Institut für Theoretische Informatik
Fakultät für Informatik
Universität Karlsruhe (TH)

Interactive Systems Labs

Sprecherlokalisierung und Eingabefusion für einen humanoiden Roboter

Studienarbeit

von

Matthias Walliczek

betreut von
Prof. Dr. Alexander Waibel
Dipl.-Inform. Hartwig Holzapfel
Dr. Thomas Schaaf

September 2005

b

Danksagungen

Ich möchte mich insbesondere bei Professor Dr. Waibel bedanken, der mir durch diese Studienarbeit Gelegenheit gegeben hat, an der Forschung in einem interessanten Bereich teilzunehmen und mir ermöglicht hat, diese Arbeit an der Carnegie Mellon University in Pittsburgh anzufertigen. Ebenfalls möchte ich mich beim InterACT-Stipendienprogramm bedanken, das mir die Reise in die USA finanziert hat. Bedanken möchte ich mich ferner bei meinen Betreuern Hartwig Holzapfel und Dr. Thomas Schaaf, die mich fachlich unterstützt haben und mich auch innerhalb der USA betreut haben. Dirk Bechler danke ich für die Unterstützung bei der Arbeit mit seinem Lokalisierungsprogramm. Mein Dank gilt außerdem meinen Eltern und der Mensarunde für die emotionale Unterstützung.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Soziale Interaktion ist für humanoide Roboter essentiell, da sich derartige Roboter immer weiter in sozialen und häuslichen Umgebungen ausbreiten. Sie werden dort als Spielzeuge im Wohnzimmer, als Dienstboten im Büro oder als Diener bei Parties eingesetzt [BBI+98]. Für diese sozialen Aufgaben benötigen die Roboter komplexe perzeptive Fähigkeiten, um beispielsweise Personen in einem Raum zu erkennen, ihrer Stimme zuzuhören und ihr Gesicht zu erkennen und Bild und Ton zu verbinden.

Um mit einem Menschen zu interagieren, benötigen Roboter multimodale Schnittstellen: Sie brauchen die optische Wahrnehmung, um ihre Umgebung zu erforschen und Menschen zu identifizieren und zu erkennen. Die Roboter müssen ebenfalls erkennen, ob jemand mit ihnen oder mit einer anderen Person redet. Die optische Wahrnehmung allein reicht jedoch nicht aus, für ein Erreichen dieses Ziels ist ebenfalls eine akustische Wahrnehmung notwendig, um Geräusche zu klassifizieren, zu lokalisieren und Sprache zu erkennen. Durch die Kombination dieser Modalitäten kann ein Roboter Geräuschquellen lokalisieren und die erkannte Sprache einer Person zuordnen.

Ziel dieser Studienarbeit ist es, einem Roboter zu ermöglichen, bei falschen optischen Hypothesen wieder zur korrekten Erkennung zurückzukommen. Wenn ein Roboter einem falsch erkannten optischen Track folgt, weil er zum Beispiel eine Holzfläche mit einem menschlichen Gesicht verwechselt, muss es dem Benutzer möglich sein, wieder die Aufmerksamkeit des Roboters zu erlangen. Es erscheint intuitiv naheliegend, durch eine Ansprache von Seiten des menschlichen Benutzers eine Ausrichtung des Roboters in Richtung des Benutzers zu erzwingen.

Dazu muss der Roboter die Äußerung durch Spracherkennung verstehen, mögliche Geräuschquellen lokalisieren und eine finale Hypothese bestehend aus der Sprache und der Lokalisierung berechnen.

In der vorliegenden Studienarbeit soll für den innerhalb des Sonderfor-

schungsbereichs 588 entwickelten Roboter eine Lokalisierungssoftware erstellt werden, die es dem Roboter ermöglicht, auf bestimmte Kommandos mit einer Drehung des Kopfes in Richtung des Sprechers zu reagieren. Dazu soll er über zwei im Kopf eingebaute Mikrofone die Richtung der Sprache ermitteln. Da die Spracherkennung über distance-speech-Mikrofone allerdings noch nicht mit ausreichender Zuverlässigkeit funktioniert, soll die Sprache übergangsweise durch ein close-talking-Headsetmikrofon aufgenommen werden.

Theoretisch existiert für diese Aufgabenstellung bereits eine Lösung: Innerhalb des ebenfalls am SFB 588 beteiligten Instituts für Nachrichtentechnik der Fakultät für Elektrotechnik und Informationstechnologie hat Dirk Bechler eine Lokalisierungssoftware [BK05] erstellt, die über eine Vorverarbeitung der akustischen Signale die Winkel ermittelt und in einem zweiten Schritt mögliche Quellen erfasst und verfolgt (Tracking). Bei genauerer Betrachtung stellt sich jedoch heraus, dass diese Software eher für Szenarien ohne Nebengeräusche geeignet ist und bei dem angedachten Einsatz auf Messen und anderen Szenarien mit starken Nebengeräuschen nur bedingt einsetzbar ist.

In der vorliegenden Studienarbeit wurde deshalb versucht, durch ein verbessertes Tracking basierend auf der bereits existierenden Vorverarbeitung auch in Szenarien mit starken Nebengeräuschen korrekte Ergebnisse zu ermitteln. Die Ergebnisse der Arbeit wurden in das Spracherkenner-Framework One4All integriert, als Segmentierer wurde der Segmentierer innerhalb des Janus-Spracherkenners (siehe [FGH+97] und [SMFW01]) eingesetzt.

Kapitel 2

Grundlagen der Geräuschlokalisierung

In der Natur ist eine präzise Geräuschlokalisierung für die meisten Tiere überlebenswichtig: Im dichten Unterholz, in der Nacht und in anderen schlechten Lichtverhältnissen ist das Gehör die einzige Möglichkeit für ein Beutetier, einen Jäger zu bemerken. Da es ebenso wichtig ist, neben der Anwesenheit auch den Ort des Angreifers zu ermitteln, macht sich die Natur unter anderem eine wichtige Eigenschaft von Schall zu nutzen: dessen Geschwindigkeit.

2.1 Schallgeschwindigkeit

Die Schallgeschwindigkeit in idealen Gasen ist abhängig vom Adiabatenexponent κ, der Dichte ρ sowie dem Druck p des Gases oder alternativ nach der thermischen Zustandsgleichung von der molaren Masse M und der absoluten Temperatur T (gemessen in Kelvin) und berechnet sich aus

$$c_{\text{Gas}} = \sqrt{\kappa \cdot \frac{p}{\rho}} = \sqrt{\kappa \cdot \frac{R \cdot T}{M}} \tag{2.1}$$

.

Adiabatenexponent $\kappa = c_p/c_V$.

Der Adiabatenexponent κ hängt auch für die meisten realen Gase über weite Temperaturbereiche nicht von T ab, die molare Masse ist eine materialspezifische und die universelle Gaskonstante R=8,3145 J/mol*K eine physikalische Konstante.

Deshalb hängt die Schallgeschwindigkeit in idealen Gasen nur von der Wurzel der (absoluten) Temperatur ab. Trotz der Wurzelabhängkeit wird

häufig die lineare Näherungsformel

$$c_{\text{Luft}} = (331{,}5 + 0{,}6 \cdot \vartheta)\ \text{m/s} \tag{2.2}$$

verwendet, wobei $\vartheta = T - 273{,}15\,\text{K}$ die Temperatur in °C ist. Diese Näherungsformel gilt im Temperaturbereich von -20°C bis +40°C mit einer Genauigkeit von besser als 0,2%.

Ein genauerer empirischer Ausdruck für die Schallgeschwindigkeit ergibt sich durch Zusammenfassen der Konstanten in eine einzige rechnerische Konstante:

$$c_{\text{Luft}} = \sqrt{1{,}402 \cdot \frac{R \cdot T}{0{,}02896\,\text{kg/mol}}} = 20{,}055\sqrt{\frac{T}{\text{K}}}\ \text{m/s} \tag{2.3}$$

wobei $M = 0{,}02896$ kg/mol die molare Masse und $\kappa = 1{,}402$ der Adiabatenexponent der Luft ist. Der genaue Betrag der Vorfaktoren wurde aus Messungen nach D.A. Bohn (1988) bestimmt. Mit dieser Gleichung beträgt die Schallgeschwindigkeit bei 25 °C (=298,15 K) etwa 346 m/s. Allgemeiner bekannt ist der Wert c = 343 m/s für 20 °C (Zimmertemperatur).[Wik05b]

2.2 Geräuschlokalisierung über die Schallgeschwindigkeit

Nachdem man die Geschwindigkeit des Schalls bestimmt hat, kann man den zeitlichen Abstand für das Erreichen von zwei voneinander entfernte Positionen berechnen - zum Beispiel der Position von zwei Ohren. Trifft der Schall wie in Abbildung 2.1 in einem Winkel α ungleich 90° zu den Ohren auf, dann erreicht er ein Ohr zuerst, das andere um das Produkt aus Entfernung und Schallgeschwindigkeit später. Aus der zeitlichen Differenz lässt sich also auf die Richtung schließen - allerdings nur in einem 180°-Halbkreis, der sich an b spiegelt.

