diese beiden Grundlagen sind die Interaural Level Difference (ILD) und die Interaural Time Difference (ITD). Die ILD beschreibt dabei den Intensitätsunterschied in der Amplitude am linken bzw. am rechten Ohr. Sie wird in Dezibel angegeben und hängt stark von der Frequenz des Geräusches ab. Bei hochfrequenten Tönen nimmt die Amplitude im Vergleich vom linken zum rechten Ohr aufgrund der Streuung und Trübung am menschlichen Kopf und Körper stärker ab. Niederfrequente Töne unterliegen jedoch am menschlichen Kopf einer Beugung und erreichen daher beide Ohren mit fast gleicher Intensität. Nun ist der Mensch jedoch auch problemlos in der Lage, Töne im niederfrequenten Bereich zu lokalisieren. Dies ist mithilfe des ITD möglich. Die Interaural Time Difference beschreibt dabei den Zeitunterschied, mit dem ein Geräusch an beiden Ohren des Menschen ankommt. Der Unterschied in der Zeit und der Unterschied in der Phase eines Signals sind dabei proportional zueinander. Die Zeitunterschiede liegen dabei im Mikrosekundenbereich. Zurecht mag also hier die Frage aufkommen, wie dieser Sachverhalt bei einem Nervensystem mit Verzögerungen im Bereich von Millisekunden überhaupt möglich ist.
Inzwischen gibt es Beweise dafür, dass ein Verarbeitungssystem, die 'superior olive' im Mittelhirn, in der Lage ist, eine Art Cross Correlation zu bewerkstelligen. Zugrunde liegend ist dafür das Modell von L. A. Jeffress, welches in Abbildung 2.6 ein grobe Vorstellung von dem Sachverhalt zeigt [9].

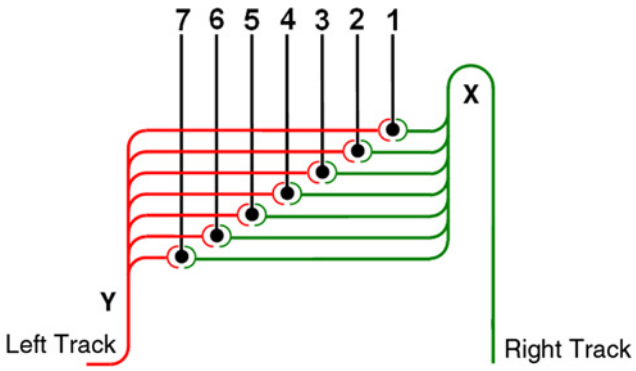


Abbildung 2.6: Delay Line Modell

Kapitel 3

Problemstellung und verwandte Arbeiten

Im folgenden Kapitel wird die Problemstellung dargestellt und verwandte Arbeiten herangezogen, welche bisherige Lösungsansätze beschreiben.

3.1 Fragestellung

Die Fragestellung dieser Arbeit ist nun, wie ein Roboter überhaupt ein Geräusch im Raum zu finden vermag. Um ein Geräusch aufzunehmen, muss er die akustischen Sensoren benutzen und das analoge Signal in ein digitales Signal umsetzen. Dieses kann dann mithilfe von verschiedenen Verfahren einen Rückschluss auf die Position der Geräuschquelle im Raum liefern. Die Geräuschquellenlokalisation sollte möglichst genau und robust erfolgen. In meiner Arbeit möchte ich dabei bioinspiriert mittels Korrelationsanalyse und Künstlichen Neuronalen Netzen vorgehen. In der Literatur verwendete Verfahren beinhalten verschiedene Grundelemente, die die Einteilung in drei unterschiedliche Rubriken möglich machen.

3.2 Verwandte Arbeiten

3.2.1 Lokalisierung mittels Korrelationsanalyse

Mithilfe der Korrelationsanalyse wird in der Regel die Interaural Time Difference und die Interaural Level Difference bestimmt.

So beschreiben Lee et. al. (2008) wie mithilfe des TDOA und 4 Mikrophonen, die vertikale Position einer Geräuschquelle bestimmt werden kann [11]. Es wird dabei ein künstliches Ohr verwendet. Mit einer Ohrmuschel ausgestattet, ermöglicht es die Vermeidung von Vieldeutigkeit in der Positionsbestimmung. Dabei zeigt das Verfahren im allgemeinen gute Erfolgsquoten für die korrekte Bestimmung des Winkels. Es treten jedoch signifikante Fehler auf, sobald sich die Geräuschquelle über dem Roboterkopf befindet.

Im Verfahren von Murray et. al. 2009 wird eine physikalisch motivierte Herangehensweise gewählt [16]. Zunächst werden zur Vorverarbeitung des Signals verschiedene Verfahren genutzt: Zur Unterscheidung, ob ein Signal relevant ist, wird der Energiegehalt der Aufnahme ermittelt. Dieser ist bei menschlicher Sprache, im Gegensatz zum einfachen Rauschen, signifikanten unterschiedlich. Weiterhin wird zur Anpassung an die Umgebung die Lautstärke der Aufnahme moduliert. Wenn das Signal relevant ist, wird darauf folgend die Interaural Time Difference, mittels General Cross Correlation bestimmt. Auf Grundlage der Ausbreitungsgeschwindigkeit des Schalls wird der Winkel bestimmt, an dem sich die Geräuschquelle befindet. Das Verfahren erreicht für Objekte, die vor dem Roboter liegen und sich demnach im Bereich von 0° aufhalten, sehr hohe Genauigkeiten. Eine Abweichung von maximal ±1.5° wird angegeben. Mit einer Abweichung von ±7.5° gibt es jedoch im Bereich von 90° signifikante Ungenauigkeiten beim Lokalisieren der Geräuschquellen. Auch wurde erwähnt, dass das Verfahren, aufgrund der zeitintensiven Berechnung der Correlation, nicht Echtzeitfähig war.

Liu und Shen (2010) nutzen eine erweiterte Form der GCC: GCC-PHAT [12]. Die Correlation wird mithilfe einer Phasentransferfunktion, also im Frequenzraum, durchgeführt. Das Hauptaugenmerk liegt dabei auf einem Modell um Echoeffekte auszuschließen und somit eine genauere Lokalisierung zu ermöglichen. Das Verfahren verwendet 4 Mikrophone zu Berechnung der Correlation. Es werden weiterhin Entfernungen unterschieden. So werden Aufnahmen mit einer Entfernung von 1 m, 2 m und 3 m verwendet. Größere Genauigkeit der Lokalisation findet sich dabei vor allem im nahen Bereich. Mit zunehmender Entfernung nimmt auch die Genauigkeit, besonders für die Bereiche von 0° und 180°, ab. Durch Hinzunahme eines weiteren Mikrophonpaars können diese Fehler auch für weitere Entfernungen drastisch reduziert werden.

3.2.2 Lokalisierung mittels Neuronalen Netzen

Ein weiterer Ansatz ist der Einsatz von Neuronalen Netzen, da diese allein mithilfe von vorhandenen Trainingsdaten ein komplexes mathematisches Modell durch geeignete Lernverfahren abstrahieren können.

Im Verfahren von Czyzewski (2003) wird gezeigt, wie aus aufgenommenen Geräuschen mithilfe von Beamforming [8] verschiedenste Features extrahiert werden [3]. Diese werden unter drei verschiedenen Trainingsalgorithmen einem Neuronalen Netz präsentiert. Bei dem eingesetzten Neuronalen Netz handelt es sich um ein Feedforwardnetz, welches in der Lage ist eine Abbildungsvorschrift zu erlernen. Die Erfolgsquote der Erkennung schwankt dabei abhängig vom gewählten Lernverfahren und den genutzten extrahierten Features. Die Erfolgsrate der Neuronalen Netze befindet sich dabei zwischen 78% und 92% bei präsentierten Testdaten.

Ein weitaus näher an die Biologie angelehntes Verfahren findet sich bei J. Liu et. al. (2010), dessen Modell sich am auditorischen System im menschlichen Gehirn orientiert [13]. Es werden spezielle Künstliche Neuronale Netze (KNN) genutzt, um die Struktur im Gehirn nachzustellen. Dabei wird die Funktionsweise der *Medial superior olive* (MSO) und der *Lateral superior olive* (LSO) zur Bestimmung des ITD (MSO) und des ILD (LSO) herangezogen. Die Genauigkeit des Systems ist jedoch über den gesamten Winkelbereich beim Nutzen nur einer Komponente, entweder MSO oder LSO, relativ nüchtern. Es werden gerade mal 25% erreicht. Bemerkenswert ist jedoch die Zunahme der Genauigkeit im gesamten System, wenn die gewonnenen Daten aus MSO und LSO im *Inferior Colliculus* (IC) kombiniert werden. Es werden damit Genauigkeiten von 80%, im Bereich von -45° bis 45° sogar 90%, erreicht. Es treten auch hier, trotz hoher Genauigkeit im Bereich von 0°, erneut Schwächen im Bereich von 90° auf. Weiterhin ist das System aufgrund der rechenintensiven Verfahren nicht Echtzeitfähig.

Eine weitere Möglichkeit, Neuronale Netze einzusetzen, ist das Voraussagen der Position der Geräuschquelle [16]. Einerseits kann es Rechenzeit sparen und andererseits dem System einen gewissen *Vorsprung* verschaffen. Murray, Erwin und Wermter setzten 2009 auf das oben erwähnte Verfahren mithilfe von Cross Correlation ein weiteres System auf, welches eine Vorherbestimmung der Position mithilfe von Rekurrenten Neuronalen Netzen (RNN) ermöglicht. Dem Netz werden dabei die Positionen der Geräuschquelle zum Zeitpuntk t_0 und danach t_1 vorgegeben und das RNN bestimmt daraus die Position, an der die Soundquelle zum Zeitpunkt t_2 voraussichtlich sein müsste. Das RNN liefert dabei recht zuverlässige Daten und bildet damit eine solide Grundlage zur Approximation der Position der sich bewegenden Geräuschquellen.

