Fakultät 3 Professur Kommunikationstechnik

Masterarbeit

Robuste Parameterschätzung für DS-Beamforming in einem dreidimensionalen Mikrofonfeld

Martin Birth
Geboren am 4. Dezember 1985 in Berlin
Matrikelnummer: 2934203
Immatrikulationsjahr: 2009

Zur Erlangung des akademischen Grades

Master of Science (M.Sc.)

Prüfer
Prof. Dr.-Ing. habil. Matthias Wolff

Zweitprüfer
Dr.-Ing. Ronald Römer

Eingereicht am: 12. März 2016

Brandenburgische Technische Universität Cottbus - Senftenberg

Professur Kommunikationstechnik

Kurzfassung

Die Kopplung akustischer Signale von verschiedenen Mikrofonen zu einem einzelnen Ausgangssignal ist ein faszinierender Sachverhalt. Ein derartiger Zusammenschluss wird Mikrofonfeld genannt. Er bietet die Möglichkeit, ohne eine physikalische Einflussnahme den Fokuspunkt auf eine spezifische Raumposition zu lenken. Die Aussteuerung von Mikrofonfeldern ist als Beamforming (dt. Strahlformung) bekannt. Grundlegend erfolgt eine Fokussierung über eine Verzögerung (engl. delay) mit anschließender Addition (engl. sum) der einzelnen Mikrofonsignale. Allgemein ist von „Delay-And-Sum“ (DS) Beamforming die Rede. Das Ausgangssignal wird zur Verbesserung der Signalqualität, Unterdrückung von Störquellen oder Positionsbestimmung von Sprechern genutzt. Die Ermittlung von robusten Verzögerungszeiten spielt in allen genannten Verwendungszwecken eine entscheidende Rolle. Die vorliegende Arbeit „Robuste Parameterschätzung für DS-Beamforming in einem dreidimensionalen Mikrofonfeld“ beschäftigt sich konkret mit dem Thema der Sprecherlokalisierung. Die parametrischen Verzögerungsglieder für das Beamforming müssen aus der Ankunftsrichtung des Schalls geschätzt werden. Die Schätzung unterliegt verschiedenen äußeren Bedingungen, die sich negativ auf das Ergebnis auswirken. Um die einzelnen Fehler zu minimieren, bedarf es eines großen Mikrofonverbundes, das die Professur für Kommunikationstechnik mit einem dreidimensionalen Mikrofonfeld im „Cognitive Systems Lab“ besitzt. Es bietet optimale Voraussetzungen für die Ortung von Sprechern in einer Laborumgebung. Für die zielsichere Bestimmung der Sprecherposition wurde weiterhin eine geschlossene Lösung eingesetzt, die Fehler bezüglich der Schätzung für Ankunftsrichtung mit berücksichtigen kann. Das Ergebnis soll eine Adaption von bekannten Algorithmen für zweidimensionale Mikrofonfelder auf ein dreidimensionales Mikrofonfeld sein. Eine neu entwickelte grafische Benutzerschnittstelle bietet darüber hinaus zahlreiche Interaktionsmöglichkeiten zur Steuerung des Mikrofonfeldes. Eine Visualisierung veranschaulicht darin das Verhalten und die Ausprägung des eingesetzten DS-Beamformers.

I Inhaltsverzeichnis

II Abbildungsverzeichnis

III Tabellenverzeichnis

IV Programmcodeverzeichnis

V Abkürzungsverzeichnis

API	Application Programming Interface
AWT	Abstract Window Toolkit
ARGB	Alpha Rot Grün Blau
BGRA	Blau Grün Rot Alpha
CSL	Cognitive Systems Lab
DFT	Diskrete Fourier-Transformation
DOA	Direction Of Arrival
DSB	Delay And Sum Beamformer
ECS	Echo Cancellation System
ESPRIT	Estimation Of Signal Parameters Via Rotational Invariance Techniques
FIR	Finite Impulse Response
GCC	Generalized Cross-Correlation
CPU	Central Processing Unit
GPGPU	General Purpose GPU
GUI	Graphical User Interface
HRTF	Head Related Transfer Function
HT	Hannon und Thomson
IDE	Integrated Development Environment
IDFT	Inverse Diskrete Fourier-Transformation
ILD	Interaural Level Difference
IIR	Infinite Impulse Response
IPC	Industry Personal Computer
JDK	Java SE Development Kit
JPA	JavaPortAudio
JVM	Java Virtual Machine
LCARS	Library Computer Access and Retrieval System
LCARSWT	LCARS Widget Toolkit
LS	Least Square
MISO	Multiple Input Single Output
ML	Maximum Likelihood
MSC	Magnitude Squared Coherence
MUSIC	Multiple Signal Classification
MVDR	Minimum Variance Distortionless Response
PHAT	Phase Transform
SCOT	Smoothed Coherence Factor
SNR	Signal-To-Noise Ratio
SRP	Steered Response Power
SWT	Standard Widget Toolkit
TDOA	Time Difference Of Arrival
UML	Unified Modeling Language
VAD	Voice Activity Detection

1 Einleitung

Die räumliche Einordnung akustischer Signale ist für den Menschen äußerst komplex. Richtungs- und Entfernungsbestimmungen beruhen auf keinen präzisen Aussagen zu der befindlichen Schallquelle. Die Quellenortung mittels Reizwahrnehmung über den auditiven Kanal basiert auf der Head Related Transfer Function (HRTF) (dt. kopfbezogene Übertragungsfunktion) (vgl. [MOK12] S.4). Diesbezüglich erlernt der Mensch über die Zeit, wie er Richtungsinformationen durch seine körperspezifischen Eigenheiten einzuordnen hat. Darunter zählt die Separation einzelner Schallquellen aus einer Geräuschkulisse heraus („Cocktail party effect“) ([Aro92]). Die Eigenheit befähigt Personen, spezifische Stimmen aus einer Gruppe von Sprechern zu filtern.

Komplizierter gestaltet sich die Umsetzung solch einer menschlichen Eigenschaft in der Mensch-Maschine-Kommunikation. Jene Sprachkommunikationssysteme sind gekennzeichnet durch eine Abhängigkeit von kabel- oder funkgebundenen Mikrofonlösungen. Entsprechende technische Hilfsmittel befinden sich unmittelbar am Körper oder in Körpernähe und offerieren dem Probanden keine natürliche Kommunikation. An dieser Stelle finden Mikrofonfelder ihre Anwendung. Sie bieten eine Vielzahl von Vorteilen gegenüber alleinstehenden Mikrofonlösungen, wie sie bei klassischen Tisch- oder Standschallwandlern zum Einsatz kommen. Ungeachtet der festen elektrischen Verkabelung bieten Mikrofonfelder einen größeren Aktionsradius für den Sprecher. Die Ausrichtung auf einen spezifischen Fokuspunkt erfolgt rein rechnerisch und stützt sich auf keine physikalischen Veränderungen. Als Ergebnis erhält das System ein qualitativ hochwertiges Ausgangssignal, welches eine Dämpfung der nebenstehenden Störquellen beinhaltet. Des Weiteren haben Mikrofonfelder das Potential, einen oder mehrere bewegliche Sprecher parallel zu lokalisieren bzw. dynamisch zu verfolgen.

Das Verfahren, das zur Zusammenfassung der Mikrofonfeldsignale Verwendung findet, wird *Beamforming* (dt. Strahlformung) genannt. Die in Verbindung stehende Mikrofonfeldverarbeitung teilt sich nach [LW10] in drei Forschungsgebiete auf: (I) Erkennung von einem oder mehreren vorhandenen Eingangssignalen, (II) Bestimmung des Schalleinfallswinkels und (III) Verbesserung des Signals unter gleichzeitiger Dämpfung von nebenstehenden Störquellen. In allen Teilgebieten sind nicht nur zeitliche, sondern gleichermaßen räumliche Informationen von Bedeutung. Einsetzen lässt sich diese Technik in lauten, lärmbehafteten, halligen Umgebungen oder überall dort, wo das Eingangssignal gestört ist. In der vorliegenden Arbeit wird vorrangig auf (II) ausführlich eingegangen, weil die Ankunftsrichtung der Schallwellen primär zur Lokalisation beiträgt.