2.3 Weitergehende Möglichkeiten

In der Natur werden noch zwei andere Möglichkeiten zur Geräuschlokalisierung verwendet, um einerseits volle 360° in der Medianebene und andererseits auch eine Lokalisierung in Transversalebene zu ermöglichen.

Zum Einen wirkt der Kopf zwischen den Ohren als Hindernis, das den Schall abschattet und so durch eine Pegelmessung eine verbesserte Erkennung in der Medianebene erlaubt - auf der der Geräuschquelle abgewandten

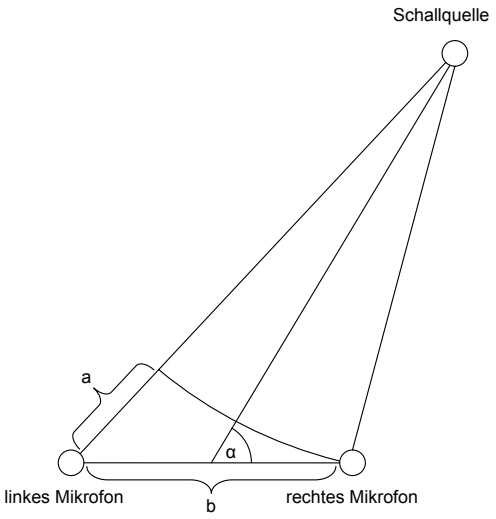

Abbildung 2.1: Geräuschlokalisierung durch die Laufzeitdifferenz

Seite hat das ankommende Signal nur einen geringeren Pegel als auf der der Geräuschquelle zugewandten Seite.

Außerdem wirkt das Außenohr des Menschen, das heißt die Ohrmuschel und der Anfang des Gehörgangs als richtungsselektives Filter. In der Struktur der Ohrmuschel werden je nach Schalleinfallsrichtung in der Medianebene unterschiedliche Resonanzen angeregt. Dieses führt dazu, dass jede dieser Richtungen (vorne, oben, hinten, unten) ein unterschiedliches Resonanzmuster besitzt. Das heißt, dass der Frequenzgang der Ohren richtungsspezifische Muster eingeprägt bekommt, die vom Gehör-Gehirn-System ausgewertet werden.

Diese Muster im Frequenzgang sind individuell, je nach Form und Größe der eigenen Ohrmuschel. Bekommt man Schall über Kopfhörer dargeboten, der von einem anderen Kopf mit anderen Ohrmuscheln aufgenommen wurde, wird die Erkennung der Richtung in der Medianebene nicht mehr problemlos möglich. Beispiel: Das überwiegende Hinten-Lokalisieren von Kunstkopfaufnahmen und die Im-Kopf-Lokalisierung (IKL).[Wik05a]

2.4 Geräuschlokalisierung per Computer

Theoretisch könnte auch bei der computergestützen Geräuschlokalisierung auf alle drei Möglichkeiten zugegriffen werden: Auf die Laufzeitunterschiede, Pegelunterschiede und spektrale Unterschiede. Die beiden letztgenannten Möglichkeiten erfordern jedoch eine intensive Anpassung an die verwendete

Hardware und den Aufbau. Außerdem fehlt je nach Aufbau ein Hindernis, das den Schall zwischen beiden Seiten abschattet. Aus diesen Gründen wird deshalb häufiger auf eine reine Laufzeitmessung zurückgegriffen, für die eine minimale Anpassung ausreicht; lediglich Temperatur und Mikrofonabstand müssen eingestellt werden.

Kapitel 3

Vergleichbare Arbeiten

Für die Aufgabenstellung „Akustische Lokalisierung und Zielverfolgung“ existieren bereits zahlreiche Arbeiten, die innerhalb der Studienarbeit geprüft wurden. Grundsätzlich ergibt sich jedoch das Problem, dass fast alle Arbeiten für Mikrofon-Arrays konzipiert sind und damit bessere akustische Eingangssignale erwarten. Außerdem sind nur in wenigen Arbeiten mehrere parallele Geräuschquellen vorgesehen, von denen eine Quelle ausgewählt werden soll.

3.1 Lokalisierung ohne Tracking

In [VMRL03] wird beispielsweise ein Ansatz vorgestellt, der mit einem 8-kanaligen 3-dimensionalen Mikrofon-Array arbeitet und dabei eine Präzision von 3° bei einer Entferung von 3 Metern erreicht. Allerdings reagiert dieses System grundsätzlich nur auf die lauteste Geräuschquelle und kann daher weder Daten vom Sprachsegmentierer berücksichtigen noch nicht-sprachliche Geräusche ausfiltern.

3.2 Tracking mit rekursiven Filtern

Zahlreiche Arbeiten greifen für das Tracking auf rekursive Filter zurück. Bei einem rekursiven Filter werden Ausgaben des Filters als neue Eingaben verwendet. Diese Rückkopplung führt zu einer nicht abgeschlossenen Impulsantwort, die entweder exponentiell wachsende, gleichbleibende oder sinusförmige Anteile enthält.

Ein Beispiel für ein derartiges Filter ist das Kalman-Filter. Dieses versucht, den Zustand eines dynamischen Systems durch eine Serie von unvollständigen und verrauschten Messungen zu schätzen. Für die akustische Lokalisierung wird es beispielsweise in [KBG02] verwendet. Für Bewegungsdaten,

die sich nicht durch größtenteils lineare oder gaußförmige Modelle darstellen lassen, ist dieses Filter jedoch nicht geeignet[NLGV05].

3.3 Tracking mit Partikelfiltern

Um auch nicht-lineare und nicht-gaußförmige Datenmodelle verfolgen zu können, werden in anderen Arbeiten Partikelfilter (auch Sequentielle Monte-Carlo Algorithmen (SMC) genannt) eingesetzt (vgl. [GD99]). Ziel der SMC-Methoden ist es, den Systemzustand (also die Position der Geräuschquelle) als Funktion der Zeit auf Basis von einer Reihe Beobachtungen des Systems und a priori Kenntnissen der Systemdynamik zu schätzen. Dazu wird die komplizierte Wahrscheinlichkeitsdichte des Zustandes diskret durch eine Menge von Partikeln approximiert.

Beispielsweise werden in [NLGV05] Partikelfilter verwendet, um mehrere Geräuschquellen gleichzeitig verfolgen zu können. Für dieses System existiert jedoch nur eine Computersimulation, die zur Evaluation verwendet wurde; der praktische Einsatz steht noch aus.

In [VB01] wird dagegen ein auf SMCs basierender Ansatz vorgestellt, der auch mit zwei omni-direktionalen Microfon-Paaren getestet wurde. Dabei wird trotz Hintergrundrauschens die Bewegung der Geräuschquelle korrekt erfasst. Allerdings ist dieser Ansatz wiederum nur für die Verfolgung einer Geräuschquelle konzipiert.

Kapitel 4

Verwendete Daten

Um bei der Erfassung der Daten möglichst realistische Bedingungen entsprechend der Aufgabenstellung herzustellen, wurden zwei Sony-Mikrofone mit batteriegestützten Vorverstärkern ähnlich der späteren Anordnung auf dem Roboterkopf mit einem Abstand von 14,8 cm in jeweils entgegengesetzter Richtung auf einer Kunststoffschiene montiert. Sie waren an ein Mischpult angeschlossen, wurden dort weiter verstärkt und zu einem Stereosignal gemischt. Das Signal wurde über den Line-In-Eingang eines Laptops mit dem Programm „Audacity“ mit einer Samplingrate von 16.000 Hz und einer Auflösung von 16 bit aufgenommen.

Bei späteren Aufnahmen wurde zusätzlich mithilfe einer Videokamera parallel eine Videoaufnahme angefertigt, um die Position und Bewegung der Sprecher aufzuzeichnen.