3.2.3 Lokalisierung mittels Lookup Tables

Eine weitere Herangehensweise, die in dieser Arbeit betrachtet wird, beinhaltet das Erstellen einer Lookup Table. Diese Verfahren erzeugen, bevor das System tatsächlich zum Einsatz kommt, eine Datenstruktur in der einfache und schnell zu berechnende Features auf ein Ergebnis (meist die Position) abgebildet werden. Um solch eine Datenstruktur herzustellen, müssen jedoch eine Menge an Trainingsdaten vorhanden sein und das System benötigt eine gewisse Vorlaufzeit um einsatzbereit zu sein.

Im Verfahren von Cho et. al. (2009) wird dabei der Raum um den Roboter herum in kleine Bereiche aufgeteilt [2]. Nach dem Aufnehmen des Signals und der Bestimmung des Time Delay of Arrival (TDOA) kann für jeden Bereich ein bestimmter Wert mithilfe SRP-PHAT bestimmt werden. Der Bereich, an dem der Wert maximal ist, kann als Position der Geräuschquelle ausgegeben werden. Da jedoch bei sehr starker Aufteilung des Raumes in kleine Bereiche unglaublich viele Punkte entstehen, die durchsucht werden müssten, beschreiben die Autoren ein Verfahren, welches Punkte im Raum mit dem gleichen ITD zusammenfasst. Die somit erzeugte Datenstruktur ist wesentlich schlanker und ermöglicht so die Echtzeitfähigkeit. Das Verfahren bietet zudem einen hohen Grad der Genauigkeit. Es werden in der horizontalen Ebene 93.9% und in der vertikalen Ebene bis zu 92.8% Genauigkeit erreicht.

In einem weiteren Verfahren von Czyzewski (2003) wird mithilfe von verschiedenen Vorverarbeitungsmethoden ein regelbasiertes System erstellt [3]. Er beschreibt, wie zunächst mithilfe von Subbandfiltern einzelne Frequenzbänder extrahiert werden. Diese werden mithilfe von Korrelationsanalysen bewertet und es wird mit den gewonnenen Daten eine Regelbasis konstruiert. Die Ergebnisse zeigen bei einer groben Bestimmung der Richtung sehr hohe Genauigkeiten. Jedoch wird beim Vorhandensein von Störgeräuschen die Genauigkeit selbst beim groben Bestimmen der Richtung drastisch reduziert. Das System erreicht bei einer Signal-to-Noise-Ratio (SNR) von 0 nur 52% bei -20 SNR nur ca. 70% Erfolg. Dies ist bei der groben Unterteilung der Herkunft in 4 Bereiche ein schlechtes Ergebnis. Bei größerer Differenzierung der Richtung erreicht das System von 0° bis 20° in 5°-Schritten jedoch einen durchschnittliche Genauigkeit von 90%. Jedoch wäre, für vergleichbare Daten, ein Test am kompletten 360° Kreis notwendig.

Das Verfahren von Guentchev und Weng (1998) bildet ein klassisches Suchverfahren [5]. Es werden aus einem mit 4 Mikrophonen aufgenommenen Geräusch sowohl die 6 Werte für ITD als auch für ILD extrahiert. So kommen 12 Werte zusammen, die auf die bekannte Positionen im Raum abgebildet werden. Dabei wird für die Positionsdarstellung, aufgrund der Empfindlichkeit der Entfernung, die Polarkoordinatendarstellung gewählt. Aus vielen solchen Beispielen wird mithilfe des SHOSLIF-RPT Algorithmus ein Baum aufgebaut, der bei der Verwendung des Sys-

tems genutzt wird. Es wird dabei sichergestellt, das die k ähnlichsten Punkte zur Eingabe, jeweils in einer Laufzeit von O(log(n)) bei n Einträgen im Baum gefunden werden. Das Verfahren liefert dabei gute Erfolge und weicht in der horizontalen Ebene nur ca. $\pm 2.2°$, in der vertikalen Ebene um $\pm 2.8°$ ab. Nur die Entfernungsbestimmung liefert Fehler von durchschnittlich $\pm 19\%$.

Da für die Roboter beim Bewegen zu einem Ort auch die Entfernung eine große Rolle spielt, möchte ich noch ein Verfahren vorstellen, welches sich hauptsächlich diesem Problem annimmt. Das Verfahren von Rodemann 2010 beschreibt dabei das Extrahieren von sogenannten *Audio Proto Objects* [19]. Es handelt sich dabei um eine Zusammenstellung charakteristischer Features eines Geräusches. So werden z.B. ITD, ILD, das Spektrum, sowie die Amplitude mit einbezogen. Auch die Kategorie, in die ein Audiosignal fällt, zeigt extreme Auswirkungen auf die Genauigkeit der Entfernungsbestimmung. Das Verfahren ist dabei mit der Kombination der eben aufgeführten Eigenschaften in der Lage, eine Genauigkeit von 77% bei der Entfernungsbestimmung zu erreichen. Das Verwechseln von nahen und entfernten Geräuschen fällt dabei nur zu einem kleinen Teil des Gesamtfehlers ins Gewicht. Die Verwechselung von nahen und fernen Geräuschen besitzt nur einen Fehler von 0.12%. In einem normalen Raum sind bei Entfernungen bis zu 6 Metern also realistische Entfernungsbestimmungen möglich.

3.2.4 Ergebnisse

Die dargestellten Verfahren beschreiben verschiedenste Verarbeitungen der gesammelten Audiodaten. Alle Verfahren basieren in ihrer Grundidee jedoch auf dem von Hartmann beschriebenen Zeit- bzw. Amplitudenunterschied. Einige Verfahren extrahieren noch weitere Eigenschaften aus den gewonnenen Audiodaten, um die Qualität noch weiter zu steigern. Es sollte jedoch überlegt werden, wie weit die Genauigkeit einer Lokalisation, für bestimmte Anwendungen bereits ausreicht und wo noch explizite Verbesserungen nötig sind.

3.3 Methodik

Um im späteren Teil meiner Arbeit eine objektive Auswertung meiner Testergebnisse zu erreichen, möchte ich nun zunächst die Methoden vorstellen, die ich dazu verwenden möchte. Im Speziellen beschreibe ich, wie einerseits mein Verfahren zur Korrelationsanalyse überprüft und zum anderen die Ergebnisse der Neuronalen Netze sinnvoll dargestellt werden können.

3.3.1 Überprüfung der Korrelationsanalyse

Für die Überprüfung der Korrelationsanalyse ziehe ich zunächst eine Theorie aus einem anderen Verfahren heran [16]. Dort wird der Winkel abhängig vom Offset der Korrelationsanalyse durch ein physikalisches Modell mit der Formel 3.1

bestimmt. Der Parameter c_{air} beschreibt dabei die Schallgeschwindigkeit und beträgt für 20 °C, 343 $\frac{m}{s}$. Θ beschreibt den Winkel der Geräuschquelle und Δt den Zeitunterschied zwischen zwei Abtastungen (Gleichung 2.6).

$$\sigma = \frac{Sin\Theta \cdot c}{c_{air} \cdot \Delta t} \tag{3.1}$$

Die angegebene Formel bezieht sich jedoch auf Mikrophone, die durch keine weiteren Hindernisse, zwischen ihnen, gestört werden. Es wird also angenommen, dass der Schall sich ungehindert zwischen den beiden Mikrophonen ausbreiten kann. Dies ist bei dem Nao jedoch aufgrund der Kopfform nicht gegeben. Die Mikrophone (links und rechts) des Nao sind laut den technischen Daten 12 Zentimeter voneinander entfernt. Der Schall muss jedoch noch den Kopf umrunden, was die Strecke um einige Zentimeter verlängert.
In der Abbildung 3.1 findet sich daher der modellbeschriebene Verlauf des Offsets bei gegebenem Winkel und verschiedenen Abständen der Mikrophone voneinander. Die rote Linie markiert dabei den Abstand von 12 Zentimeter, die der Schall aufgrund der technischen Daten mindestens zurücklegen muss. Die blaue Linie markiert einen Abstand von 22 Zentimetern, die als empirische Obergrenze dieser Analyse festgelegt wurde. Die Farbe der Quadrate markiert den entsprechenden Winkel.
Die Grafik zeigt weiterhin einen Verlauf von Datenpunkten und deren Mittelwerten, die durch eine Korrelationsanalyse des linken und rechten Mikrophons gewonnen wurden. Durch die Grafik lässt sich erkennen, das der Kurvenverlauf des physikalischen Modells und der Korrelationsanalyse ähnlich sind. Um diese Ähnlichkeit auch statistisch zu stützen, möchte ich den mittleren quadratischen Fehler und die Standardabweichung der Daten in Abhängigkeit vom simulierten Mikrophonabstand betrachten. Dazu wurden der mittlere quadratische Fehler aller Datenpunkte eines bestimmten Winkels, bezüglich zum Datenpunkt der entsprechenden Mikrophonentfernung, bestimmt. Die Ergebnisse dieser Analyse sind in Abbildung 3.2 dargestellt. Dort ist auch die Standardabweichung vom mittleren Fehler notiert. In dieser Grafik ist zu erkennen, das bei einem simulierten Abstand der Mikrophone, mit c = 16 cm ein mittlerer Fehler von 1.8 auftritt. Im Mittel weicht meine Korrelationsanalyse bei dem angenommenen Abstand der Mikrophone nur 2 Sample ab. Allgemein ist der mittlere Fehler für die simulierten Werte von ca. 12-19 cm gering und die genutzte Korrelationsanalyse liefert Ergebnisse, die dem physikalischen Modell hinreichend nahe sind.