Die Professur Kommunikationstechnik besitzt einen derartigen Verbund von Mikrofonen in Form eines Mikrofonfeldes. Das Besondere an dem Feld ist seine dreidimensionale Anordnung. Zwei Mikrofonfelder stehen zu diesem Zweck annähernd orthogonal zueinander (Abb. 19 in Anhang A.1). Entworfen wurde das Mikrofonfeld von Dipl.-Ing. Thomas Fehér an der TU Dresden. Er lieferte die Entwurfspläne für die jeweiligen Teilmikrofonfelder, nach denen die Konstruktion an der Brandenburgischen Technischen Universität Cottbus-Senftenberg durchgeführt wurde.

Das erste Mikrofonfeld (Abb. 20, in Anhang A.1) befindet sich um einem 72" Multifunktionsbildschirm mit nachgerüstetem Touchscreen. Die Messmikrofone vom Typ *Beyerdynamic MM1* sind durch einen Metallrahmen befestigt, der unmittelbar um den Monitor montiert ist. Zur Verbesserung der Optik diente

eine frontseitige Umrahmung mit einer Plexiglasscheibe.
Das zweite Teilfeld (Abb. 21, in Anhang A.1) wurde spiralförmig in eine 20 mm dicke lackierte Holzplatte eingebracht. Zur Verbesserung der Mobilität erfolgte eine deckenseitige Befestigung auf einem Metallschlitten. Durch eine Schrittmotorsteuerung ist das Mikrofonfeld in der Raumposition veränderbar. Das „Flatterecho“ aufgrund der schallharten, glatten Konstruktion wird durch eine frontlastige Neigung des Teilfeldes ausgeglichen.

Grundsätzlich benötigt jede Audioanwendung, die sich eines Mikrofonfeldes bedient, exakte Verzögerungsglieder. Dadurch ist eine Nachführung des Beamformers bzw. eine entsprechende Anvisierung der Schallquelle möglich. Die Zeitverzögerung zwischen den verschiedenen Signaleingangszeiten an den einzelnen Mikrofonen wird durch eine Reihe von Operationen geschätzt. Die Robustheit dieser Schätzung für die nachfolgenden Signalverarbeitungen ist von äußerster Wichtigkeit. Eine exakte Verarbeitung hat ebenso Einfluss auf die Qualität des nachgeführten Beamformers.

Daraus ergab sich folgend die Aufgabenstellung seitens der Professur Kommunikationstechnik:

I Schwerpunkt ist die Erarbeitung einer robusten Lösung zur Verbesserung der Zielausrichtung des eingesetzten Delay And Sum Beamformers (DSBs).

II Dies schließt die akustische Lokalisation von Sprechern im Raum ein.

III Darüber hinaus soll eine grafische Benutzeroberfläche geschaffen werden, die eine Steuerung und Visualisierung der Parameter für das Beamforming ermöglicht.

Die Implementierung erfolgt in das rein Java-basierte Softwareprojekt Cognitive Systems Lab (CSL), das alle Leistungsmerkmale des gleichnamigen Labors beinhaltet. Der komplette Funktionsumfang beinhaltet Module zur Gerätesteuerung, Verarbeitung von auditiven Signalen sowie einer grafischen Benutzerschnittstelle. Verknüpft mit der Implementierung ist eine entsprechende Dokumentationspflicht des entstandenen Quellcodes. Das geschieht in Form einer schriftlichen Schnittstellendokumentation und entsprechender Kommentierung der programmierten Codes.

Ziel der vorliegenden Arbeit ist es, einen Algorithmus zur Quellenlokalisation zu entwickeln, der sich auf ein dreidimensionales Mikrofonfeld anwenden lässt. Bisher existierende Ansätze für zweidimensionale Mikrofonfeldanordnungen können dahingehend adaptiert werden, dass ein robuster Einsatz für die verwendete Mikrofonfeldgeometrie möglich ist.

Die Gliederung der folgenden Abhandlung teilt sich in fünf Kapitel auf. Der erste Abschnitt stellt alle wichtigen theoretischen Grundlagen vor, die zum weiteren Verständnis der darauffolgenden Kapitel beitragen. Der nächste Abschnitt behandelt eine ausführliche Analyse der eingesetzten Mikrofonfeldverarbeitung und stellt damit konkrete Lösungen zur deren Verbesserung vor. Kapitel drei umfasst die Zusammenhänge zur akustischen Laufzeitschätzung und die Lokalisation von Schallquellen. Die konkrete Umsetzung wird in einem fünften Kapitel genauer geschildert. Der letzte Abschnitt präsentiert eine Visualisierung der Mikrofonfeldsteuerung im CSL.

2 Theoretische Grundlagen

2.1 Mikrofonfelder

Der Begriff des Mikrofonfeldes beschreibt eine Anordnung von räumlich verteilten Mikrofonen. Es ist nach [Sar13] durch die Mikrofonanzahl M, Form der Struktur (linear, quadratisch, dreidimensional usw.) sowie durch die Gesamtabmessung gekennzeichnet. Ein derartig verteiltes System ermöglicht den parallelen, passiven Empfang von Schallwellen an verschiedenen Raumpositionen.

Die genaue Beschreibung des Mikrofonfeldes erfolgt über eine Sensitivitätsverteilung, auch „*Beampattern*“ (dt. Strahlverteilung) genannt (vgl. [McC01], [BW01]). Sie entspricht einer Systemantwort, die für eine spezifische Frequenz und Ankunftsrichtung des Schalleinfalls definiert ist. Abbildung 1 zeigt die Sensitivitätsverteilung nach [Bir14] für das konkrete dreidimensionale Mikrofonfeld.

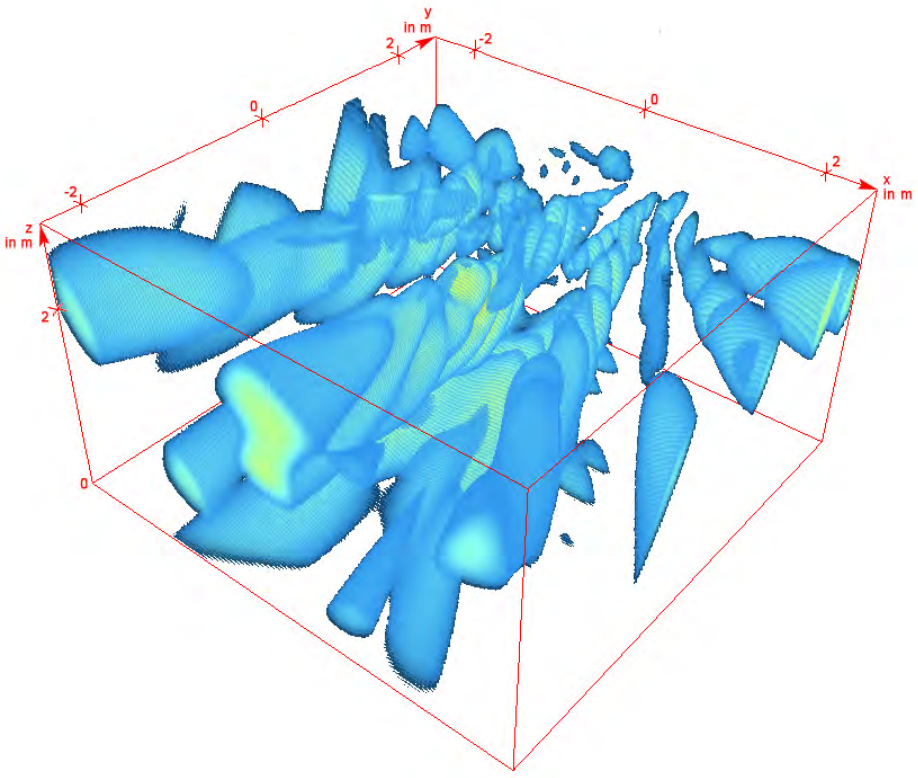


Abbildung 1: Sensitivitätsverteilung nach [Bir14] für das dreidimensionale Mikrofonfeld aus Abbildung 19 ($f = 1000\,\mathrm{Hz}$, $M = 64$ Mikrofone, Zielposition $q_s = (0; 0; 1,6)$, Mikrofonfeld 2 Position $p = 0\,\mathrm{m}$)