4.1 Aufnahmenszenarien

Die ersten Test-Aufnahmen wurden im Poolraum im Karlsruher Institutsgebäude gemacht. Dabei wurde zu Beginn mit vielen Nebengeräuschen und mehreren gleichzeitig sprechenden Personen und anschließend ohne Nebengeräusche experimentiert. Gesprochen wurden Phrasen, auf die der Roboter später reagieren soll: „Hi Robbi!“, „Schau hier her!“, „Schalte das Licht aus“ usw. Für die spätere Annotation wurden ebenfalls die geschätzen Winkel im Bezug zu den Mikrofonen aufgenommen. Die Aufnahmen hatten eine Dauer von 2 bis 4 Minuten.

Am 1.3.2005 wurde eine neue Aufnahme in Pittsburgh, ebenfalls im Poolraum im alten Institutsgebäude gemacht. Um die Positionen besser festzuhalten und die Annotation zu erleichtern, wurde hier eine Videokamera verwendet. Diese wurde hinter dem Mikrofonstativ positioniert, um dieses und

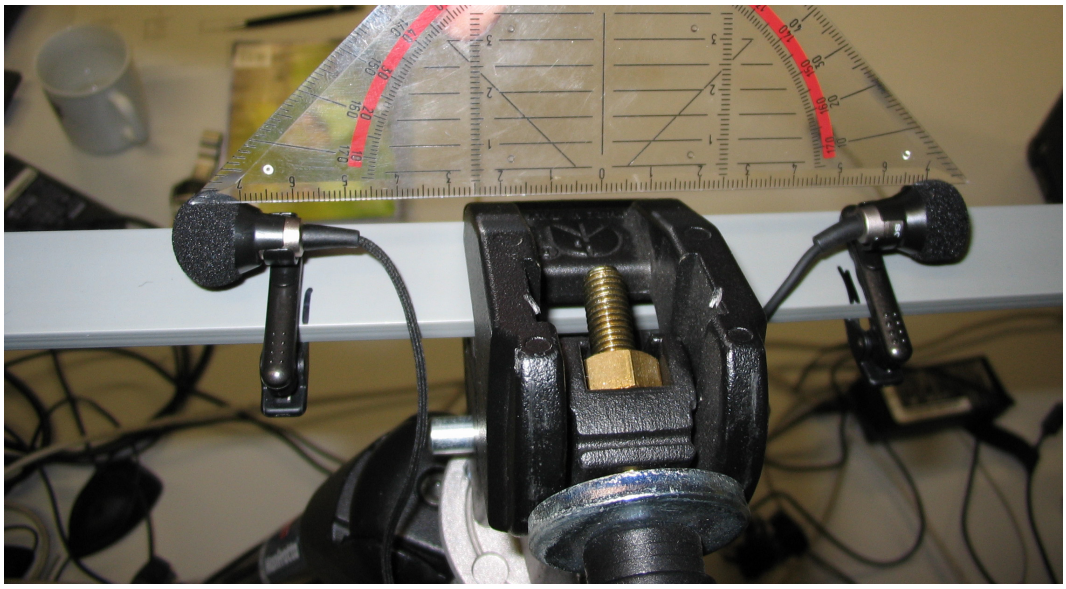

Abbildung 4.1: Aufbau des Aufnahmeszenarios

einen möglichst großen Bereich darum herum erfassen zu können. Zunächst wurden auch hier einzelne Befehle von wechselnden Positionen aufgenommen sowie anschließend ein Gespräch zwischen drei Personen mit teilweise wechselnden Positionen. Die Dauer der Aufnahme betrug 13 Minuten. Eine weitere Aufnahme wurde am 3.3.2005 gemacht. Dabei wurde eine Diskussion zwischen zwei Personen aufgenommen, die an einem Tisch saßen und sich meistens nur geringfügig bewegten. Die Dauer dieser Aufnahme betrug 20 Minuten.

Eine letzte Aufnahme wurde am 11.5.2005 gemacht. Das Szenario war ein 2-Personen-Büro, im wesentlichen ohne Nebengeräusche durch Computer. Aufgenommen wurde ein 11 minütiges Gespräch zwischen drei Personen.

4.2 Manuelles Markieren der Daten

Da der Sprachsegmentierer zu Beginn der Studienarbeit noch nicht zur Verfügung stand, wurden die Daten manuell annotiert. Dabei wurde zunächst die akustische Analyse Start- und Endzeitpunkt einer Sprachsequenz markiert.

Im Laufe der Analyse des Lokalisierungsprogramms (vgl. 5.1) stellte sich heraus, dass das Lokalisierungsprogramm die Winkel durchaus richtig schätzt, diese allerdings von Störgeräuschen überlagert sind. Deshalb bestand die Aufgabe darin, die richtigen Winkelschätzungen zu erkennen und zu isolieren. Deshalb konnte auf grundlegendere Messtechniken (beispielsweise eine exak-

te Markierung der Position des Sprechers) verzichtet werden. So wurde die Audioaufnahme mithilfe des Lokalisierungsprogramms und des inzwischen erstellten ersten regelbasierten Trackingprogramms (vgl. 6.1) analysiert, und durch gemeinsame Betrachtung der jeweiligen Programmergebnisse in Verbindung mit dem Videobild für jede Sprachsequenz der Winkel des Sprechers zu den Mikrofonen ermittelt.

Aus der Aufnahme vom 1.3.2005 konnten insgesamt 38 Tracks annotiert werden, aus der Aufnahme vom 3.3.2005 insgesamt 60, und aus der Aufnahme vom 11.5.2005 115. Die einzelnen Phrasen hatten dabei eine Länge von einer knappen Sekunde bis zu mehreren Sekunden.

4.3 Verwendung der Daten

Die manuell annotierten Tracks der Aufnahme vom 3.3.2005 wurden im folgenden als Trainingsdaten verwendet, um den regelbasierten Tracker zu optimieren und den genetischen Algorithmus und das neuronale Netz zu trainieren. Die Tracks der Aufnahme vom 1.3.2005 wurden als Testdaten verwendet. Mit der Aufnahme vom 11.5.2005 wurden schließlich alle Verfahren evaluiert.

Kapitel 5

Lokalisierung

Für die Lokalisierung konnte auf ein innerhalb des SFB 588 entwickeltes Programm zurückgegriffen werden. Dirk Bechler vom Institut für Nachrichtentechnik an der Fakultät für Elektrotechnik hat in Matlab ein Programm entwickelt[BK05], das aus einem Stereosignal mithilfe der generalisierten Kreuzkorrelation (GCC) die Laufzeitdifferenz ermittelt und daraus den Winkel berechnet. Auch in anderen Arbeiten wird dieses Verfahren zur Lokalisierung verwendet (vgl. [VMRL03]); das Programm entspricht damit dem aktuellen Stand in der Forschung.

Das Programm liest in einer Offline-Version die von den Mikrofonen M_1 und M_2 aufgenommenen Audiodaten $x_1(t)$ und $x_2(t)$ für das Signal $s(t)$ aus einer angegebenen Audio-Datei, in der spätere Online-Version werden die Daten direkt von der Soundkarte gelesen. Das Programm betrachtet anschließend jeweils ein 16-ms-Fenster. Liegt der Energiepegel innerhalb des Fensters über einem vorgegebenen Schwellenwert, wird mithilfe des GCC-Algorithmus (vgl. 5.1) die relative Laufzeitdifferenz τ_i berechnen. Ebenfalls ausgegeben wird ein Konfidenzwert. Außerdem wird versucht, über einen einfachen mittelwert-basierenden Tracking-Algorithmus einen Track zu ermitteln. Bei Erfolg wird der Winkel des Tracks ausgegeben.

$$R_{12}^{(\mathrm{g})}(\tau) = \int_{-\infty}^{+\infty} \psi_{12}(\omega) X_1(\omega) X_2(\omega)^* e^{j\omega\tau} d\omega \tag{5.1}$$

Durch die Gewichtungsfunktion $\psi_{12}(\omega)$ wird der Einfluss von Rauschen und Hall verringert.

5.1 Erste Analyse des verwendeten Programms

Für eine erste Analyse des Programms wurden zunächst Testdaten aufgenommen. Dabei wurde ein ähnlicher Aufbau wie bei der späteren Evaluation verwendet. Die aufgenommene .wav-Datei wurde anschließend von einer Offline-Variante des Lokalisierungsprogramms bearbeitet, das daraus die Winkel berechnet und versucht, die Quelle zu tracken.