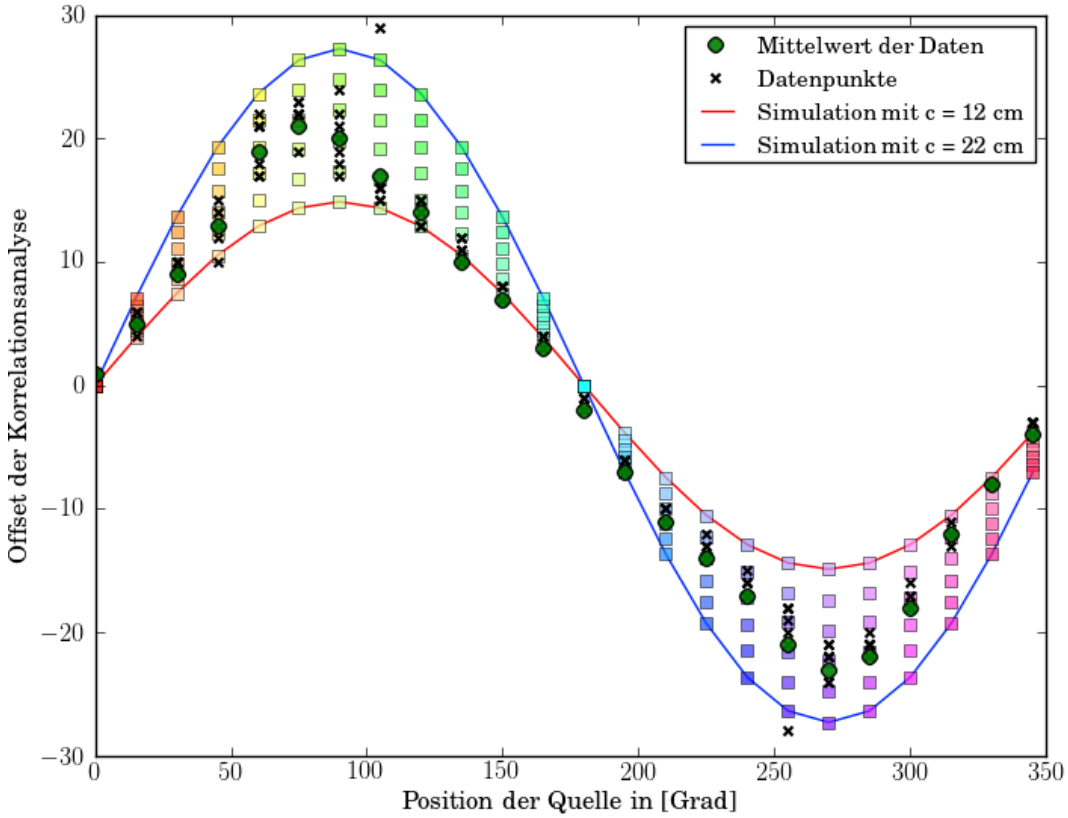


Abbildung 3.1: Verlauf der Simulation und der Korrelationsanalyse

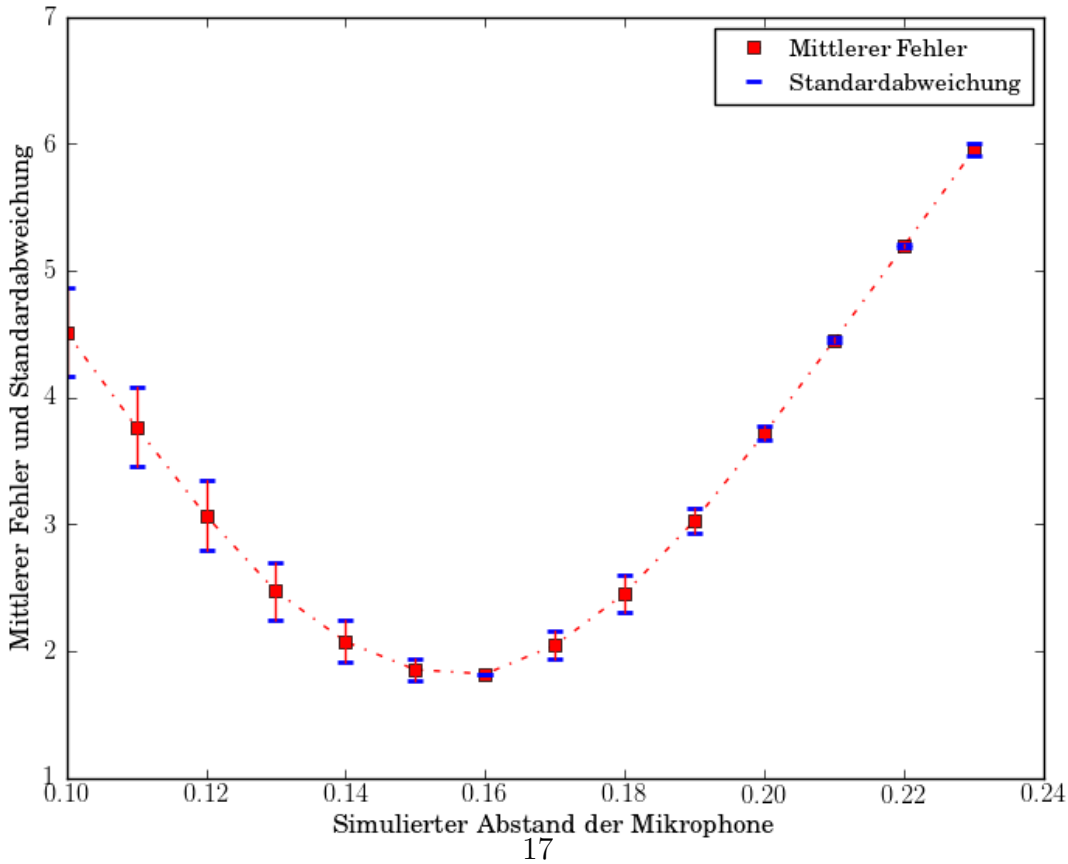


Abbildung 3.2: Mittlerer Fehler in Abhängigkeit des simulierten Mikrophonabstandes

3.3.2 Confusion Matrix

Die Qualität des Künstlichen Neuronalen Netzes möchte ich mithilfe einer Confusion Matrix bestimmen [22]. Eine Confusion Matrix gibt bei einem gegebenem Klassifizierungsverfahren seine Qualität an. Es handelt sich bei der Confusion Matrix um eine 2-dimensionale Matrix, in der jeweils die Zeilen die tatsächliche Klasse eines Testdatums angeben und die Spalten die Klasse, in die es vom Klassifizierungsverfahren eingeordnet wurde. Da im Beispiel des Neuronalen Netzes jeder Eingabewinkel vom System auch als Ausgabewinkel erkannt werden soll, eignet sich die Confusion Matrix perfekt zum Analysieren dieser Aufgabe. Ein Beispiel für solch eine Matrix ist in Abbildung 3.3 zu sehen.
Die Abbildung zeigt dabei ein mögliches Ergebnis einer Confusion Matrix. Die

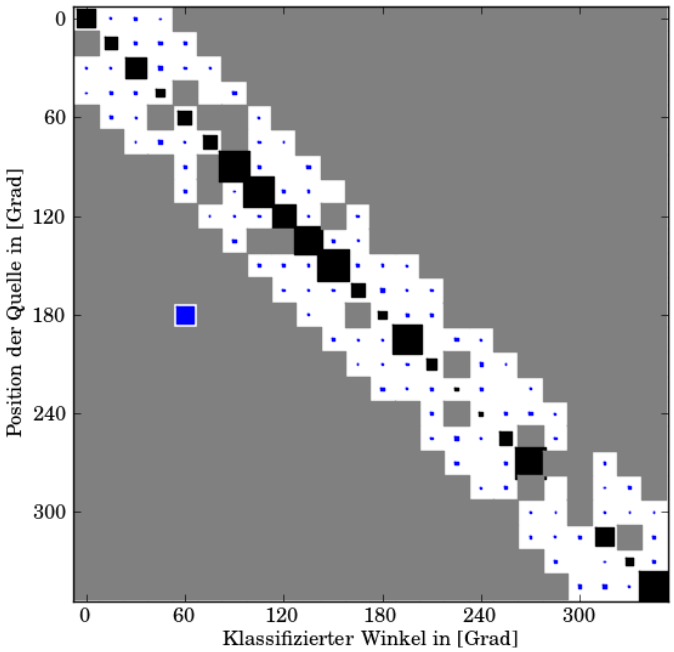


Abbildung 3.3: Beispielhafte Confusion Matrix

schwarzen Einträge auf der Hauptdiagonalen sind richtige Klassifizierungen. Dort hat das Klassifizierungsverfahren den korrekten Winkel ermittelt. Ein Beispiel für eine extreme Fehlklassifikation ist bei 180° zu erkennen. Dort wurde der Winkel fälschlicherweise in die Klasse 60° eingeordnet.

Die Berechnung der Güte der Klassifikation lässt sich anhand des Verhältnisses von richtiger Klassifikationen zur gesamten Datenmenge beschreiben. Alle Einträge auf der Hauptdiagonalen der Confusion Matrix sind korrekte Klassifikationen. Alle anderen Einträge sind fehlerhafte Klassifikationen. Es wird also der Quotient aus den Elementen auf der Hauptdiagonalen und allen Elemente in der gesamten Matrix als Güte der Klassifikation angegeben.
Weiterhin ließe sich in meinem speziellen Anwendungsfall auch noch eine weitere Metrik für die Güte der Klassifikationen einführen, da die Klassen direkte physika-

lische Nachbarn am 360 Grad Kreis sind. Sollten sich zum Beispiel nur Elemente auf der Haupt- und den beiden Nebendiagonalen der Matrix finden, so lässt sich daraus ableiten, dass kein Datum weiter als eine Klasse zu weit *links* bzw. *rechts* eingeordnet wurde.

Kapitel 4

Eigenes Verfahren

4.1 Theoretische Überlegungen

Bei der Betrachtung der bisherigen Verfahren sind an manchen Stellen einige Probleme aufgetreten. So finden sich an den Seitenbereichen, wo die Soundquelle jeweils einen Winkel von ca. 90 Grad einnimmt, weitaus größere Fehler, als um 0 Grad herum [16]. Weiterhin sind zwei Mikrophone in der horizontalen Ebene nicht in der Lage, eine genaue Bestimmung der Position der Geräuschquelle zu gewährleisten. Aufgrund der Front-Back-Konfusion sind stets zwei Punkte möglich, an denen sich eine Geräuschquelle befinden kann.

Unter Hinzunahme eines zweiten Mikrophonpaars kann einerseits die Front-Back-Confusion aufgelöst werden. Andererseits kann bei geeigneter Positionierung der Mikrophonpaare zueinander eine höhere Genauigkeit an den Seitenbereichen erreicht werden. Im Idealfall wäre ein Mikrophonpaar zu verwenden, welches orthogonal zu dem bisherigen Mikrophonpaar liegt. So wäre der ungenaue Bereich des einen Paares gleichzeitig der genaue Bereich des zweiten Paares.