2.2 Räumliches Aliasing

Das Mikrofonfeld und die verknüpfte Sensitivitätsverteilung ist abhängig vom Abstand der Mikrofone. Wie in der zeitdiskreten Signalverarbeitung kommt es auch durch die räumliche Diskretisierung zu Alias-Effekten. Der Begriff des Aliasings beschreibt in diesem Zusammenhang eine mögliche Unterabtastung des Signals.

Um die Eigenschaften der Sensitivitätsverteilung weiter zu verdeutlichen, wird eine lineare Mikrofonfeldgeometrie angenommen. Ausgangspunkt ist die zeitdiskrete Abtastung, welche über die Nyquist-Frequenz definiert wird. Diese besagt, dass für eine verlustfreie Rekonstruktion eines Signales die Abtastrate f_a mindestens doppelt so hoch sein muss wie die höchste auftretende Frequenz f_{max} des Signalspektrums (vgl. [McC01]). Zusammengefasst wird das Theorem mit

$$f_a = \frac{1}{T_s} \geq 2 f_{max}, \tag{1}$$

wobei T_s einem Abtastintervall entspricht. Auf die räumliche Abtastung übertragen ergibt das

$$f_{x_s} = \frac{1}{d} \geq 2 f_{x_{max}}, \tag{2}$$

zusammen mit einer räumlichen Abtastrate f_{x_s}, dem Abstand d zwischen zwei Mikrofonen und der maximalen räumlichen Abtastfrequenz $f_{x_{max}}$. Die eindimensionale räumliche Abtastfrequenz f_{x_s} lässt sich entlang der x-Achse durch

$$f_{x_s} = \frac{(\sin\theta \cos\phi)}{\lambda} \tag{3}$$

definieren, mit dem Elevationswinkel θ, dem Azimutwinkel ϕ und der Wellenlänge λ. Naturgemäß besitzt diese Funktion kein Maximum. Ein Supremum existiert nicht. Nur für ein kleines λ und einen Zähler gegen Eins wird der Ausdruck maximal. Das lässt auf die Beziehung

$$f_{x_{max}} = \frac{1}{\lambda} \tag{4}$$

schließen. Damit ergibt sich für den Abstand d und der kleinsten auftretenden Wellenlänge λ_{min} das Verhältnis

$$d < \frac{\lambda_{min}}{2}. \tag{5}$$

Sollte demnach der Abstand der Mikrofone größer sein als die halbe Wellenlänge des Empfangssignals, kommt es zu räumlichen Aliasing.

Um übermäßige Nebenkeulen in der Sensitivitätsverteilung zu vermeiden, sollte der Abstand zwischen den Mikrofonen hypothetisch gegen Null laufen. Eine praktische Umsetzung dieser Forderung ist jedoch nicht möglich. Auf Grund dessen muss ein Kompromiss zwischen Frequenzauflösung und Mikrofonabstand gefunden werden. Dafür erfolgt eine Verschachtelung von verschiedenen äquidistanten, linearen oder spiralförmigen Teilfeldern (siehe Abb. 20/21, in Anhang A.1). Das sorgt für eine Verbesserung der Frequenzauflösung durch die Teilsysteme. Jede der Teilanordnungen bildet so ein entsprechendes Teilband des Frequenzspektrums ab. Die Grenzfrequenzen lassen sich mit (5) bestimmen (vgl. [ZGET00]).

2.3 Beamforming

Das Zusammenführen einzelner Mikrofonsignale eines Feldverbundes führt zu einer Raum-Zeit-Filterfunktion, mit der es möglich ist, nebenstehenden Lärm, Nachhallelemente oder Nebensprecher zu unterdrücken. Jenes Nebengeräuschreduktionsverfahren für Mikrofonfeldgeometrien wird als Beamforming (dt. Strahlrichtung) bezeichnet.
Der Einsatz richtet sich an die Signalaufbereitung, Schätzung der Einfallswinkel von Schallwellen oder an die Verbesserung von verrauschten und halligen Signalen. Weiterführend kommt das Beamforming in vielen anderen technischen Gebieten zum Tragen. Es wird in Bereichen der Astronomie, Seismologie, Radar, Sonar oder kabellosen Kommunikation eingesetzt.

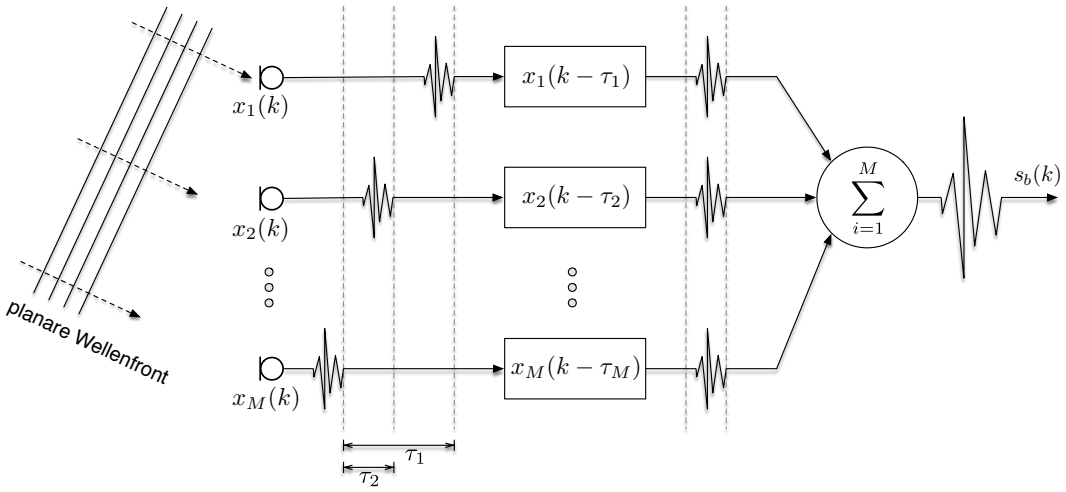


Abbildung 2: Schematische Darstellung eines einfachen Delay-And-Sum-Beamformers

Für ein Mikrofonfeld mit M Mikrofonen wird das diskrete Ausgangssignal nach [BAS95a] für den einfachsten Fall mit

$$s_b(k) = \sum_{i=1}^{M} x_s(k - \tau_i) + \eta_i(k) \tag{6}$$

bestimmt. Hierbei entspricht $s_b(k)$ einer durch Verzögerungen, Nachhall und Rauschen veränderten Version der originalen Schallquelle $s(k)$. Die diskreten Mikrofonsignale $x_i(k)$ mit $i = 1, \ldots, M$ werden mit räumlich unkorrelierten Rauschen $\eta_i(k)$ additiv überlagert. Ziel ist eine annähernde Phasenäquivalenz durch die zeitlichen Verschiebung τ_i. Daraus folgt eine Zurückhaltung des Signals beim ersten Mikrofon, bis das am Ende stehende Mikrofon in der räumlichen Abfolge erreicht ist. Anschließend findet eine Zusammenfassung der verzögerten Signale zu einem Ausgangssignal statt. Jener Algorithmus ist in der Literatur unter dem Begriff Delay-And-Sum-Beamformer (kurz DSB) bekannt - dargestellt in Abbildung 2. Eine derartige Vorgehensweise bezeichnet in der Literatur ebenso einen *Fixed Beamformer* (vgl. [McC01]) oder schmalbandigen *Narrowband Beamformer* (vgl. [LW10], S. 4).