Die ersten Aufnahmen wurden im Poolraum des Instituts in Karlsruhe gemacht. Zunächst wurde eine Aufnahme mit einer Vielzahl von parallelen Sprechern gemacht. Der eingebaute Tracker kann bei dieser Aufnahme keine Tracks erkennen, betrachtet man jedoch die vom Lokalisierer ermittelten und in einem Winkel-Zeit-Diagramm dargestellten Ergebnisse (vgl. Abb. 5.1), so kann man durchaus einzelne Tracks erkennen (siehe Pfeile).

In einem zweiten Test hat ein Sprecher in einem Szenario ohne Nebengeräusche von festgelegten Positionen (0° , 45° , 90° etc.) kürzere Kommandos in Richtung des Roboters gesprochen. Es zeigte sich, dass das Lokalisierungsprogramm den Winkel korrekt ermittelt und der Tracker ebenfalls korrekt arbeitet.

Als Nachteil des Programms erschien zunächst, dass nur die jeweils lauteste Quelle berücksichtigt wird und alle anderen Signale ignoriert werden.

5.2 Andere Ansätze

Um den erwähnten Nachteil abzustellen, wurde eine neue Version des Lokalisierungsprogramms getestet, die die jeweils vier stärksten Winkel ausgab. Nach einigen Tests zeigte sich jedoch, dass die jetzt vierfache Zahl an Eingangsdaten keine Verbesserung bringt, sondern im Gegenteil zu einem Absinken der Erkennungsgenauigkeit von 88% auf 14% führt, da sich auch die Nebengeräusche vervierfachten und eine eindeutige Identifikation der Geräuschquellen damit unmöglich wird, da diese von den Nebengeräuschen und Fehlanalysen des Lokalisierungsprogramms übertönt werden.

Außerdem wurde versucht, durch eine eigene Implementation des Lokalisierungsalgorithmus' ein besser für die Aufgabenstellung optimiertes Programm zu erhalten. Dies sollte statt der GCC-Funktion ein „Delay and sum"-Beamforming-Algorithmus (vgl. [VMHR04]) zur Lokalisierung verwenden:

$$y(n) = \sum_{m=0}^{M-1} x_m(n - \tau_m) \tag{5.2}$$

Wenngleich dieser Ansatz unter idealen Bedingungen (also keine Nebengeräusche und wenig Hall) die gleichen Werte wie das zu Anfang genannte

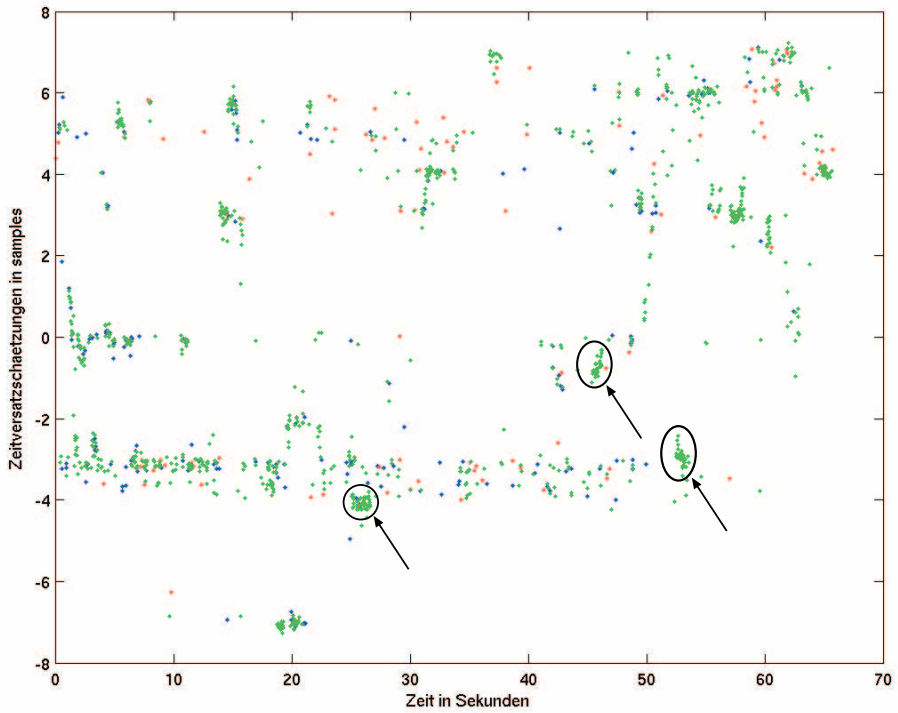

Abbildung 5.1: Winkel-Zeit-Diagramm der ersten Aufnahme

Referenzprogramm liefert und damit dessen korrekte Funktionsweise belegt, so kann es unter erschwerten Bedingungen aufgrund der mangelnden Komplexität keine verwertbaren Ergebnisse erzielen.

5.3 Einsatz des Programms

Für die weiteren Untersuchungen wurden die aufgenommenen Audiodaten vom Lokalisierungsprogramm bearbeitet und die Ergebnisdatei mit den Winkel-Zeit-Konfidenz-Tripletts (im folgenden als Audio-Spots bezeichnet) vom Tracker bearbeitet.

Kapitel 6

Tracking

Zunächst wurde für das Tracking ein per „trial-and-error“ ermittelter regelbasierter Algorithmus verwendet. Anschließend wurde getestet, inwieweit durch ein neuronales Netz Verbesserungen erzielt werden können. Da dies zunächst nicht erfolgreich war, wurde versucht, den regelbasierten Algorithmus durch einen genetischen Algorithmus zu optimieren. Anschließend wurde erneut getestet, ob basierend auf dem jetzt optimierten Algorithmus Verbesserungen möglich waren.

6.1 Regelbasierter Algorithmus

Der regelbasierte Tracker geht von der Annahme aus, dass die Sprecher-Signale teilweise von Störgeräuschen überlagert sein können, ihre Position im Verhältnis zur Abtastrate (62,5 Fenster/Sekunde) nur geringfügig verändern. Daher wird nach räumlich und zeitlich zusammenliegenden Signalen gesucht und diese zu einem Track zusammengefügt.

Beim Erstellen des Trackers wurden die in Karlsruhe gemachten Aufnahmen analysiert und der Tracker immer besser an diese Aufnahmen angepasst.

Abhängig von der Anzahl der Signale und ihrem zeitlichen Abstand sowie des Konfidenzwertes $R_{12}^{(g)}(\tau)$ (vgl. Gleichung 5.1, hier Spot_Score genannt) der Signale wird ein sogenannter Score-Wert für jeden Track berechnet:

$$\text{Track\_Score}_{\text{n}} = \text{Spot\_Score}^2 + \frac{\text{Track\_Score}_{\text{n}-1} * \alpha}{(\text{Spot\_Zeit} - \text{letzter\_Zeitstempel})^2} \qquad (6.1)$$

Um ein sich möglicherweise bewegendes Signal zu verfolgen, wird für jeden Track eine Winkelhypothese berechnet und mit jedem neuen Signal wie folgt

aktualisiert:

$$\text{Track\_Winkel}_{\text{n}} = \frac{Round((\text{Spot\_Winkel} * \beta + \text{Track\_Winkel}_{\text{n}-1} * (1 - \beta)) * \gamma)}{\gamma}. \tag{6.2}$$

Durch die manuell optimierten Parameter α bis γ wird der Algorithmus besser an das Aufnahmeszenario angepasst.

Geht innerhalb einer Timeout-Zeit von 1 s kein passendes Signal mehr ein, wird der Track gelöscht. Auf diese Weise entsteht eine Track-Tabelle, die alle zur Zeit aktuellen Track-Hypothesen verwaltet.

Für jeden neuen Audio-Spot wird geprüft, ob dieser zu einem bestehenden Track passen könnte. Falls nicht, wird ein neuer Track angelegt. Für diese Prüfung wird der Abstand des Winkels von der Trackhypothese mit einem festen Schwellenwert verglichen: Ist der Abstand ≤ 5, wird der Spot hinzugefügt, ansonsten nicht.

Bei den ersten Tests zeigte sich schnell, dass für den Timeout eine Berücksichtigung des aktuellen Score-Wertes nicht sinnvoll ist. Eine zusätzliche Variable, die die Bewegung erfassen soll, brachte ebenfalls keine Verbesserung.