4.2 Eigenes Verfahren

Meine Zielstellung ist es, mit dem Nao Roboter und 3 seiner 4 Mikrophone eine Geräuschquellenlokalisierung am vollen 360 Grad Kreis zu realisieren. Das Verfahren besteht dabei aus mehreren Teilschritten:

1. Das Signal wird von den Mikrophonen des Naos aufgenommen und über WLAN/LAN an einen leistungsstärkeren Rechner übertragen.

2. Dieser Rechner verarbeitet das Signal in mehreren Schritten.

3. Zunächst wird mithilfe eines Energiewertverfahrens entschieden, ob das Signal zur Weiterverarbeitung geeignet erscheint.

4. Anschließend wird optional das Rauschen mit einem einfachen Verfahren reduziert.

5. Darauf folgend wird die Cross Correlation von zwei Mikrophonpaaren bestimmt und in ein trainiertes Neuronales Netz gegeben.

6. Dieses ermittelt dann den Winkel, an dem sich die Geräuschquelle befindet.

7. Die Effektoren des Roboters drehen dann den Kopf in die entsprechenden Richtung bzw. der Roboter teilt auf andere Weise den bestimmten Winkel mit.

4.3 Aufnahme und Transfer

Die Berechnungen werden aufgrund der schwachen Leistung, nicht direkt auf dem Roboter durchgeführt. Die Aufnahme eines Audiosamples erfolgt jedoch auf dem Roboter mithilfe eines einfachen Python-Skriptes (Quellcode 4.1). Die vom Hersteller zu Verfügung gestellte Basissoftware erlaubt das Aufnehmen einer 4 Kanal-Wavedatei mit einer Abtastrate von bis zu 48 kHz. Dabei unterliegt der Kanal 4 jedoch stets dem Einfluss des Lüfters, der ein starkes Störrauschen verursacht. In Abbildung 2.2 lässt sich dicht neben dem hinteren Mikrophon die Lüfteröffnung erkennen. Dadurch ist der Kanal 4 in der Regel nicht für eine Analyse geeignet. Die aufgenommene Datei wird im */tmp* Verzeichnis des Roboters abgelegt. Mithilfe einer ssh-Verbindung kann dann mittels scp die aufgenommene Datei auf einen leistungsstärkeren Rechner übertragen werden, auf dem die eigentliche Berechnungen ausgeführt werden (Quellcode 4.2).

```
1     def recordSample(self, naopath, interval):
2         """
3         This method tells the robot to record a sound sample of a given length and
4         put it into the /tmp directory on the robot's file system
5         """
6         #Tell the microphones to begin the recording
7         self.microphoneProxy.startMicrophonesRecording(naopath +  'sample' + str(self.ID_CODE)
              + '.wav')
8         #Sleep the thread as long as the soundsample is beeing recorded
9         time.sleep(interval)
10        #Stop the recording on the robot
11        self.microphoneProxy.stopMicrophonesRecording()
12        #Return the ID_CODE of the recorded sample and go one ID_CODE further
13        return self.boundCyclingParameters()
```

Quellcode 4.1: Aufnahme

```
1     def __init__(self, NAO_IP, USER, PASSWD):
2         """ This method sets up the scp transfer objects """
3
4         #Create the ssh connection to the robot
5         transport = paramiko.Transport((NAO_IP, 22))
6
7         #Connect to it with a username and password
8         transport.connect(username=USER, password=PASSWD)
9
10        #Build the SFTP Object to get files from the robot
11        self.sftp = paramiko.SFTPClient.from_transport(transport)
12
13    def transferSample(self, id_code, naopath, localpath):
14        """ This method transports the recorded file to the local file system """
15
16        #Determine the filename on the nao file system as well as on the target filename
17        filename = 'sample' + str(id_code) + '.wav'
18
19        #Transport the filename
20        self.sftp.get(naopath + filename, localpath + filename)
21
22        #Printing a Message of the Transportation on the Console
23        print "Transported: nao:" + naopath + filename  + "  >  local:" + localpath + filename
```

Quellcode 4.2: Transfer

4.4 Vorverarbeitung

4.4.1 Energiewertbestimmung

Um eine gute Unterscheidung zwischen gewünschten und unerwünschten Signalen zu erreichen wird in meinem Verfahren eine Energiefunktion genutzt [16]. Der Energiegehalt ϵ wird bei jeder Aufnahme bestimmt und gibt dabei ein Maß für den Anteil von menschlicher Stimme in der Aufnahme an. Die Quadrate der Amplitudenwerte des Signals werden über die Zeit aufsummiert. Anschließend wird der aufsummierte Wert durch die Anzahl der Samples geteilt, um einen von der Länge der Aufnahme unabhängigen Wert zu erhalten (Gleichung 4.1).

$$\epsilon = \frac{\sum_{i=0}^{n} [y_i]^2}{n} \tag{4.1}$$

Im Beispielsample aus Kennedys Rede (Abbildung 5.2) erkennt man ein gewisses Grundrauschen. Aufgrund des Energiegehalts dieses Grundrauschens lässt sich eine Entscheidungsgrenze für das Verwenden des Signals bestimmen. Da in jeder Umgebung gewisse Störquellen (Computerlüfter, Summen von Aggregaten, usw.) zu finden sind, wird bei meinem Verfahren zu Beginn eine Kalibrierung vorgenommen, indem der Roboter eine bestimmte Zeit lang die *Stille* aufnimmt, um daraus eine Grundlinie zu berechnen. Die Berechnung der Energie in der Software lässt sich im Quellcode 4.3 einsehen.

```
class Energy():
    """ This Class is able to estimate the Energy of a Sample """

    def __init__(self, frame, threshold):
        # Save the frame data
        self.frame = frame
        # Save the given threshold
        self.threashold = threshold
        # Set the activation to false
        self.activation = False
        # Set the Energy to zero
        self.energy = 0
        # Call the method to calculate Energy
        self.__calculateEnergy()

    def __calculateEnergy(self):
        """ This Method calculates the Energy of the Sample """
        # Estimate minimum and maximum values of the sample
        self.minimum = min(self.frame)
        self.maximum = max(self.frame)
        # Calculate the squared sum over all samples
        for i in range(len(self.frame)):
            self.energy += (self.frame[i]*self.frame[i])
        # Normalize it by dividing it through the length of the sample
        self.energy = self.energy / len(self.frame)
        # Check if the energy is above the threshold
        if (self.energy > self.threashold):
            # If so set the activation to true
            self.activation = True
```

Quellcode 4.3: Energieberechnung

4.4.2 Rauschreduzierung

Nachdem die Energieanalyse eines Geräuschs entschieden hat, dass dieses weiter genutzt werden soll, kann das Rauschen mithilfe von *Spectral Substraction* reduziert werden. Diese Vorverarbeitung wurde bei meinen Versuchen jedoch als starke Fehlerquelle identifiziert. Bei dem Verfahren handelt es sich im Wesentlichen um eine Implementierung des von S. Boll beschriebenen Ansatzes [1]. Eine Aufnahme, die lediglich Rauschen enthält, wird in einzelne Blöcke unterteilt. Von jedem Block wird das Spektrum mithilfe der Fouriertransformation bestimmt und ein Mittelwert über alle Blöcke gebildet. Anschließend wird das so gebildete Spektrum vom Spektrum jedes anschließend aufgenommenen Geräusches subtrahiert. Durch die inverse Fouriertransformation wird das so gebildete Spektrum wieder in den Amplitudenverlauf über die Zeit umgewandelt.

4.5 General Cross Correlation

Sobald die Vorverarbeitung erfolgt und die Aufnahme zur Weiterverarbeitung geeignet ist, wird mit der Korrelationsanalyse begonnen (Quellcode 4.4). Dazu muss jedoch noch ein Interval festgelegt werden, in dem der zweite Kanal gegenüber dem ersten verschoben werden soll. Auf Grundlage der Gleichung 2.5 wissen wir, wie viel Zeit zwischen zwei Samples vergeht. Weiterhin kennen wir die Positionen der Mikrophone und können den Kopf des Roboters als Sphäre annehmen. Mithilfe dieser Informationen lässt sich die maximale Zeit bestimmen, die die Schallwellen von einem zum anderen Mikrophon benötigen. Um ein wenig Spielraum zu besitzen wurde der Offset um den verschoben werden soll auf 30 Sample festgelegt. Dies ermöglicht einen maximalen bestimmbaren Zeitunterschied von 684.9 μs.

Die Correlation wird dabei zwischen den Mikrophonen *Front-Left* und *Front-Right* bestimmt. Wie oben beschrieben ist aufgrund der Architektur des Naos eine solide Nutzung des hinteren Mikrophons aufgrund der in der Nähe liegenden Bauteile nicht möglich. Die aus den verbleibenden Mikrophonpaaren gewonnenen Daten werden in das nächste System eingegeben.

```
1  class CorrelationAnalysis():
2      """ A Class to calculate the Cross Correlation between two Vectors """
3
4      def __init__(self, vect1, vect2, abweichung):
5          self.offset = abweichung
6          self.vector1 = [0 for i in xrange(abweichung)] + vect1 + [0 for i in xrange(abweichung)
               ]
7          self.vector2 = [0 for i in xrange(abweichung)] + vect2 + [0 for i in xrange(abweichung)
               ]
8
9
10         self.vector1 = self.normalizeVector(self.vector1)
11         self.vector2 = self.normalizeVector(self.vector2)
12
13         self.correlation = self.generateCorrelationVector()
14
15
16     def generateCorrelationVector(self):
17         """ This Method generates the Correlation Vector based on a given method """
18         return self.correlate()
19
20     def correlate(self):
21         """ This method uses the build in SciPy tools but makes the CC only over the
               interesting elements """
22         self.vector2 = deque(self.vector2)
23         self.vector2.rotate(-self.offset-1)
24         v = []
25         for i in xrange(0,self.offset*2+1):
26             self.vector2.rotate(1)
27             v.append(list(S.correlate(np.array(self.vector1), np.array(self.vector2), mode = '
                   valid'))[0])
28         return v
```

Quellcode 4.4: Cross Correlation

4.6 Künstliches Neuronales Netz

Nachdem die Correlationsanalyse die beiden Werte für die Interaural Time Difference ermittelt hat, können diese dem Künstlichen Neuronalen Netz präsentiert werden. Bei dem zugrunde liegenden Neuronalen Netz handelt es sich um ein Feedforward Netz, welches die Eingabe (ITD) auf eine entsprechende Ausgabe (Winkel in der Horizontalen der Geräuschquelle) abbildet. Bei der Wahl der Anzahl der Eingabeneuronen und der Anzahl der Ausgabeneuronen sind wenig Variationen möglich. Wir benötigen zwei Eingabeneuronen um jeweils einen ITD Wert in das Neuronale Netz zu speisen. Weiterhin benötigen wir 24 Ausgabeneuronen, da wir 24 verschiedene Positionen an dem sich die Geräuschquelle befinden kann, unterscheiden wollen. Interessant wird es bei der Wahl der Anzahl der Neuronen in der versteckten Schicht. Die dort gewählte Anzahl an Neuronen kann einen großen Einfluss auf das Verfahren haben (vlg. [20]).
Ich werde daher in meinem Versuch verschiedene Werte für die Anzahl von versteckten Neuronen verwenden und deren Ergebnisse betrachten. Grundlegend kann jedoch folgende Argumentation zur Wahl der Anzahl der Neuronen in der versteckten Schicht herangezogen werden:
In Abbildung 4.1 sieht man eine Beispielverteilung der von beiden Mikrophonpaaren ermittelten ITDs im 2-dimensionalem Raum.