Die Lenkung (engl. steering) des Fokuspunktes vom Mikrofonfeld wird in (6) durch Verzögerung (engl. delay) τ_i gesteuert. Die entsprechende Lenkverzögerung dient folglich der passiven Ausrichtung auf einen konkreten Raumpunkt. Es bedarf keiner physikalischen Einflussnahme.

Für die praktische Umsetzung wird ein Referenzmikrofon aus dem Feldverbund heraus festgelegt (vgl. [BAS95a], [Dre99] und [CHB05]). Ausgehend von dem einen Mikrofon werden $L = M - 1$ Mikrofonpaare gebildet. Anstelle von absoluten Verzögerungen finden die relativen Lenkverzögerungen Anwendung. Jene beziehen sich auf das Referenzmikrofon. Die Verwendung eines einzelnen Mikrofons als Referenz bedeutet, dass die direkte Bestimmung der Lenkverzögerung aus der Ankunftsrichtung des Schalls heraus erfolgen kann. Ferner wäre eine genaue Bestimmung der Position der Zielschallquelle nicht notwendig. Für die Evaluation ist es essentiell, die genauen kartesischen Koordinaten zu kennen.

Es muss gewährleistet sein, dass der Fokuspunkt eines DSBs der gesuchten Zielschallquelle entspricht. Eine korrekte und robuste Bestimmung ist demnach unerlässlich. Die nicht triviale Einflussnahme von akustischen Nebeneffekten erschwert den Prozess erheblich.

2.4 Bestimmung der Laufzeitdifferenz

Begründet in der räumlichen Ausdehnung des Mikrofonfeldes erreichen die akustischen Signale mit einer Zeitverzögerung die Mikrofone (Abb. 2). Darauf Bezug nehmend setzt das folgende Kapitel die Grundlage für die Bestimmung der Latenzperiode.

Ein zeitdiskretes Signal $s(k)$ mit $k = 0, \ldots, K-1$ wird für zwei im System befindliche und räumlich getrennte Mikrofonsensoren beschrieben durch:

$$x_1(k) = h_1(k) * s(k) + \eta_1(k), \tag{7}$$

$$x_2(k) = h_2(k) * s(k - \tau) + \eta_2(k). \tag{8}$$

Die einzelnen Kanäle sind mit den jeweiligen Raumimpulsantworten $h_1(k)$ bzw. $h_2(k)$ linear über den $*$-Operator gefaltet. Die kanalspezifischen Signale sind zusätzlich mit unkorreliertem Umgebungsrauschen $\eta_1(k)$ bzw. $\eta_2(k)$ additiv überlagert. $x_2(k)$ besitzt eine messbare, zeitliche Verschiebung um τ, welche durch die räumliche Trennung der beiden Mikrofone begründet ist. Dieser zeitliche Unterschied gibt dem Verfahren Time Difference Of Arrival (TDOA) seinen Namen. In der Literatur findet sich häufig die Bezeichnung Interaural Level Difference (ILD) (vgl. [Aar03], [HR09], [Sen02]). Es beschreibt eine Pegeldifferenz in Bezug auf das Richtungshören. Die Bestimmung von τ ist bei der TDOA eine der wichtigsten Überlegungen und kann nur geschätzt werden. Die Schätzung der Verschiebung wird über den Erwartungswert E der Kreuzkorrelationsfunktion der beiden Eingangssignale realisiert:

$$R_{x_1 x_2}(\tau) = E\left[x_1(k) x_2(k - \tau)\right]. \tag{9}$$

Eine Maximierung des Erwartungswertes erfolgt über τ.

Aufgrund der zeitlichen Beschränkung der Beobachtungszeit findet eine Schätzung über

$$\hat{R}_{x_1x_2}(\tau) = \frac{1}{K}\sum_{k=0}^{K-1} x_1(k)x_2(k-\tau) \tag{10}$$

statt, also mit K diskreten Abtastwerten. Die Zirkumflexnotation $\hat{\cdot}$ findet in der vorliegenden Arbeit für Schätzwerte Verwendung. Das Argument des maximalen Funktionswertes vom Korrelationsintegral liefert als Ergebnis die zeitliche Verschiebung

$$\hat{\tau} = \arg\max_{\tau}\{\hat{R}_{x_1x_2}(\tau)\} \tag{11}$$

der beiden Signale. Für die diskreten Signalbetrachtungen erfolgt analog eine Angabe von $\hat{\tau}$ in Abtastwerten.

Die im Zeitbereich aufwendige Faltung, wird im Frequenzbereich durch eine Multiplikation ersetzt. Das führt zu einer Vereinfachung der Kreuzkorrelation. Grundlage ist die diskrete Fourier-Transformation (DFT) der Eingangssignale. Eine Abbildung der Signale $x_1(k)$ und $x_2(k)$ in dem Frequenzbereich wird definiert über

$$\underline{X}(n) = \frac{1}{N}\sum_{k=0}^{N-1} x(k)\mathrm{e}^{-\mathrm{j}2\pi\frac{nk}{N}} \quad , n = 0, \ldots, N-1, \tag{12}$$

mit N diskreten Frequenzen (vgl. [HW14]). Das Kreuzleistungsdichtespektrum für die Transformierten $\underline{X}_1(n)$ und $\underline{X}_2(n)$ kann mit

$$\underline{G}_{x_1x_2}(n) = \underline{X}_1(n)\underline{X}_2^*(n) \tag{13}$$

angegeben werden. Der Operator * entspricht der komplexen Konjugation. Eine anschließende Rücktransformation über die inverse diskrete Fourier-Transformation IDFT vereinfacht die Kreuzkorrelation aus (10) zu

$$\hat{R}_{x_1x_2}(\tau) = \sum_{n=0}^{N-1} \underline{G}_{x_1x_2}(n)\mathrm{e}^{\mathrm{j}2\pi\frac{nk}{N}} \quad , k = 0, \ldots, N-1. \tag{14}$$

Die Anwendung der Formel (11) liefert damit die gesuchte zeitliche Verschiebung, die dem Argument mit der höchsten Wahrscheinlichkeit entspricht (Abb. 3).

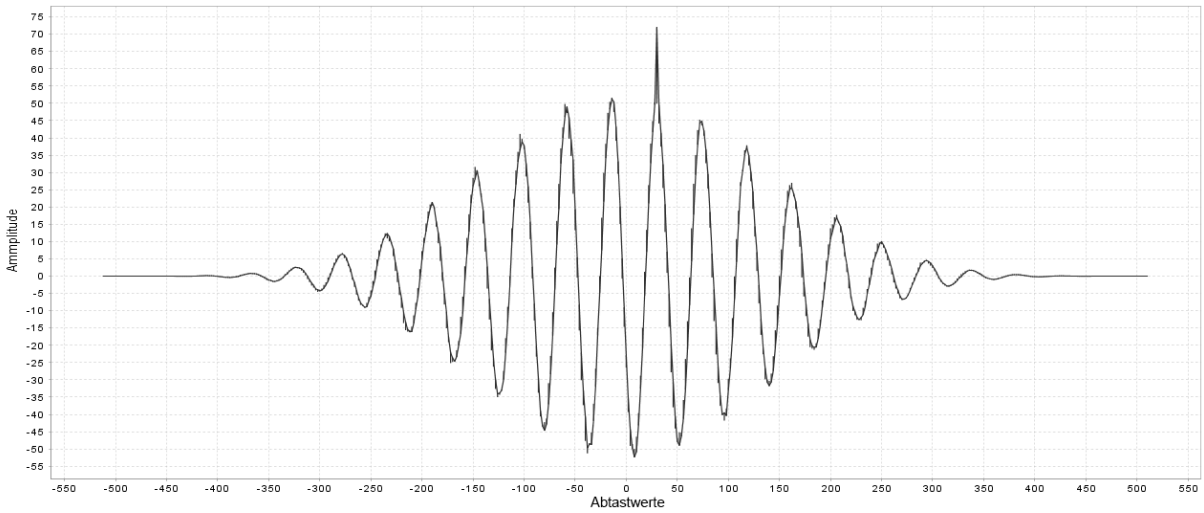


Abbildung 3: Simuliertes Beispiel: Kreuzkorrelationsfunktion zweier mit Rauschen überlagerter $1\,\mathrm{kHz}$ Sinussignale mit einer Verschiebung von $\tau = 30$ Abtastwerten