In der Abbildung 6.1 sind die Ausgaben des regelbasierten Trackers am Beispiel eines Ausschnitts der Aufnahme vom 3.3.2005 dargestellt. Bereinigt um zu kurze und schwache Tracks sind alle Trackhypothesen angegeben. Mit einem waagerechten roten Strich sind im Vergleich zur manuellen Annotation falsche Hypothesen gekennzeichnet, korrekt erkannte Tracks sind dagegen mit einem grünen Strich unterlegt.

6.2 Genetischer Algorithmus

Da der regelbasierte Tracker zahlreiche Variablen verwendet, die seine Trefferquote bestimmen, erschien es zielführend, diese Variablen durch einen genetischen Algorithmus zu optimieren. Die Fitness-Funktion wurde entsprechend der Aufgabenstellung festgelegt: Der Tracker muss am Ende der Sprachsequenz den Winkel des Sprechers derart angeben, dass der Roboter seinen Kopf in die entsprechende Richtung drehen kann. Da eine exakte Bestimmung der Richtung unmöglich und für die Kopfdrehung auch nicht notwendig ist, wurde für den Winkel ein Zielbereich von $\pm\ 5°$ festgelegt. Letztendlich wurde gezählt, wie viele Tracks entsprechend dieser Festlegung korrekt erkannt wurden.

Ein Individuum wurde durch die Parameter und die damit ermittelte Trefferquote gekennzeichnet. Zu Beginn wurde mit einer Population von 100 Individuen gearbeitet. Es zeigte sich jedoch, dass bei dieser Konfiguration die Ergebnisse stark variieren, da häufig nur ein lokales Minimum, jedoch

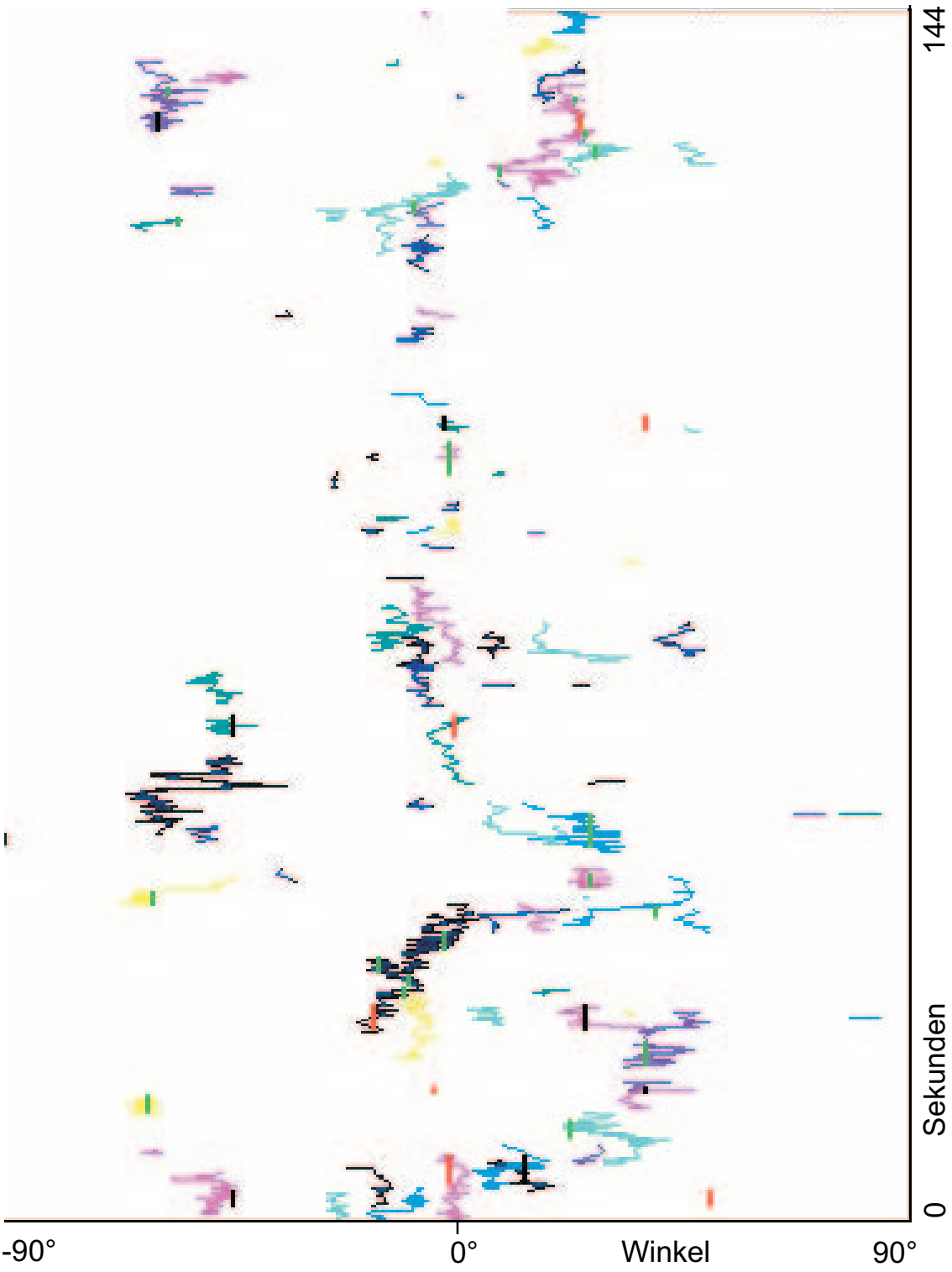

Abbildung 6.1: Grafische Ausgabe der Tracks des regelbasierten Trackers

nicht der beste erreichbare Wert erzielt wird. Die Populationsgröße wurde deshalb auf 200 Individuen erhöht. In einem Durchgang wurden sämtliche Individuen getestet, und die besten 40 in die nächste Generation unverändert übernommen und aus diesen durch zufällige Mischung der Parameter 140 neue Individuen erzeugt. Aus den besten 40 Individuen wurden außerdem 20 Exemplare zufällig ausgesucht und um mit zufallsgesteuert um bis zu 20% variierten Parametern in die nächste Generation übernommen. Da mit dieser Konfiguration eine Fehlerquote von 12 % und damit sehr gute Ergebnisse erzielt werden können, wurden weitere Konfigurationen nicht getestet.

Als Trainingsdaten wurden dabei die Aufnahmen vom 3.3.2005 verwendet.

Im Laufe der Untersuchungen zeigte sich, dass einerseits mit sehr unterschiedlichen Parametern gleiche Ergebnisse erzielt werden konnten, andererseits aber schon geringe Änderungen der Parameter großen Einfluss hatten. Deshalb waren die Trainingsergebnisse teilweise auch zufällig bestimmt, und es hing von der zufallsbasierten Initialisierung der Parameter ab, ob tatsächlich das globale Maximum im Bezug auf die Ergebnisse erreicht werden konnte.

6.3 Neuronales Netz

Für den regelbasierten Tracker ist ein Kriterium besonders entscheidend: die Entscheidung, welche Punkte zu welchem Spot hinzugefügt werden. Bereits bei einigen wenigen Punkten, die zu einem anderen Track hinzugefügt werden, verändert sich die Winkelhypothese, die Lebensdauer und der Score-Wert aller anderen Tracks massiv, da diese Punkte neue Tracks begründen oder andere Tracks beeinflussen können. Deshalb bietet es sich an, diese Entscheidung nicht nur von einem regelbasierten Tracker zu treffen, sondern durch ein neuronales Netz einen komplexeren Ansatz zu wählen, der zusätzliche Parameter berücksichtigen kann und dabei optimal trainiert werden kann.

Folgende Parameter wurden dabei gewählt:

- (a) Die Differenz zwischen der Winkelhypothese für den Track und der Winkel des Spots
- (b) Der durch das Lokalisierungsprogramm berechnete Winkel des Spots
- (c) Der „Root-Mean-Square“-Wert aus der jeweiligen Differenz zwischen Winkelhypothese und der hinzugefügten Punkte
- (d) Die Anzahl der Spots im Track

- (e) Der durch das Lokalisierungsprogramm berechneten Konfidenz-Wert
- (f) Der durch Funktion 6.1 berechneten Score-Wert für den Track
- (g) Die Rate der Spots pro Sekunde
- (h) Die Differenz zwischen dem Zeitstempel des letzten hinzugefügten Spots und der aktuellen Zeit

Als Ausgabe soll das Netz eine 1 ausgeben, wenn der Punkt zu dem Track hinzugefügt werden soll, und 0, wenn nicht.