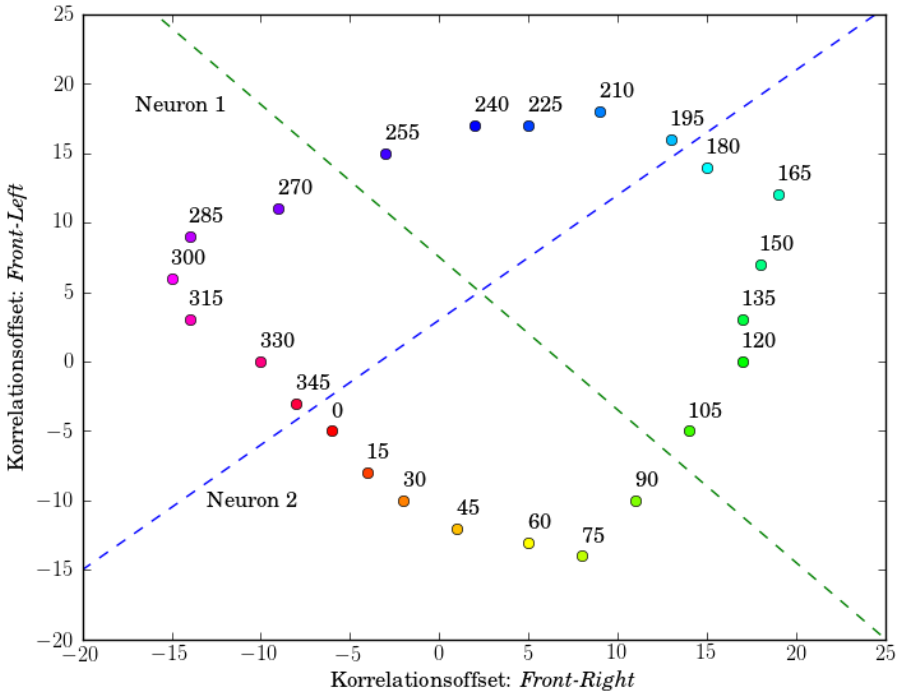


Abbildung 4.1: Entscheidungsgrenzen durch versteckte Neuronen

Jede Farbe repräsentiert dabei eine andere Position, an der sich die Geräuschquelle befindet. Jedes versteckte Neuron ist nun in der Lage, mit einer geraden Linie den Raum in zwei Klassen einzuteilen. Die beiden Neuronen in der Grafik unterteilen den Raum in 4 verschiedene Bereiche. Wenn man davon ausgeht, dass die Geraden stets durch den Ursprung verlaufen, lassen sich mithilfe von m versteckten Neuronen genau $2 \cdot m$ Klassen unterscheiden. Um das Neuronale Netz beim Lernen flexibler zu halten, wird jedoch ein Biasneuron hinzugefügt, wodurch die Linien nun nicht mehr durch den Ursprung laufen müssen. Das Neuronale Netz muss zwischen 24 verschiedenen ITD-Paaren unterscheiden. Eine Abgrenzung zu allen anderen Klassen erfolgt dabei mit zwei Geraden. Es werden also 48 versteckte Neuronen als Richtwert genutzt.

Das Training des Neuronalen Netzes erfolgt mit dem einfachen Backpropagation Trainingsalgorithmus. Die ITD Paare werden jeweils in eine Trainingsmenge und eine Testmenge aufgeteilt. Das Verhältnis zwischen Trainings- und Testmenge wird dabei ungefähr bei 60:40 liegen [14].

4.7 Effektoren

Sobald das Neuronale Netz bestimmt hat, wo sich die Geräuschquelle befindet, wird die Position der Geräuschquelle dem Benutzer zunächst sprachlich mitgeteilt. Eine Bewegung des Kopfes in die entsprechende Richtung ist auch möglich. Dazu werden die vom Hersteller mitgelieferte Bewegungsfunktionen genutzt.

Kapitel 5

Experiment

Um das Verfahren zu testen und seine Richtigkeit zu verifizieren wurden zwei zeitlich und räumlich unterschiedliche Versuchsreihen betrachtet. Die Versuchsdurchführung bestand dabei aus dem Sammeln von Audiodaten an einem 360° Kreis, der im Abstand von 15° Markierungen aufwies. Der Roboter wurde in die Mitte dieses Kreises - mit einem Radius von 1 Meter - aufgestellt. Das vordere Mikrophon war in Richtung 0° ausgerichtet. Der im folgenden verwendete Winkel bezieht sich immer auf diese 0°-Position und steigt bei der Draufsicht auf den Roboter mit dem Uhrzeigersinn. In Abbildung 5.1 findet sich ein Foto des zweiten experimentellen Aufbaus. Bei dem Raum, in dem beide Experimente durchgeführt wurden, handelte es sich um einen simuliertes Wohnzimmer (engl. Homelab) in dem Lüfterrauschen und leise Nebengeräusche von anderen arbeitenden Studenten zugegen waren. Aus den aufgenommenen Daten wurden die ITD Paare extrahiert. Diese wurden anschließend dreimal in je Trainingsdaten- und Testdaten aufgeteilt, sodass für jedes Training drei unterschiedlich zufällig gewählten Teilmengen aus der gesamten Datenmenge verwendet werden konnten.

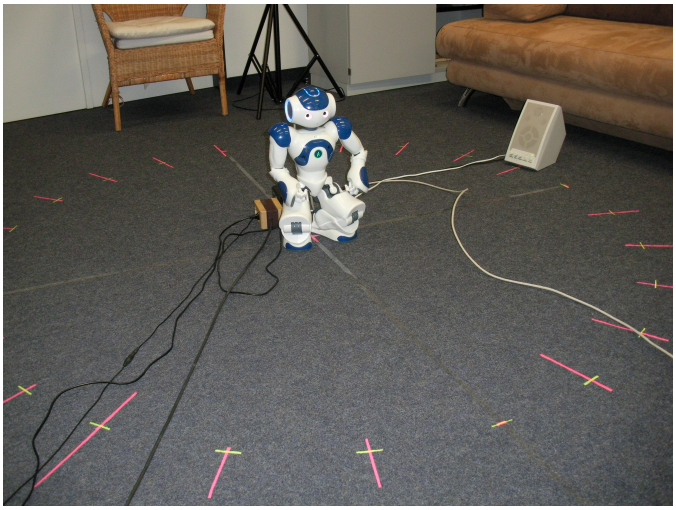


Abbildung 5.1: Experimenteller Aufbau

5.1 Erster Test

Der erste Test wird mit dem Vorspielen, einer einfachen Audiodatei umgesetzt. Es handelt sich bei der Aufnahme um den Anfang der Rede von John F. Kennedy zum bevorstehenden Raumfahrtprogramm an der Rice University in Houston, Texas am 12. September 1962 [10]. In Abbildung 5.2 ist dazu der Amplitudenverlauf, der Aufnahme abgebildet, die verwendet wurde. Die Datei war im MP3 Format mit 64 Kbps. Die markierten Teile der Rede wurden für die Extraktion der ITD Paare genutzt.

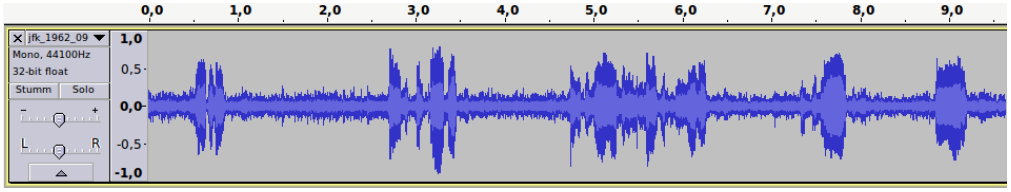


Abbildung 5.2: Rede: *"We choose to go to the moon."*

Dieser kurze Ausschnitt seiner Rede wurde aus 24 verschiedenen Richtungen im Abstand von jeweils 15° rund um den Roboter herum aufgenommen. Aus diesen Daten wurden für die signifikanten Bereiche (siehe 5.2) jeweils die ITD Paare bestimmt. Die schattierten Abschnitte 2 und 3 wurden bei der Extraktion aufgrund ihrer Länge jeweils in zwei Bereiche aufgeteilt. Es liegen somit für jede Richtung 7 ITD Paare vor. Diese wurden anschließend jeweils dreimal in eine Trainings- und eine Testmenge zerlegt. Die Trainingsmenge umfasste dabei 4 ITD Paare, die Testmenge 3. In Abbildung 5.3 ist die gesamte Datenmenge vor der Aufteilung in Trainings- und Testdaten abgebildet. Die Farbe der Datenpunkte gibt dabei den Winkel an, dem die extrahierten ITD Paare angehören. Mithilfe der generierten Trainingsdaten (96 ITD Paare) wurde verschiedene Neuronale Netze trainiert, die sich durch ihre interne Struktur unterschieden. Zu Beginn lag die verwendete Lernrate bei 0.9, um eine schnelle und grobe Anpassung an die Daten zu ermöglichen. Das Netz wurde dann in 20 Zyklen je 1200 mal mit jeweils einem zufällig gewähltem Datum der Trainingsmenge trainiert. Nach jedem Zyklus wurde die Lernrate um 0.0125 gesenkt, um eine feinere Anpassung des Neuronalen Netzes zu gewährleisten. Die Wahl der Lernrate zu Beginn und der Anzahl der Zyklen wurde aus empirischen Gründen gewählt.