Mit Abbildung 3 ist offensichtlich, dass ohne eine scharfe Ausprägung des Extremums die Detektion der Zeitverschiebung äußert problematisch ist. Unter nicht idealen Bedingungen wird die Annahme von Schallreflexionen an der Raumoberfläche getroffen, die zu Mehrdeutigkeiten in der Kreuzkorrelation führen können.

2.5 Kohärenz

Die lineare Abhängigkeit von zwei Eingangssignalen (7) lässt sich über die Kohärenz bestimmen. Der Korrelationszusammenhang wird über das Verhältnis vom diskreten Kreuzleistungsdichtespektrum (13) zu den Autoleistungsdichtespektren $G_{x_1x_1}(n)$ und $G_{x_2x_2}(n)$ nach

$$\gamma_{x_1x_2}(n) = \frac{\underline{G}_{x_1x_2}(n)}{\sqrt{G_{x_1x_1}(n)\,G_{x_2x_2}(n)}} \tag{15}$$

definiert. Darin enthalten ist das Leistungsdichtespektrum

$$G_{x_1x_1}(n) = \underline{X}_1(n)\underline{X}_1^*(n) = |\underline{X}_1(n)|^2 \tag{16}$$

für die Fouriertransformierte von $\underline{X}_1(n)$. Die Aufstellung des Autoleistungsdichtespektrums von $\underline{X}_2(n)$ erfolgt analog. Statt (15) findet in der Literatur (vlg. [Bau05], S.5) das Betragsquadrat Anwendung:

$$|\gamma_{x_1x_2}(n)|^2 = \frac{|\underline{G}_{x_1x_2}(n)|^2}{G_{x_1x_1}(n)\,G_{x_2x_2}(n)}. \tag{17}$$

Dieser lineare Zusammenhang wird Magnitude Squared Coherence (MSC) bezeichnet ([Bit01], S. 24). Nach [Dre99] besitzt die MSC die Schranken

$$0 \leq |\gamma_{x_1x_2}(n)|^2 \leq 1. \tag{18}$$

2.6 Rauschprozesse

Rauschprozesse lassen sich gemäß [McC01] in drei Klassen unterteilen (vgl. [McC01], S. 21). Für jeden Typ ist eine konkrete Kohärenz bekannt:

Für den Fall, dass $|\gamma_{x_1x_2}(n)|^2 \approx 1$ entspricht, haben die beiden stochastischen Rauschprozesse einen linearen Zusammenhang. Übertragen auf ein Mikrofonfeld sind die Signale stark korreliert und liegen demnach in einem **kohärenten Rauschfeld**. Die Signalausbreitung erfolgt frei, ohne auf Hindernisse zu stoßen. Das beutet eine Abwesenheit von Reflexionen, Streuungen oder Zerstreuungen. Dieser Fall tritt, bedingt durch Luftverwirbelungen und Temperaturschwankungen von divergenten Rauschprozessen, in der Praxis äußerst selten auf.

Das **inkohärente Rauschfeld** beschreibt genau den umgekehrten Prozess. Konkret sind die Mikrofonsignale unkorreliert und haben eine MSC $|\gamma_{x_1x_2}(n)|^2 \approx 0$. Der Fall tritt ebenso selten in der Realität auf. Jedoch wird bereits das zufällig verteilte elektrische Verhalten der Mikrofonmembran als Rauschen angesehen.

Der letzte und dritte Fall beschreibt das **diffuse Rauschfeld**. Die Signale in dieser Umgebung werden durch schwach korreliertes, normal verteiltes Rauschen überlagert. Der überwiegende Teil der realen Rauschumgebung wird dahingehend gekennzeichnet - hier die Laborumgebung.
Diese Bedingung gilt gleichermaßen, wenn im Raum unendlich viele und unendlich kleine Schallquellen existieren. In dem Fall lässt sich nach [Bau05] die Spaltfunktion

$$|\gamma_{x_1x_2}(n)|^2 = \left[\frac{\sin\left(2\pi n\frac{d}{c}\right)}{2\pi n\frac{d}{c}}\right]^2 \tag{19}$$

$$= \mathrm{si}^2\left(\frac{2\pi n d_{ij}}{c}\right) \tag{20}$$

für eine kugelförmige Mikrofoncharakteristik zusammen mit den Mikrofonabstand d und der Geschwindigkeit im Medium c (hier Luft mit $c = 343{,}2\ \frac{\mathrm{m}}{\mathrm{s}}$) bilden. Es ist offensichtlich, dass sich die Kohärenz bei einem geringen Mikrofonabstand an Eins annähert. Analog geht bei einem großen Abstand die Kohärenz gegen Null.

2.7 Nachhallmodell

Der ideale Fall beschreibt eine isotrope und reflexionsfreie Schallausbreitung in einem freien Schallfeld. Für ein Mikrofonpaar entspricht das einem kohärenten Geräuschfeld mit einer MSC von Eins. Das Modell ist näherungsweise in reflexionsarmen Räumen oder unter freiem Himmel für eine einzelne Schallquelle vorzufinden.

Die Betrachtungen des idealen Modells gelten in der Praxis für den direkten Signalpfad. Jede Reflexion kann als verzögerte, gedämpfte Version des Direktsignals aufgefasst werden. Aus diesem Grund wird das Multipfadmodell eingeführt. Die Erfassung des Mikrofonsignales erfolgt rein rechnerisch über eine Summierung der jeweils gewichteten Pfade. Anwendung findet das Modell überwiegend in submarinen Betrachtungen, wo es zu Beugungen an Wasseroberflächen und Seeböden kommt.

Durch vielfältige Reflexionen der Schallwellen ist es in der Realität nicht möglich, alle Signalpfade einzeln zu erfassen. Aus diesem Grund wird eine Erweiterung auf das Nachhallmodell getroffen. Die Beugungen, Reflexionen und Verzerrungen des Signals $s(k)$ durch den Raum lassen sich über die Raumimpulsantwort $h_i(k)$ mit $i = 1, \ldots, M$ für eine spezifische Schallquellen- und Mikrofonposition beschreiben. Zusätzlich besitzt das Mikrofonsignal ein beigemischtes Rauschen $\eta(k)$. Das dahinter stehende Signalmodell ist in Gleichung (7) definiert.