6.3.1 Berechnung der Trainingsdaten

Um das neuronale Netz ausreichend trainieren zu können, mussten zunächst ausreichend Trainingsdaten gewonnen werden. Der bisherige regelbasierte Tracker konnte dafür nicht verwendet werden, da das Netz sonst lediglich dessen Regeln lernen würde. Um Daten zu erhalten, die ein darüber hinausgehendes Training ermöglichten, wurde ein neuer Tracker konzipiert, der basierend auf bereits getaggten Daten einen optimalen Pfad vom Startwinkel und der Startzeit zum Endwinkel und der Endzeit berechnen soll.

Da dieses Programm lediglich nachträglich bereits vorhandene Daten auswerten sollte und nicht für den Echtzeit-Einsatz gedacht war, konnten Laufzeit- und Komplexitätsprobleme außer Acht gelassen werden. Deshalb wurde ein rekursiver Ansatz gewählt, bei dem alle möglichen Pfade ermittelt und deren Güte bestimmt wurde. Die Güte wurde dabei folgendermaßen berechnet:

$$\text{Track\_score} = \sum_{k=2}^{n} (\text{Winkel}_k - \text{Winkel}_{k-1})^2 + (\text{Zeit}_k - \text{Zeit}_{k-1})^2. \qquad (6.3)$$

Je niedriger der Wert Track_score war, desto weniger Winkelabweichungen gab es auf dem Pfad, gleichzeitig sind aber auch so viele Punkte wie möglich auf dem Pfad, damit eine stabile Verfolgung möglich ist.

Um die Laufzeit kontrollierbar zu machen, wurden einige Einschränkungen eingebaut: So wurde die Zahl der Iterationen pro Track auf 1 Millionen festgelegt, und nach einem gefundenen lokalen Minimum offensichtlich schlechtere Äste im Iterationsbaum abgeschnitten.

Testweise wurde diese Algorithmus zunächst rückwärts laufen gelassen, er sollte also einen Pfad vom Endpunkt bis zum Startpunkt suchen. In einem zweiten Durchgang wurde der Algorithmus vorwärts gestartet. Dabei zeigte sich, dass die Erkennung in diesem Fall etwas besser ist.

Auf diese Weise konnte nach einer Durchlaufzeit von 30 Minuten bei den als Trainingsdaten verwendeten Aufnahme vom 3.3.2005 45 Tracks erkannt

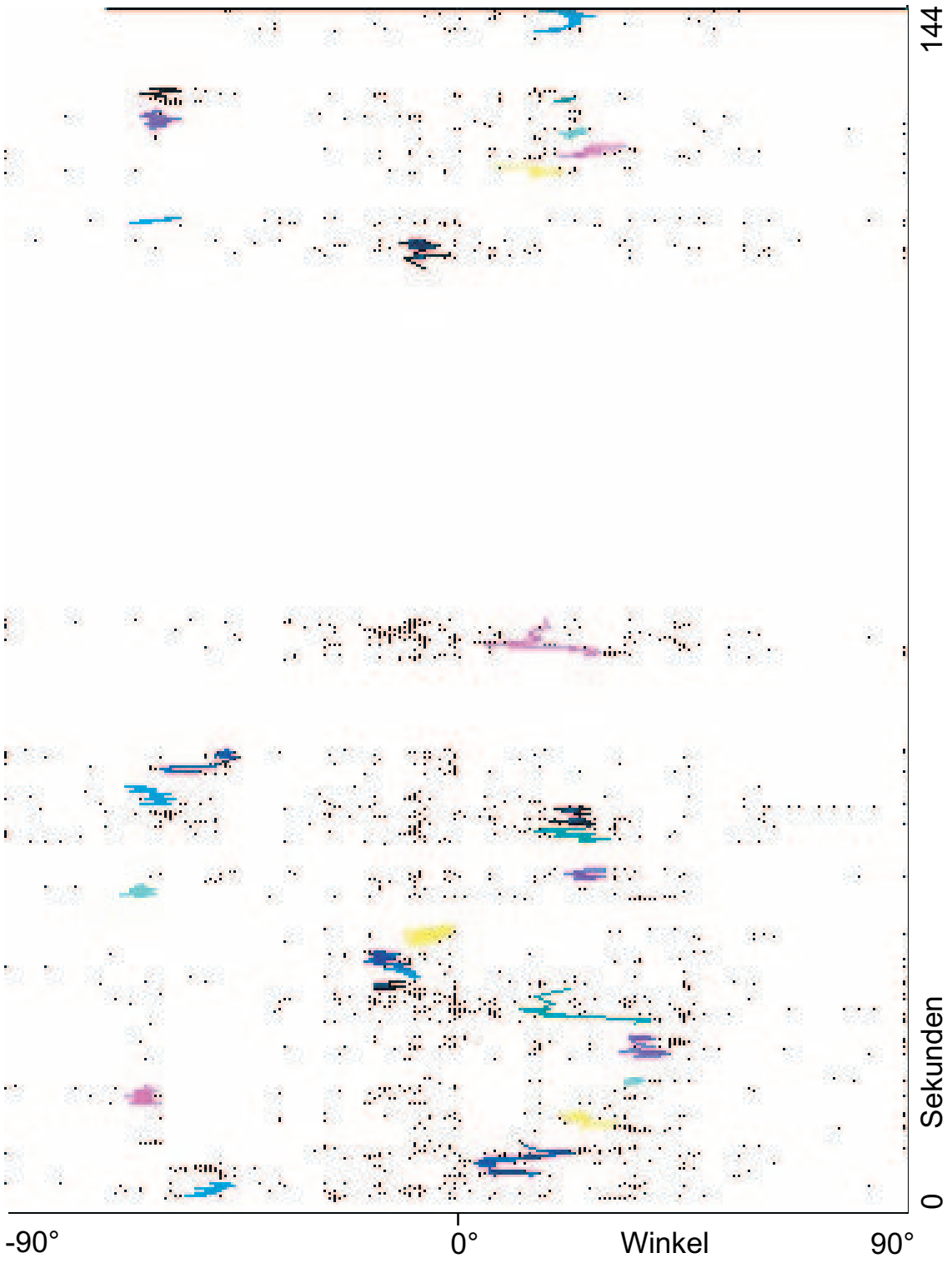

Abbildung 6.2: Grafische Ausgabe der Tracks des rekursiven Trackers

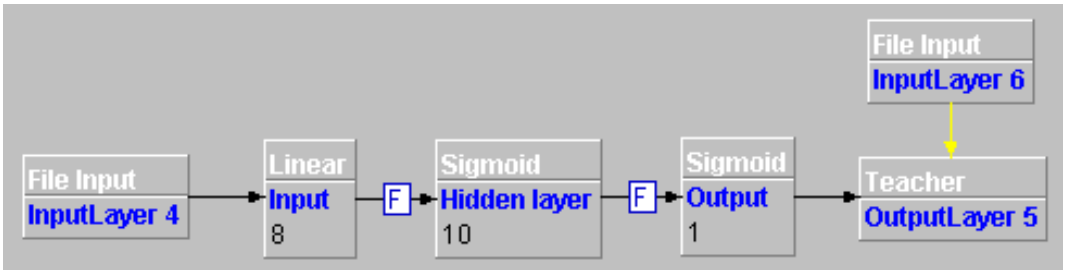

Abbildung 6.3: Aufbau des neuronalen Netzes

werden - 15 konnten allerdings nicht erkannt werden. Anschließend wurden für die einzelnen Punkte die 9 Parameter berechnet und um die Information ergänzt, ob der Punkt zum Track hinzugefügt wurde oder nicht. Auf diese Weise ergaben sich 4642 Trainingsdatensätze.

In Abbildung 6.2 ist dargestellt, welche Punkte zu einem Track hinzugefügt wurden und welche nicht.

6.3.2 Aufbau des neuronalen Netzes

Das neuronale Netz wurde in Joone[Joo] erstellt. Joone erlaubt ein direktes Einbinden des Netzes durch Java-Bibliotheken und kann prinzipiell auf allen von Java unterstützen Rechnerarchitekturen laufen; bei den für die Studienarbeit vorgenommenen Experimenten zeigt sich jedoch, dass es nicht mehrprozessor-fähig ist. Das neuronale Netz besteht aus einer linearen Eingabeschicht mit 9 Neuronen, einer versteckten Schicht und einer Ausgabeschicht mit einem Neuron (vgl. Abbildung 6.3). Versteckte und Ausgabeschicht bestehen aus Sigmoid-Neuronen.