Nachdem alle Trainingsdurchläufe abgeschlossen waren, wurde mithilfe der Testdaten und einer Confusion Matrix die Güte des Neuronalen Netzes bestimmt. Diese Vorgehensweise erfolgte für alle drei Teilmengen aus dem gesamten Datensatz. Die anschließend gewonnenen Ergebnisse wurden gemittelt. In Abbildung 5.4 findet

sich jeweils die Anzahl der Neuronen in der versteckten Schicht, die durchschnittliche Erfolgsquote über drei Testläufe und die empirische Laufzeit[1].

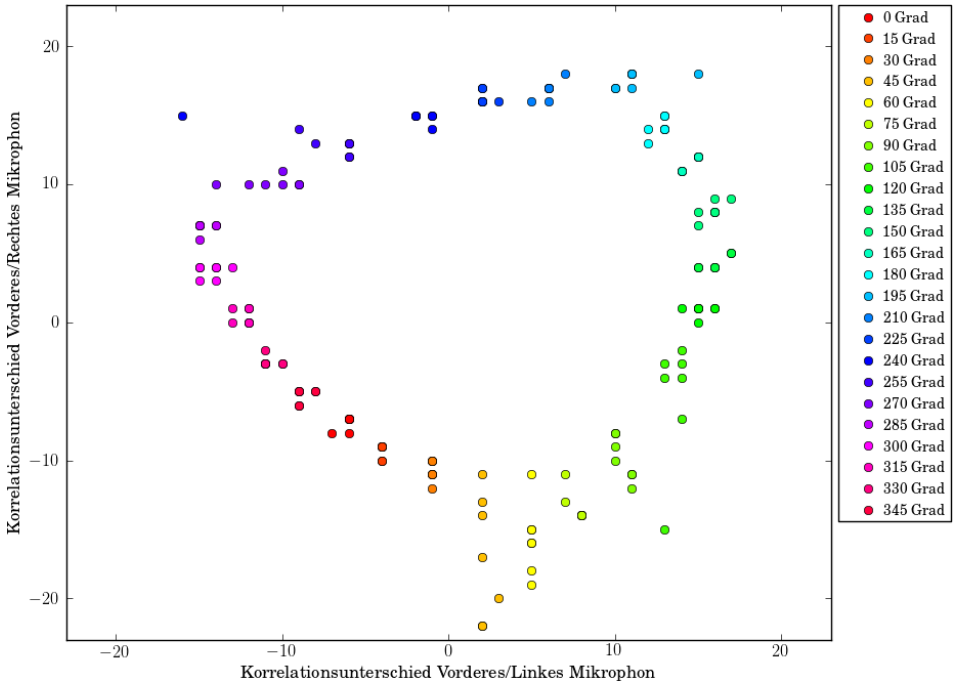
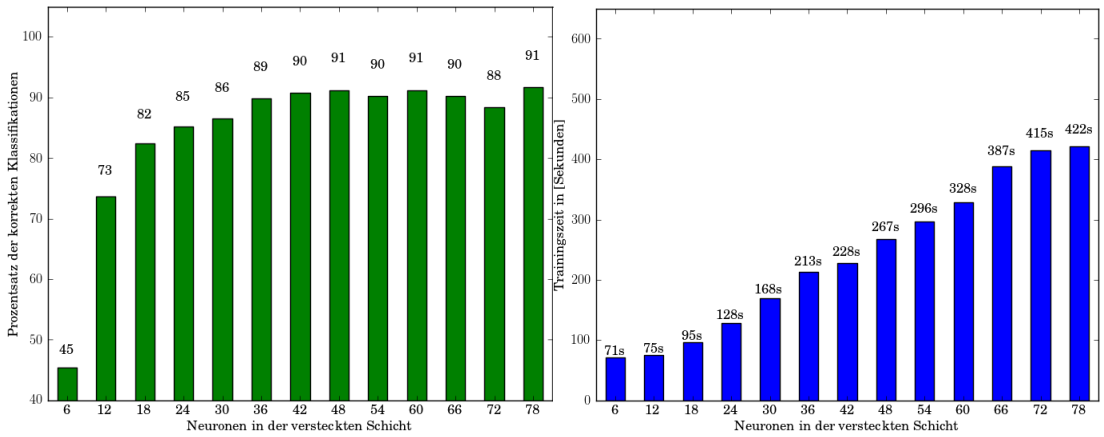


Abbildung 5.3: Extraktion - *Kennedy*

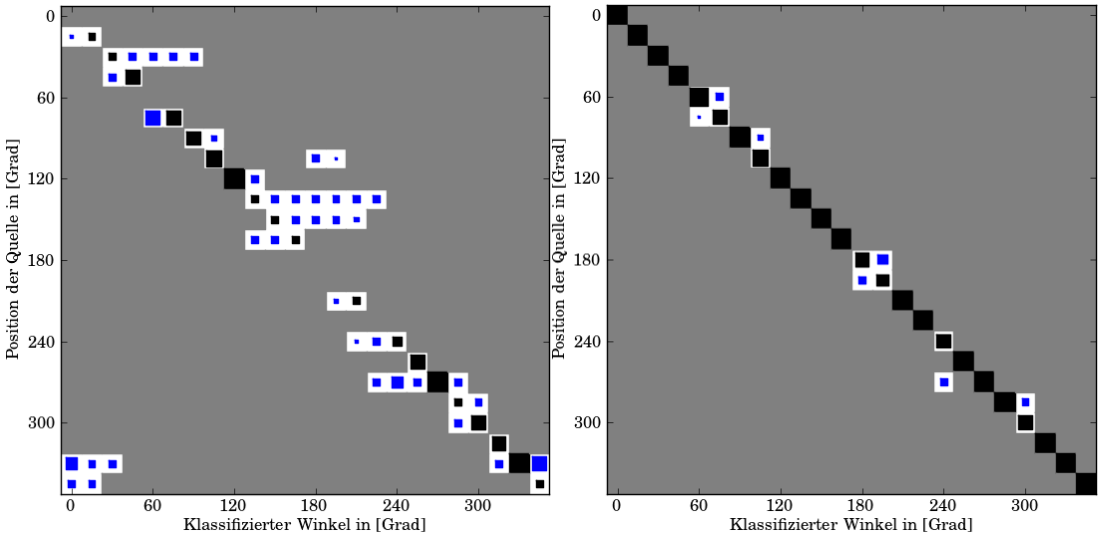


(a) Erfolg der Neuronalen Netze (b) Trainingsdauer der Neuronalen Netze

Abbildung 5.4: Ergebnisse - Datenmenge *Kennedy*

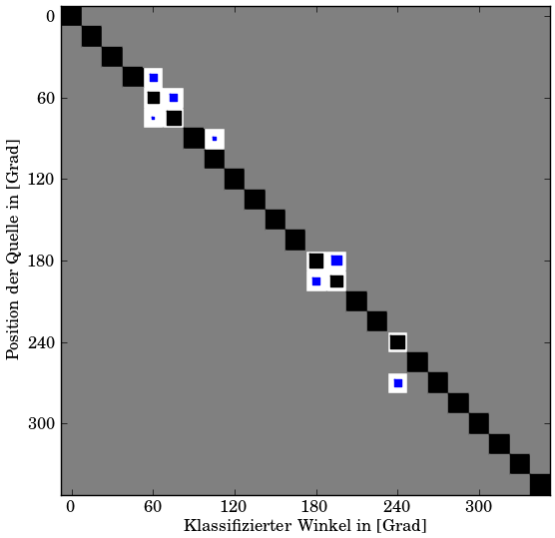
[1]Die Empirische Laufzeit wurde durch die Zeit bestimmt, die das Training des Neuronalen Netzes in Anspruch nahm. Dabei wurde jedes Training auf einem Lenovo K320-i7 durchgeführt. Die technischen Daten dazu finden sich im Anhang.

In Abbildung 5.5 sind die Confusion Matrizen der Neuronalen Netze mit 6, 48 und 78 Neuronen in der versteckten Schicht dargestellt. Es wurden gerade diese Matrizen gewählt da sie als Repräsentant für wenig, mittel und viele versteckte Neuronen in dieser Versuchsreihe auftauchen.



(a) 6 versteckte Neuronen

(b) 48 versteckte Neuronen



(c) 78 versteckte Neuronen

Abbildung 5.5: Confusion Matrizen - *Kennedy*

5.2 Versuch mit einfachen Wortensequenzen

In der nächsten Versuchsreihe wurde das Neuronale Netz mit wesentlich mehr Daten trainiert. Es wurden dafür die Worte *Hello*, *Fish*, *Look Here*, *Coffee* und *Tea* aufgenommen. Es ergaben sich aus den Aufzeichnungen nach der Extraktion insgesamt 26 Datenpaare (Abbildung 5.6). Diese Daten wurden erneut 3 mal zufällig in Trainingsdaten (16 Elemente) und Testdaten (10 Elemente) aufgeteilt. Anschließend wurden erneut verschiedene Neuronale Netze trainiert. Diesmal wurden insgesamt 32 Zyklen verwendet - ansonsten unterscheidet sich das Vorgehen jedoch nicht zum zuvor beschriebenen. Mittels einer Confusion Matrix wurden erneut die Neuronalen Netze in ihrer Güte eingeordnet. Die Ergebnisse finden sich in Abbildung 5.7. In Abbildung 5.8 sind die Confusion Matrizen der jeweiligen Neuronalen Netze dargestellt.