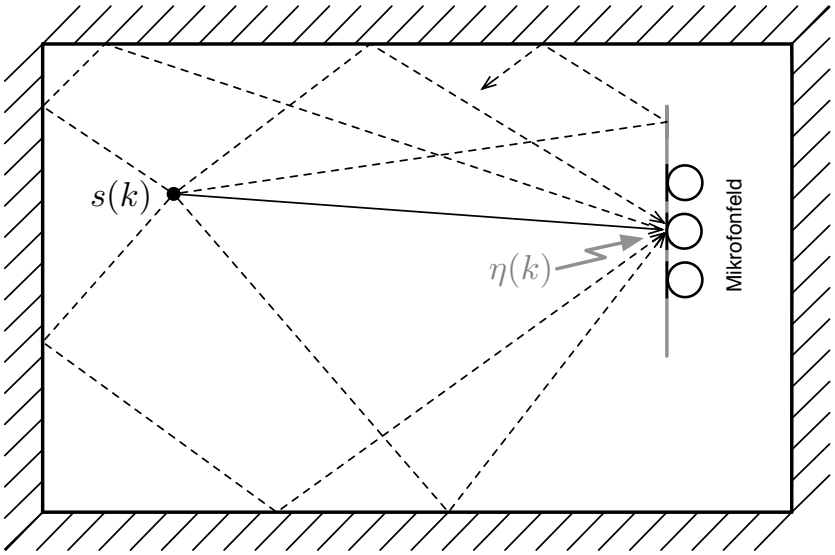


Abbildung 4: Nachhallmodell für den direkten Signalpfad, die korrelierten Reflexionen einer Signalquelle und dem unkorreliertem Rauschen (eigene Abb. nach [CBH06])

3 Robuste Mikrofonfeldverarbeitung

Die Verbesserung der Robustheit bezüglich der Mikrofonfeldverarbeitung setzt eine Analyse der Fehlerquellen voraus. Eine Isolation und die Eliminierung von einzelnen Inkorrektheiten wird in einem zweiten Schritt durchgeführt.

3.1 Problematiken der Lokalisation

Bei der Verarbeitung von Mikrofonfeldsignalen kann eine Vielzahl von Problemen auftreten, die zu einer Verfälschung der Lokalisationsposition führen. Daraus resultiert eine ebenso fehlerhafte Richtcharakteristik des Beamformers nach Gleichung (6).

Die auftretenden Probleme sind vielfältig begründet. Ein mögliches Szenario ist die falsche Wahl des Mikrofonfeldes. Dazu gehört die verwendete Feldgeometrie und die Anzahl der verwendeten Mikrofone. Des Weiteren ist für Genauigkeit der Zielrichtung die Position der einzelnen Mikrofone entscheidend. Der Zusammenhang aus Mikrofonpositionsfehler und dem Lenkfehler ist in Abbildung 5 veranschaulicht. Beide Inkorrektheiten haben direkt auf den Lokalisationsfehler Einfluss (vgl. [KMR12]). Die geschätzte Schallquellenposition entspricht folglich nicht der wahren Position.

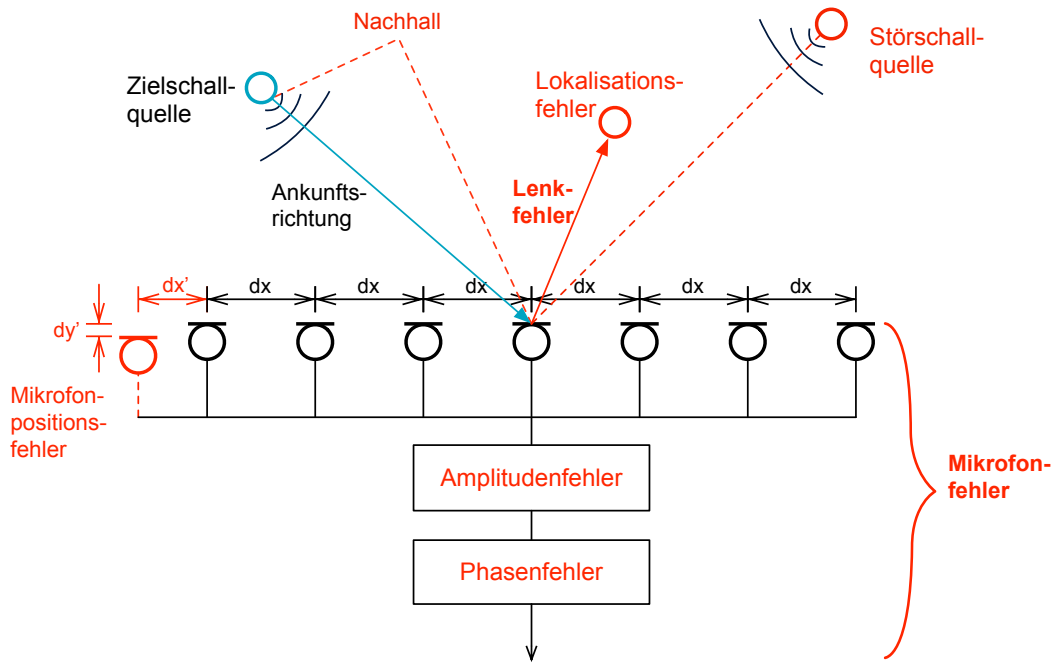


Abbildung 5: Illustration der möglichen Fehlerquellen in der Mikrofonfeldverarbeitung anhand eines linearen, äquidistanten Mikrofonfeldes (eigene Abb. nach [KMR12])

Ein weiterer Fehler entsteht ungeachtet einer einheitlichen Mikrofonbauart. Der mögliche Amplitudenfehler unterliegt den äußeren klimatischen Gegebenheiten. Demnach liegen am Ausgang unterschiedliche Schalldruckpegel an. Das betrifft gleichermaßen die Phasenlage der Mikrofone (vgl. [Moe09], S. 55 ff.).

Auch das Sprachsignal selbst verursacht Probleme bei der Lokalisation. Es wird durch eine große Bandbreite $70\,\mathrm{Hz}$ bis $4\,\mathrm{kHz}$ für Telefonanwendungen sowie bei breitbandiger Verwendung bis zu $10\,\mathrm{kHz}$ (vgl. [HR09], S. 252) charakterisiert. Durch Variation in der Sprachintensität kommt es zu hohen Schwankungen des Signal-Rausch-Verhältnisses (engl. Signal-To-Noise Ratio (SNR)) im Zeitverlauf. Die Sprache besitzt nur innerhalb eines kleinen Zeitfensters von $20 - 30\,\mathrm{ms}$ statistisch stationäre Eigenschaften (vgl. [BAS95b], S. 155).

Von besonderer Bedeutung im Bereich der Richtungsschätzung sind die Betrachtungen der Reflexionen (Abschnitt 2.7) und die hinzukommenden Rauschquellen (Kapitel 2.6). Die räumlichen Gegebenheiten sind für die Lokalisation von äußerster Wichtigkeit. Eine große Anzahl von schallharten Oberflächen würde zu einer Vermehrung an ersten Reflexionen sorgen. Durch den Einsatz von Schallabsorptionsflächen lässt sich dem entgegenwirken (siehe Laborumgebung in Abb. 19, Anhang A.1).

Unkorrelierte Störquellen stellen eine zusätzliche Fehlerquelle dar. Damit verbunden sind unmittelbare Auswirkungen auf das SNR des Mikrofonfeldsignals. Eine Optimierung derartiger Fehler kann durch explizite Steuerung der Richtcharakteristik des Beamformers realisiert werden.

3.2 Fehleroptimierungen

Der überwiegende Teil der Fehlerquellen aus Abschnitt 3.1 wird nachfolgend erörtert.

3.2.1 Verbesserung der Mikrofonfehlstellungen

Ein Fehler betrifft die Positionsdaten der einzelnen Mikrofone. Die Vernachlässigung der Qualitätskontrolle während bzw. nach der Fertigstellung des gesamten Verbundes erforderte eine erneute Vermessung der Mikrofonkoordinaten. Grundlage waren die relativen Positionen aus den Konstruktionsplänen (siehe Anhang A.1.3). Die entsprechenden Koordinaten wurden bereits in das Raumkoordinatensystem transformiert (vgl. [Bir13]). Für das *Mikrofonfeld 1 (TV)* bedeutet das eine Vertauschung von y- und z-Koordinate, sowie das Hinzufügen eines additiven Ausgleichs auf der y-Koordinate. Ein Ausgleich der z-Koordinate von *Mikrofonfeld 2* sorgte parallel für eine Translation an die Raumdecke. Ebenso berücksichtigt wurde dessen Plattenneigung.