Für das Training wurde eine Lernrate von 0,6, ein Momentum von 0,4 und 2000 Epochen eingestellt. Anschließend wurde das Netz in serieller Form exportiert und mit den Joone-eigenen Funktionen in den Tracker geladen und verwendet.

6.4 Stand-Alone-Version

Da zum Zeitpunkt der Anfertigung der Studienarbeit noch kein mit far-distance-Mikrofone einsetzbarer Sprachsegmentierer existierte, wurde zu Testzwecken eine Stand-Alone-Version des Trackers entwickelt. Dabei wurde für die Signaldetektion der errechnete Track-Score verwendet: Sobald dieser Wert über einem bestimmten Schwellenwert lag, wurde der Winkel des entsprechenden Tracks ausgegeben.

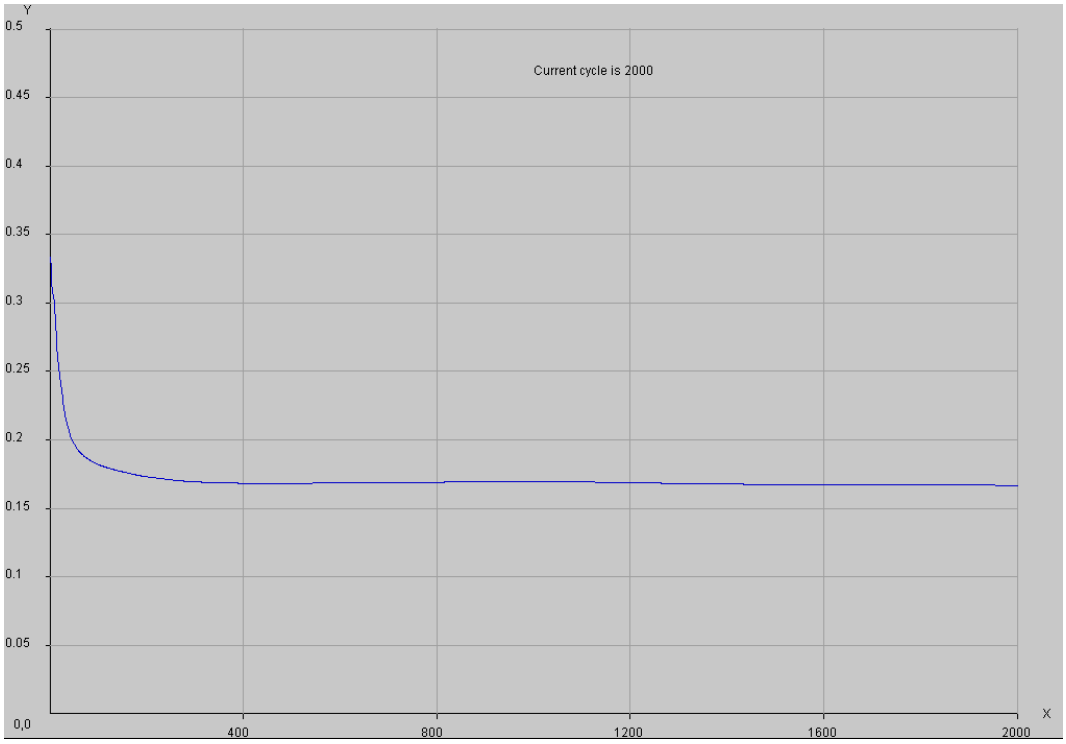

Abbildung 6.4: Verlauf der Fehlerrate des neuronalen Netzes während des Trainings

Mit einem Testaufbau, bei dem vom Lokalisierer der Roboterkopf in die ermittelte Richtung gedreht wurde, konnte die Erkennungsqualität anschaulich demonstriert werden. Dabei musste jedoch darauf geachtet werden, während der Motorlaufzeit alle neuen Signale auszublenden und bei allen Track-Hypothesen den veränderten Winkel zu berücksichtigen.

6.5 Eingabefusion

Bei der Eingabefusion der Tracking-Ergebnisse muss für einen vorgegebenen Zeitabschnitt, den der Sprachsegmentierer ermittelt, der am besten passende Track gefunden werden. Gesucht ist also ein Track, dessen erstes Signal mindestens vor der Startzeit des Segmentierers und dessen letzten Signal frühestens nach der Stoppzeit des Segmentierers gespeichert ist. Um mögliche Fehler bei der Segmentierung oder Lokalisierung auszugleichen, wurde ein Toleranzbereich für die Abweichung der Zeitstempel festgelegt - das erste Signal darf also auch um einen gewissen Betrag der Startzeit eingetroffen sein bzw. das letzte Signal vor der Stoppzeit.

Existieren mehrere passende Tracks, wird ein Score-Wert für die zeitliche Übereinstimmung gebildet und der höchste Score-Wert der einzelnen Tracks dabei mitbewertet. Ein Signal, das nur kurz vor der Segmentierer-Startzeit beginnt, wird dabei einem Signal vorgezogen, das bereits schon längere Zeit aktiv ist, selbst wenn dieses möglicherweise einen höheren Track-Score hat. Als Ausgabewert wird schließlich der letzte registrierte Winkel des gewählten Tracks ausgegeben.

$$\delta_{\text{Start}} = \text{Track\_Start} - \text{Segmentierer\_Stopp} \tag{6.4}$$

$$\delta_{\text{Stopp}} = \text{Track\_Stopp} - \text{Segmentierer\_Stopp} \tag{6.5}$$

$$\text{Match\_Score} = \frac{\text{Track\_maxScore}}{\sqrt{\eta * {\delta_{\text{Start}}}^2 + \theta * {\delta_{\text{Stopp}}}^2}}. \tag{6.6}$$

Die Parameter η und θ wurden über durch den genetischen Algorithmus optimiert.

Zum Testen der Fusion in Echtzeit wurde zunächst eine Anwendung erstellt, bei der der Benutzer durch Klicken mit der Maus eine manuelle Segmentation durchführen konnte - solange das Mausbutton gedrückt war, wurde ein Sprachsignal angenommen, sobald er ihn losgelassen hatte, wurde versucht, den Winkel zu ermitteln, indem die aufgezeichneten Start- und Stoppzeiten an die Fusionskomponente weitergegeben wurden. Dabei zeigte sich, dass eine Wartezeit von mindestens 300 ms nach Ende des Sprachsegments notwendig ist, damit der Lokalisierer das Ende des Tracks registrieren kann.

Für den Einsatz im Roboter wurde die Fusion in den One4All-Spracherkenner integriert. Dort wird sie innerhalb des Segmentierungsmoduls aufgerufen, sobald eine Sprachsequenz erkannt worden ist. So ist die notwendige Wartezeit garantiert, und der erhaltene Winkel kann zur weiteren Verarbeitung durch den Tapas-Dialogmanager den anderen Erkennungsinformationen hinzugefügt werden. Auf diese Weise kann eine Kopfdrehung nur bei bestimmten Kommandos oder in bestimmten Szenarien initiiert werden.

Kapitel 7

Ergebnisse

Gemäß der Aufgabenstellung ist das Ziel dieser Studienarbeit, nach einem gesprochenen Satz den Roboterkopf in die Richtung des Sprechers zu drehen. Bei den Tests der verschiedenen Tracker wurde daher ermittelt, inwieweit dieses Ziel erreicht werden konnte. Es wurde daher untersucht, ob für den gegebenen Zeitabschnitt überhaupt ein zusammenhängender Track gefunden werden konnte und ob der Winkel des von der Fusionskomponente ausgewählten Tracks innerhalb eines Bereiches von $\pm$ 5° entsprechend der manuellen Notation lag.

Für das Training, den Test und die Evaluierung wurden Daten in verschiedenen Szenarien aufgenommen (vgl. 4.1). Insbesondere das Evaluierungsszenario unterscheidet sich stark von den anderen beiden: Diese Aufnahme wurde in einem Bürozimmer gemacht, während die anderen beiden Aufnahmen in einem Poolraum gemacht wurden, der sich im Bezug auf Hall und Nebengeräusche von einem Bürozimmer unterscheidet.