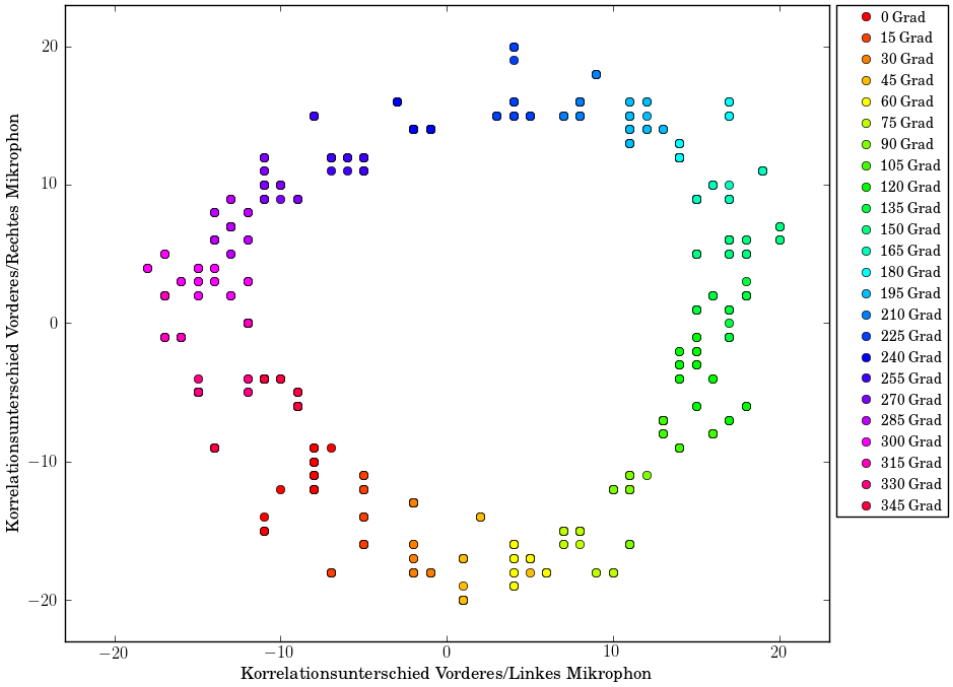
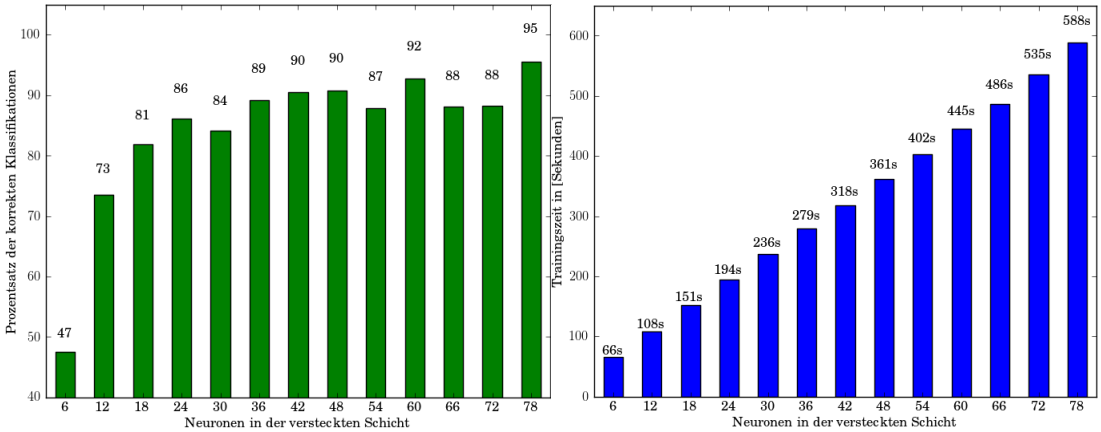


Abbildung 5.6: Extraktion - *Wortsequenzen*



(a) Erfolg der Neuronalen Netze (b) Trainingsdauer der Neuronalen Netze

Abbildung 5.7: Ergebnisse - Datenmenge *Wortsequenzen*

5.3 Vorverarbeitung mit Rauschreduzierung

Zuletzt wurde mittels einem einfachen Verfahren zur Rauchunterdrückung der Versuch unternommen, die Qualität des Verfahrens noch zu verbessern. Das Trainieren des Neuronalen Netzes wurde jedoch nach der Betrachtung der extrahierten Daten fallen gelassen, da sich die generierten ITD Paare ersichtlicherweise nicht zur weiteren Nutzung eigneten. Abbildung 5.9 zeigt die dabei extrahierten Daten bei zuvor erfolgtem reduzieren des Rauschens.

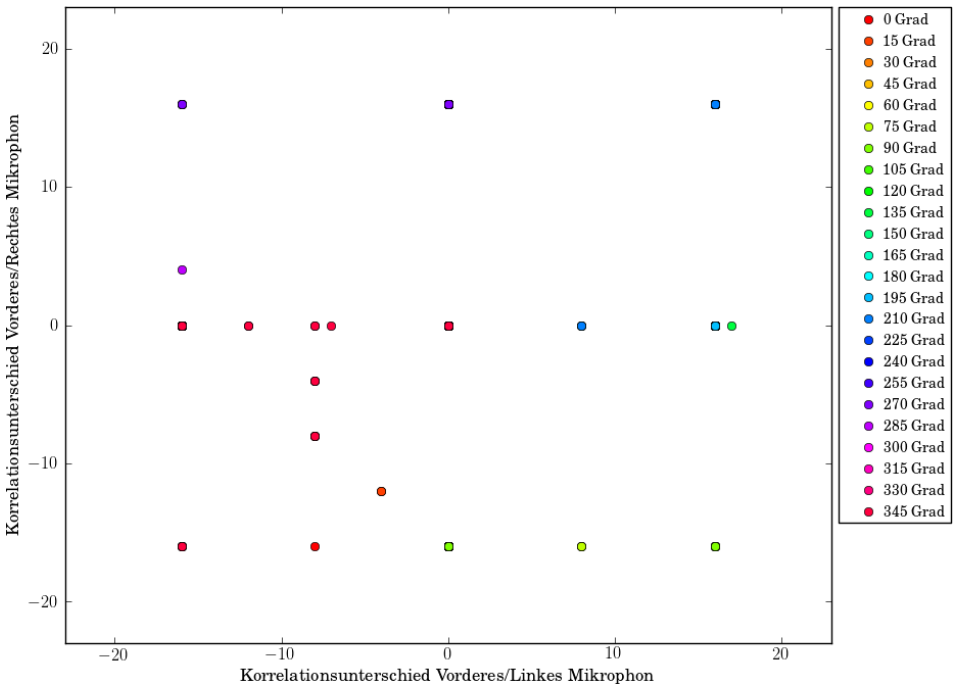
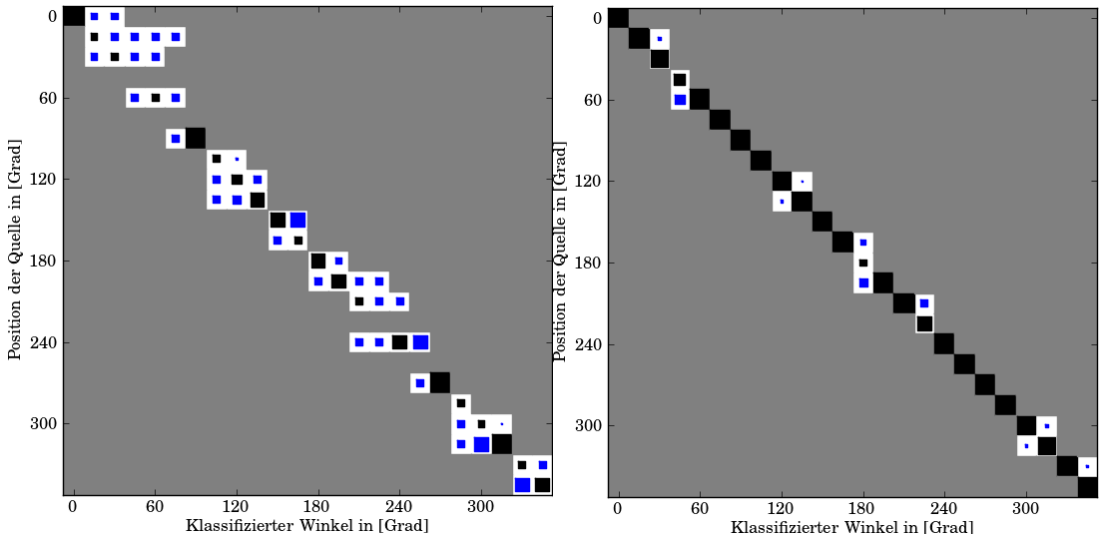
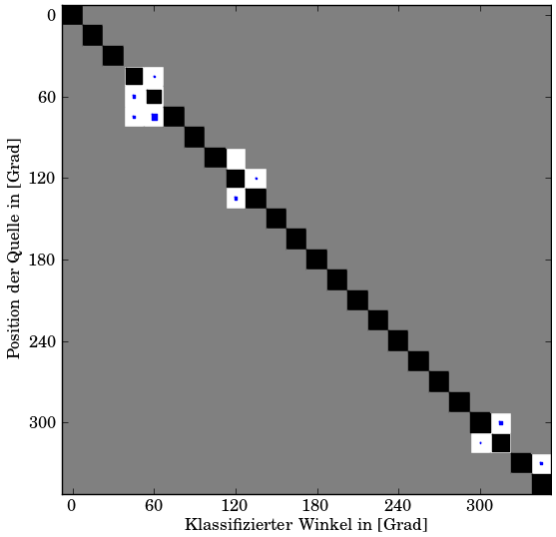


Abbildung 5.9: Noisy Words Extraction



(a) 6 versteckte Neuronen

(b) 48 versteckte Neuronen



(c) 78 versteckte Neuronen

Abbildung 5.8: Confusion Matrizen - *Wortsequenzen*

5.4 Zusammenfassung

Mein Verfahren wurde zweimal mit verschiedenen Daten getestet. Dabei hat sich gezeigt, dass die Lokalisation bei wenigen Neuronen in der versteckten Schicht nicht zuverlässig funktioniert. Auffallend ist jedoch der Anstieg der Erfolgsquote, sobald die Anzahl der versteckten Neuronen steigt. Gleichzeitig bedarf es für das Training mit zunehmender Zahl an Neuronen eine längere Trainingszeit. Es muss also ein gutes Mittelmaß zwischen der Trainingsdauer und der Erfolgsquote gefunden werden. Die Versuche das Verfahren noch zu verbessern, indem eine Rauschreduzierung angewandt wurde hat sich als nicht tragbar erwiesen. Dies liegt daran, dass die Methode zur Rauschreduktion auch den Informationsgehalt des Sprachsignals vermindert. Dadurch ist eine stabile Bestimmung der Cross Correlation nicht mehr möglich.