In einem ersten Schritt zur Qualitätssicherung fand eine Neuberechnung aller Mikrofonpositionen statt. Die Messungen vom Raummittelpunkt zu den Mikrofonen erfolgten mit einem *Bosch DLE50 Laser-Entfernungsmessgerät*. Zur Fehlerminimierung ergab sich der gemessene Wert aus dem Mittelwert von drei einzelnen Messungen (siehe Tabelle aus Abb. 22 im Anhang A.1.4). Zur Verifikation dienten die Beträge der Ortsvektoren zu den einzelnen Mikrofonen. Alle Mikrofonkoordinaten, die eine Abweichung aufzeigten, wurden daraufhin neu vermessen und die ermittelten Koordinaten in die Tabelle 22 übertragen. Das sorgte für eine händische Optimierung der durch die fehlerhaften Lagedaten entstandenen Abweichungen. Zum näheren Vergleich ist in Abbildung 23 (Anhang A.1.5) aufgezeigt, wie groß die Abweichung der neuen Ortsvektorbeträge von den bestehenden Distanzen ist. Des Weiteren sind in der Tabelle 23 alle Koordinaten und deren Ausgleichswerte aufgelistet.
Die Neuvermessung ergab, dass die Anbringung des Mikrofonfeldes am Hauptbildschirm durch ungleichmäßige Verformungen des Rahmens gekennzeichnet ist. Eine Erklärung für diese Deformierung bleibt offen.

In einem zweiten Schritt fand eine Verifizierung des Neigungswinkels von *Mikrofonfeld 2 (Decke)* statt. Die Mikrofone mit den Nummern 39 und 55 entsprechen den Referenzpunkten, durch ihre Lage auf der y-Achse. Gemessen wurde von der Fußbodenoberfläche bis zur Mitte der Mikrofonmembran. Das ergab die folgenden Entfernungsdaten:

Entfernung Fußboden/Mikrofon	
$h_{\mathbf{m_{55}}}$ in m	$h_{\mathbf{m_{39}}}$ in m
2,465	2,239

Tabelle 1: Entfernungsmessung zur Neigungswinkelbestimmung

Aus den gemessen Daten lässt sich eine Höhendifferenz von

$$a = |h_{\mathbf{m_{55}}} - h_{\mathbf{m_{39}}}| = 0{,}226\,\mathrm{m} \tag{21}$$

ermitteln. Der Abstand beider Mikrofone beträgt:

$$c = \|\mathbf{m_{55}} - \mathbf{m_{39}}\| = 1{,}534\,\mathrm{m}. \tag{22}$$

Aus den Beziehungen im rechtwinkligen Dreieck ist der Neigungswinkel

$$\alpha = \arcsin\left(\frac{a}{c}\right), \tag{23}$$

mit den Seitenlängen von a und c zu bestimmen. Der Winkel für die oben gemessenen Entfernungen beträgt $\alpha = 8{,}47°$. Das bestätigt den bisher angenommen Winkel von $\alpha = 8{,}4°$ aus den Konstruktionsplänen.

Die Realisierung des Rotationswinkels wird durch eine affine Transformation umgesetzt. Für die Rotation um die x-Achse ist infolgedessen die Drehmatrix

$$\mathbf{R_x}(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \tag{24}$$

anzuwenden. Bei einer aktiven Drehung erfolgt eine Transformation der geometrischen Informationen des Ortsvektors. Das Verfahren hat keinen Einfluss auf das verwendete Koordinatensystem. Die Drehung wird im mathematischen positiven Sinn durchgeführt, die einer Rotation gegen den Uhrzeigersinn gleich kommt.

Die bisher angenommenen $2{,}35\,\mathrm{m}$ für die Anbringungshöhe der Plattenkonstruktion wurden ebenfalls in einer Messung mit $h = 2{,}35\,\mathrm{m}$ bestätigt.

Eine Disparität hinsichtlich der Anbringung des beweglichen Schlittensystems konnte weiterhin festgestellt werden. Das Fällen eines Lotes am Mikrofonfeld auf den Raummittelpunkt machte ersichtlich, dass die Deckenkonstruktion um $\Delta x = 0{,}053\,\mathrm{m}$ von den Plänen abweicht. Durch das neue Aufmaß besitzt das Teilmikrofonfeld 2 eine zusätzliche Verschiebung von $\mathbf{p_{offset}} = (0{,}053; 0; 2{,}35)$.

3.2.2 Verarbeitungskette

Zur Verbesserung der Systematik und Minimierung von Fehlerquellen innerhalb des CSL-Projektes ist eine Kette für die Signalverarbeitung entstanden. Diese enthält alle Verarbeitungsmodule und deren Verbindungen (Abb. 6).

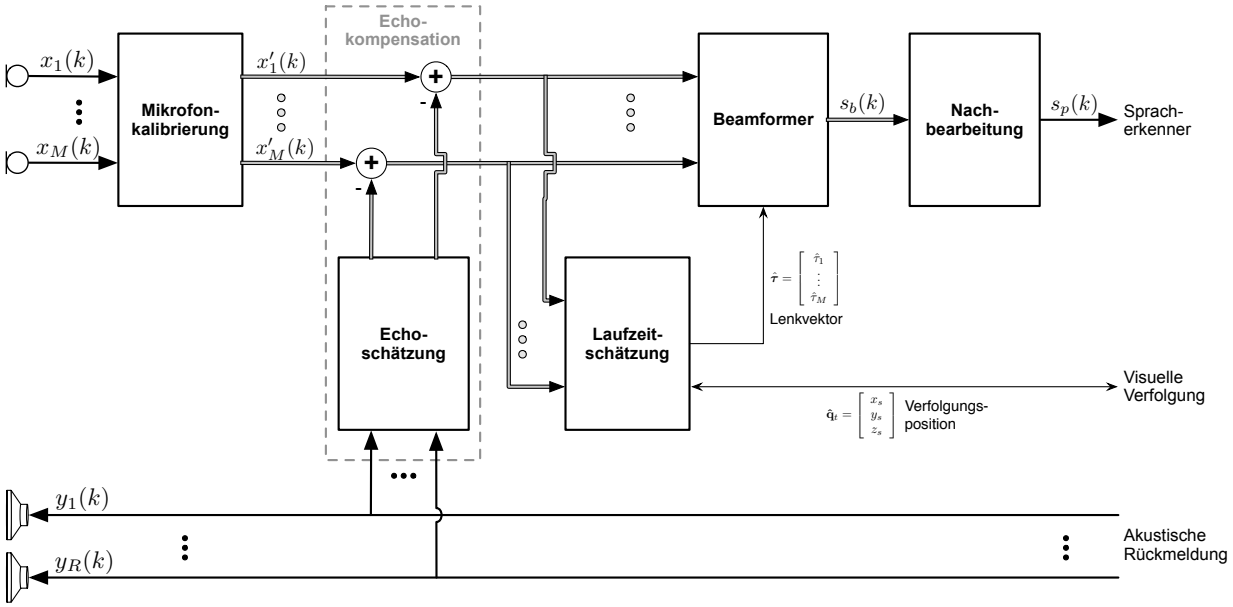


Abbildung 6: Verarbeitungskette des Mikrofonfeldes (eigene Abb. in Anlehnung an [HR09])

Die Grafik 6 zeigt M Mikrofonsensoren. Jedes der Mikrofone liefert ein diskretes Signal über den elektroakustischer Wandler. In einem ersten Schritt steht eine optionale Kalibrierung der Mikrofonsignale bereit (Abschnitt 3.2.3). Hinsichtlich der Mensch-Maschine-Kommunikationseinheit ist nachfolgend eine Echokompensation vorgesehen (Abschnitt 3.2.4). Sie ist begründet durch die Ausgabe von akustischen Rückmeldungen über die Monitorlautsprecher und der damit verbundenen Wiederzuführung der Signale in den Verarbeitungsprozess durch die Mikrofone.