Zu Beginn der Studienarbeit wurde bereits ein erster Test durchgeführt, um die Erfolgsquote des vorhandenen, im Lokalisierer eingebauten mittelwertbasierenden Trackers zu ermitteln. Da dieser Tracker keine einzelnen Tracks und somit keine Zeitabschnitte verwaltet, wurde untersucht, ob der letzte ermittelte Winkel des Trackers in einem zeitliche Bereich von bis zu 5 Sekunden vor dem Stopp-Zeitstempel innerhalb des Zielbereichs lag. Es zeigte sich, dass bei einer Aufnahme mit vielen Nebengeräuschen (der Aufnahme vom 1.3.2005) nur 29% der Tracks korrekt erkannt wurden.

Der zu Beginn erstellte regelbasierte Tracker erkannte dagegen bei derselben Aufnahme bereits 49% der Tracks korrekt.

Der Einbau eines neuronalen Netzes in diesen Tracker brachte keine Verbesserung:

Durch die Verwendung des genetischen Algorithmus zur Optimierung des regelbasierten Trackers ergab sich ebenso auf den Testdaten keine Verbesse-

Anzahl der Neuronen	2	4	5	6	10	20
Erkennungsquote	32%	37%	45%	39%	34%	29%

Tabelle 7.1: Ergebnisse der Integration des neuronalen Netzes im Bezug auf die Zahl der Neuronen in der versteckten Schicht

rung: Wie zuvor ergab sich eine Trefferquote von 50%. Eine größere Verbesserung ergab sich jedoch auf den Trainingsdaten: Hier konnten statt zuvor 50% jetzt 70% erzielt werden. Offenbar führte die stärkere Optimierung für ein Szenario dazu, dass andere Szenarien nicht besser erkannt wurden.

Deutlich besser fielen die Ergebnisse für die Kombination aus genetischer Optimierung und neuronalem Netz aus: Hier wurde auf den Testdaten eine Erkennungsrate von 80% erreicht, auf den Trainingsdaten allerdings nur 79% und damit weniger als beim Tracker ohne neuronales Netz, ebenso wie bei den Evaluationsdaten mit 54%.

	Mittelwert-basierender Tracker	Regelba-sierender Tracker	Regelba-sierender Tracker m. genetischer Optimie-rung	Genetisch optimierter Tracker m. neuronalem Netz
Trainingsszenario	30%	49%	88%	79%
Testszenario	32%	58%	74%	82%
Evaluationsszenario	49%	37%	62%	54%

Tabelle 7.2: Alle Ergebnisse im Überblick

7.1 Ergebnisse des Stand-Alone-Trackers

Zur Evaluation des Stand-Alone-Trackers wurde manuell gezählt, wie oft der Roboter auf ein gesprochenes Kommando mit einer Kopfdrehnung in Richtung des Sprechers reagierte. Dabei wurde eine Erkennungsquote von 69% erzielt.

7.2 Diskussion

Die vorliegenden Ergebnisse zeigen, dass die entwickelten Verfahren im für die Aufgabenstellung geforderten Szenario eine deutliche Verbesserung ge-

genüber dem mittelwertbasierenden Tracker darstellen. Mit Erkennungsquoten von 80% können sie ihren Zweck durchaus erfüllen, zumal der Benutzer bei einer falschen Erkennung durch eine Wiederholung des Kommandos eine erneute Erkennung auslösen kann.

Bei genauerer Betrachtung zeigt sich auch, dass die genetische Optimierung ebenfalls eine Verbesserung gegenüber dem zu Beginn verwendeten Tracker darstellt.

Ingesamt zeigt sich also, dass durch das neuronale Netz nicht in jedem Fall eine Verbesserung der Erkennungsquote zu erreichen ist.

Kapitel 8

Ausblick

Während der Erstellung der Studienarbeit ergab sich das Problem, dass eine auf Far-Distance-Mikrofonen zuverlässig arbeitende Sprachsegmentierung noch nicht verfügbar war. Aus diesem Grund erfolgten sämtliche Messungen und Evaluierungen mit manuell segmentierten Daten. Da allerdings bereits zahlreiche Projekte zur Entwicklung einer Spracherkennung für Far-Distance-Mikrofone laufen [WNM05], könnte sich diese Situation bald ändern.

Eine funktionierende Schalllokalisierung kann theoretisch auch in anderen Szenarien eingesetzt werden, als in der Einleitung beschrieben: Neben der Lokalisierung von Sprache könnten auch Geräusche geortet werden, beispielsweise zur Verbesserung der Geräuscherkennung. In diesem Fall könnten eventuell auch parallele Geräusche besser segmentiert und erkannt werden, indem eine bandselektive Lokalisierung verwendet wird. Außerdem könnte durch ein Umweltwissen, das die Position der Geräuschquellen beinhaltet, Fehlerkennungen vermieden werden. Notwendig von Seiten des Lokalisierers wäre ein neues Training des genetischen Algorithmusses und des neuronalen Netzes, um das Programm an die veränderte Charakteristik anzupassen.

Da der Lokalisierer und Tracker bisher nur mit manuell evaluierten Daten getaggt worden sind, würde es interessant, mit einer automatischen Evaluation die Funktionsweise zu prüfen. Möglich wäre eine visuelle Evaluation mit Hilfe der Kameras, die auf dem Kopf auf gleicher Ebene wie die Mikrofone installiert sind. Verwendet werden könnte beispielsweise ein System zur Gesichtserkennung, wie es beispielsweise von Christoph Schaa verwendet wird [VJ01]. Dazu müßte jedoch zuerst die Zuverlässigkeit des Gesichtserkenners überprüft werden.

Literatur

[BBI+98] Brooks, R. A., C. Breazeal, R. Irie, C. C. Kemp, M. Marjancvic, B. Scassellati und M. M. Williamson: *Alternative essences of intelligence*. In: *Proceedings of 15th National Conference on Artificial Intelligence*, Seiten 961–968, 1998. 1

[BK05] Bechler, Dirk und Kristian Kroschel: *Reliability Criteria Evaluation for TDOA Estimates in a Variety of Real Environments*, 2005. 2, 13

[FGH+97] Finke, M., P. Geutner, H. Hild, T. Kemp, K. Ries und M. Westphal: *The karlsruhe-verbmobil speech recognition engine*. In: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 1997. 2

[GD99] Gordon, N.J. und A. Doucet: *Sequential Monte Carlo for maneuvring target tracking in clutter*. In: *Proceedings SPIE*, Seiten 493–500, 1999. 8

[Joo] *Joone*. http://www.joone.org/. 23

[KBG02] Kroschel, K., D. Bechler und M. Grimm: *Speaker Tracking with a Microphone Array Using Kalman Filtering*. In: *Kleinheubacher Tagung, Miltenberg, Germany*, 2002. 7

[NLGV05] Ng, William, Jack Li, Simon Godsill und Jaco Vermaak: *Multitarget Tracking Using a new Soft-Gating Approach and Sequential Monte Carlo Methods*. In: *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2005. 8

[SMFW01] Soltau, H., F. Metze, C. Fuegen und A. Waibel: *A one pass-decoder based on polymorphic linguistic context assignment.*

In: *Proceedings of the Automatic Speech Recognition and Understanding Workshop, ASRU-2001*, 2001. 2

[VB01] Vermaak, J. und A. Blake: *Nonlinear filtering for speaker tracking in noisy and reverberant environments.* In: *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2001. 8

[VJ01] Viola, Paul und Michael Jones: *Robust Real-time Object Detection.* In: *Second International Workshop on Statistical and Computational Theories of Vision Modeling, Learning, Computing, and Sampling*, 2001. 31

[VMHR04] Valin, Jean-Marc, François Michaud, Brahim Hadjou und Jean Rouat: *Localization of Simultaneous Moving Sound Sources for Mobile Robot Using a Frequency-Domain Steered Beamformer Approach.* In: *IEEE International Conference on Robotics and Automation*, 2004. 14

[VMRL03] Valin, Jean-Marc, François Michaud, Jean Rouat und Dominic L´etourneau: *Robust Sound Source Localization Using a Microphone Array on a Mobile Robot.* In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003. 7, 13

[Wik05a] Wikipedia: *Lokalisation (Akustik)*, Juni 2005. http://de.wikipedia.org/wiki/Lokalisation_%28Akustik%29. 5

[Wik05b] Wikipedia: *Schallgeschwindigkeit*, Juni 2005. http://de.wikipedia.org/wiki/Schallgeschwindigkeit. 4

[WNM05] Wölfel, Matthias, Kai Nickel und John McDonough: *Microphone Array Driven Speech Recognition: Influence of Localization on the Word Error Rate.* In: *2nd Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, 2005. 31