Kapitel 6

Fazit

Die Problemstellung dieser Arbeit bestand darin, zu untersuchen wie ein humanoider Roboter eine Geräuschquelle lokalisieren kann. Der vorgestellte Ansatz ermittelt dabei zunächst die Interaural Time Difference mithilfe von Cross Correlation. Anschließend wurde die Geräuschquellenposition mit einem trainierten Künstlichen Neuronalen Netz berechnet.
Die experimentellen Ergebnisse zeigen eine gute Erfolgsquote für das vorgestellte Verfahren. Die zuvor aufgestellten Bedingungen werden durch das Experiment bestätigt. Der Roboter ist in der Lage eine Lokalisation der Geräuschquellen am vollen 360 Grad Kreis vorzunehmen und somit auch zuverlässig die Front-Back-Confusion aufzulösen. Die Erfolgsquote der Klassifikation ist, wie die Experimente gezeigt haben, dabei von der Anzahl der Neuronen in der versteckten Schicht des Netzes abhängig.
Probleme traten hingegen besonders bei der Vorverarbeitung - dem Reduzieren von Rauschen - auf. Auch ist die Qualität der Mikrophone des Roboters eher als gering einzustufen. Geräusche, die weiter als ca. 1 Meter von den Mikrophonen entfernt sind, sind nur noch schwach zu vernehmen.

Eine Verbesserungsmöglichkeit wäre es, die Cross Correlation mit einem geeigneten Verfahren zu beschleunigen. Da eine Reihe von Vektoren auf Ähnlichkeit geprüft werden, ist eine Parallelisierung der Berechnung ein möglicher Ansatz.
Das Neuronale Netz bietet weiterhin eine schnelle Adaption auf eine neue Umgebung. Es werden dafür jedoch erneut Trainingsdaten benötigt. Ein Verfahren, das neben der Nutzung weitere verlässliche Trainingsdaten sammeln würde, wäre eine sinnvolle Weiterentwicklung meines Ansatzes. So wäre der Roboter problemlos in der Lage, sich auf neue ihm unbekannte Lokalitäten einzustellen.
Auch eine Datenbasis von unterschiedlichen Neuronalen Netzen für unterschiedliche Situationen und Orte wäre eine mögliche Anpassung, um eine höhere Genauigkeit, gerade in größeren Gebäudekomplexen zu erreichen.

Letztendlich zeigt die Kombination von Korrelationsanalyse und der anschießenden Winkelbestimmung mithilfe von Neuronalen Netzen einen sinnvollen und hinreichend guten Ansatz für den Einsatz auf einem humanoidem Roboter.

Desktop Computer
Lenovo IdeaCentre K320
Serial - ES06527419

Operating System
Linux 2.6.38-12-generic #51-Ubuntu SMP Wed Sep 28 14:27:32 UTC 2011
x86_64 x86_64 x86_64 GNU/Linux

CPU
Intel(R) Core(TM) i7 CPU 870 @ 2.93GHz

Size	1197MHz
Capacity	2930MHz
Width	64 bits
Clock	533MHz
Cores	4
Enabledcores	4
Threads	2
L1 Cache	32KiB
L2 Cache	256 KiB
L3 Cache	8MiB

RAM

Slot 1 - 2GiB DIMM DDR3 Synchronous 1066 MHz (0.9 ns) 64 bits
Slot 2 - 2GiB DIMM DDR3 Synchronous 1066 MHz (0.9 ns) 64 bits
Slot 3 - 2GiB DIMM DDR3 Synchronous 1066 MHz (0.9 ns) 64 bits
Slot 4 - empty

NAO H25 DATASHEET

ALDEBARAN Robotics

NAO^{H25} is a trusted platform for education and research in various topics, from robotics and computer science to autism and human-robot interaction. NAO^{H25} is ALDEBARAN Robotics' most advanced robot. This fully-featured humanoid robot provides an open platform with full integration of state-of-the-art hardware and softwares. NAO^{H25} is robust, interactive and easy to use allowing you to focus on your core research.

GENERAL FEATURES

BODY CARACTERISTICS

HEIGHT: ~58 CM - 22.8''

WEIGHT: ~5 KG - 11LB

BODY MATERIAL: ABS - PC

ENERGY

CHARGER: AC 90-230 volts / DC 24 volts

BATTERY CAPACITY: ~90 min. autonomy

DEGREES OF FREEDOM

HEAD: 2 DOF

ARM: 4 DOF in each arm

PELVIS: 1 DOF

LEG: 5 DOF in each leg

HAND: 2 DOF in each hand

MULTIMEDIA

SPEAKERS: 2 Loudspeakers

MICROPHONES: 4 Microphones

VISION: 2 CMOS Digital Cameras

NETWORK ACCESS

CONNECTION TYPE:
- WI-FI (IEE 802.11 b/g)
- Ethernet Connection

ACTUATORS

ALDEBARAN ROBOTICS ™
ORIGINAL DESIGN BASED ON:
- Hall effect sensors
- dsPICS microcontrollers

SENSORS

- 36 x Hall effect sensors
- 2 x gyrometer 1 axis
- 1 x accelerometer 3 axis
- 2 x bumpers
- 2 x sonar channels
- 2 x I/R
- Tactile sensors (head, hands)
- 8 x FSRs

MOTHERBOARD

- x86 AMD GEODE 500MHz CPU
- 256MB SDRAM / 2GB Flash Memory

LED

TACTILE SENSOR: 12 LEDs 16 Blue levels

EYES: 2 x 8 LEDs RGB Fullcolour

EARS: 2 x 10 LEDs 16 Blue levels

TORSO: 1 LED RGB Fullcolour

FEET: 2 X 1 LED RGB Fullcolour

SOFTWARE COMPATIBILITIES

OS: Embedded Linux (32bit x 86 ELF) using custom OpenEmbedded based distribution

PROGRAMMING LANGUAGES:
C++, Urbi script, Python, .Net

Literaturverzeichnis

[1] S. F. Boll. Suppression of acoustic noise in speech using spectral subtraction. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 27(2):113–120, 1979.

[2] Y. Cho, D. Yook, S. Chang, and H. Kim. Sound source localization for robot auditory systems. *Transactions on Consumer Electronics, IEEE*, 55(3), August 2009.

[3] A. Czyzewski. Automatic identification of sound source position employing neural networks and rough sets. *Pattern Recognition Letters*, 24(6):921–933, March 2003.

[4] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier. Mechatronic design of nao humanoid. *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009.

[5] K. Y. Guentchev and J. J. Weng. Learning-based three dimensional sound localization using a compact non-coplanar array of microphones. In *AAAI SYMPOSIUM ON INTELLIGENT ENVIRONMENTS*, 1998.

[6] J. Key C. Schauer C. Schröter H. Gross T. Hempel H. Böhme, T. Wilhelm. An approach to multi-modal human-machine interaction for intelligent service robots. *Robotics and Autonomous Systems*, 44(1):83–96, 2003.

[7] W. M. Hartmann. How we localize sound. *Physics Today*, 52(11):24–29, November 1999.

[8] P. Stoica J. Li. *Robust Adaptive Beamforming*. John Wiley & Sons, Inc., 2005.

[9] L. A. Jeffress. A place theory of sound localization. *Journal of comparative and physiological psychology*, 41:35–39, 1948.

[10] John F. Kennedy. http://www.archive.org/download/jfks19620912/jfk_1962_0912_spaceeffort_64kb.mp3. Presidential Library, 9 1962. Last Access 28.11.2011 12:58.

[11] S. Lee, S. Hwang, Y. Park, and Y. Park. Sound source localization in median plane using artificial ear. *International Conference on Control, Automation and Systems*, 2008.

[12] H. Liu and M. Shen. Continuous sound source localization based on microphone array for mobile robots. In *International Conference on Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ*, 2010.

[13] J. Liu, D. Perez-Gonzalez, A. Rees, H. Erwin, and S. Wermter. A biologically inspired spiking neural network model of the auditory midbrain for sound source localisation. *Neurocomputing*, 74:129–139, December 2010.

[14] S. Marsland. *Machine learning: an algorithmic perspective*. Chapman & Hall/-CRC machine learning & pattern recognition series. CRC Press, 2009.

[15] R. Martin, U. Heute, and C. Antweiler. *Advances in Digital Speech Transmission*. John Wiley & Sons, Ltd, 2008.

[16] J. C. Murray, H. Erwin, and S. Wermter. Robotic sound-source localisation architecture using cross-correlation and recurrent neural networks. *Neural Networks*, 22(2):173–189, 2009.

[17] M. Obando, L. Liem, W. Madauss, M. Morita, and B. Robinson. Robotic surgery in pituitary tumors. *Operative Techniques in Otolaryngology-Head and Neck Surgery*, 15(2):147–149, 2004.

[18] Aldebaran Robotics. User guide version 1.10.10. Digital Manual.

[19] T. Rodemann. A study on distance estimation in binaural sound localization. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference*, 2010.

[20] R. Rojas. *Neural networks: a systematic introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1996.

[21] M. Russ. *Sound Synthesis and Sampling*. Focal Press, 2004.

[22] C. Sammut and G. I. Webb. *Encyclopedia of Machine Learning*. Springer, 2010.

[23] A. Spanias, T. Painter, and V. Atti. *Audio Signal Processing and Coding*. John Wiley & Sons, Inc., 2005.

Abbildungsverzeichnis

Erklärung der Urheberschaft

Ich versichere an Eides statt, dass ich die vorliegende Bachelorarbeit selbstständig und ohne unerlaubte Hilfe Dritter angefertigt habe. Alle Stellen, die inhaltlich oder wörtlich aus anderen Veröffentlichungen stammen, sind kenntlich gemacht. Diese Arbeit lag in gleicher oder ähnlicher Weise noch keiner Prüfungsbehörde vor und wurde bisher noch nicht veröffentlicht.

Ort, Datum Unterschrift