Im oberen Teil folgt auf dem direkten Weg nach der Kalibrierung und Echokompensation die Verarbeitung des Beamformers (Abschnitt 3.2.5). Die kompensierten Signale werden durch einen DSB-Algorithmus zu einem einzigen Ausgangssignal zusammengefasst (siehe Abschnitt 2.3). Der Vorgang ist unter Multiple Input Single Output (MISO) bekannt.

Für die Steuerung des Beamformers kommen Lenkvektoren zum Einsatz. Die Bestimmung ist in der Laufzeitschätzung realisiert. Diese Verarbeitungseinheit stellt die Hauptkomponente der vorliegenden Arbeit dar und ist demzufolge in dem Kapitel 4.2 näher erläutert. Die darin zu bestimmten Positionsdaten dienen weiterführend zur Evaluierung der visuellen Nachverfolgung bzw. Lokalisation und umgekehrt.

Im letzten Schritt der Verarbeitungskette findet eine Nachbearbeitung des Ausgangs vom Beamformer $s_b(k)$ statt (Abschnitt 3.2.6). Das Ziel ist die Angleichung des Signals $s_b(k)$ an das Ausgangssignal $s(k)$ von der Schallquelle.

3.2.3 Kalibrierung von Mikrofonfeldern

In der Mikrofonfeldverarbeitung kann von gleichen Mikrofonempfindlichkeiten und Übertragungsverhalten ausgegangen werden (vgl. [Sar13], S.53). Es treten dennoch häufig Amplitudenfehler bei den verwendeten Mikrofonen auf. Sie können durch vielfältige Gründe entstehen: Sensitivitätsschwankungen von Kondensatormikrofonen durch örtliche Druckänderungen, Regelungenauigkeiten von Verstärkerpotentiometern oder verschiedenartigen Befestigungen der Mikrofone. Um dem entgegen zu wirken und ein invariantes Signal auf allen Kanalwegen zu gewährleisten, ist eine Kalibrierung des gesamten Mikrofonfeldes nötig. Aus diesem Grund wurde die neu entwickelte Verarbeitungskette um eine automatisierte Mikrofonfeldkalibrierung erweitert. Nachfolgend ist das zugrundeliegende mathematische Modell beschrieben.

Die Angleichung der Signale unterliegt der Annahme, dass bei einer konstanten, geräuschreduzierten Raumsituation alle Mikrofonpegel annähernd gleich sind. Die mittlere Leistungen, für eine endliche Anzahl an Mikrofonsignalen, entsprechen folglich einer Normalverteilung.
Die mittlere Signalleistung eines diskreten Eingangssignals x_i mit $i = 1, \ldots, M$ kann über das zweite empirische Moment mit

$$\tilde{x}_i^2 = \frac{1}{K} \sum_{k=1}^{K} \left[x_i(k) \right]^2 , \tag{25}$$

für K Abtastwerte bestimmt werden. Die Anzahl der gewählten Signalabtastwerte darf hierbei nicht zu klein gewählt sein, um eine ausreichend große Stichprobe zu erlangen.

Für die Umsetzung der oben getroffenen Annahme, wird jeder Abtastwert $x_i(k)$ um einen Faktor a_i ausgeglichen. Mit der Erweiterung der Gleichung (25) folgt für die veränderten quadratischen Mittelwerte:

$$\underline{\tilde{x}}_i^2 = \frac{1}{K} \sum_{k=1}^{K} a_i^2 \cdot \left[x_i(k) \right]^2 . \tag{26}$$

Der Durchschnitt aus allen kanalspezifischen Effektivwertquadraten ergibt sich aus

$$\tilde{x}_{ref}^2 = \frac{1}{M} \sum_{i=1}^{M} \underline{\tilde{x}}_i^2 \tag{27}$$

$$= \frac{1}{M \cdot K} \sum_{i=1}^{M} \sum_{k=1}^{K} \left[x_i(k) \right]^2 , \tag{28}$$

für alle M Kanäle und K Abtastwerte. Dies dient der Ermittlung eines globalen Referenzwertes $\tilde{x}_{ref}^2$. Zur Berechnung des gesuchten Verstärkungsfaktors a_i, kann die Wurzel des Quotienten vom Gesamtmittelwertquadrat $\tilde{x}_{ref}^2$ und des kanalspezifschen Effektivwertquadrats $\tilde{x}_i^2$ mit

$$a_i = \sqrt{\frac{\tilde{x}_{ref}^2}{\tilde{x}_i^2}} = \frac{\tilde{x}_{ref}}{\tilde{x}_i} \tag{29}$$

gezogen werden. Somit setzt sich der Verstärkungsfaktor aus dem Verhältnis von Referenzwert bzw.

Gesamtmittelwert $\tilde{x}_{ref}$ zu dem kanalspezifschen Effektivwert $\tilde{x}_i$ zusammen. Das angepasste Eingangssignal $x'_i(k)$ wird abschließend als

$$x'_i(k) = a_i \cdot x_i(k) \tag{30}$$

festgelegt. Wobei der Faktor a_i das Eingangssignal $x_i(k)$ für alle $i = 1, \ldots, M$ Kanäle normalisiert.

3.2.4 Echokompensation

In einem auditiven Mensch-Maschine-Kommunikationssystem gilt die Echokompensation als unabdingbar. Grund dafür sind die erneuten Erfassungen von akustischen Signalen über die Monitorlautsprecher. Dies sorgt für eine erhebliche Beeinträchtigung von nachgeschalteten Algorithmen und muss kompensiert werden. Eine dahinter liegende Laufzeitschätzung könnte möglicherweise einen Lautsprecher an Stelle eines Sprechers detektieren. Die Lautsprecherpositionen sind weiterhin als nicht plausibel einzustufen.
Die Entwicklung eines Echo Cancellation System (ECS) ist nicht Bestandteil der gestellten Aufgabe und wird hier vernachlässigt. Durch Deaktivierung der akustischen Rückmeldungen wird sichergestellt, dass es zu keinen Störungen kommt. Für den effektiven Einsatz des CSL-Gesamtsystems ist zu empfehlen, die Entwicklung eines ECS voranzubringen. Mögliche vielfältige Erweiterungen der Verarbeitungskette hinsichtlich eines ECSs präsentieren [HR09] (Seite 232 ff.) oder [BW01] (Seite 281 ff.).

3.2.5 Verbesserung der Beamformer-Methodik

Der bisher eingesetzte DSB (Abschnitt 2.3) ist für multiple Quellen und direkten, kohärenten Schalleinfall von Störquellen als suboptimal zu bewerten. Der Vorteil des einfachen Aufbaus eines DSB wird durch die schlechte Auflösung in Bändern mit niedrigen Frequenzen und der generellen Schmalbandigkeit aufgehoben (siehe Abb. 24, Anhang A.2) (vgl. [McC01], S. 34). Das begründet einen Austausch oder eine Verbesserung des eingesetzten Verfahrens.

Eine Optimierung ist bezüglich einer fixen oder adaptiven Technik möglich. Beide Verfahren unterteilen sich in verschiedenen Methoden von Beamforming, die jeweils durch Vor- und Nachteile gekennzeichnet sind (konkrete Auflistung in [McC01], S. 35). Die Korrektur des verwendeten DSB ist nicht Bestandteil der Arbeit.

Wichtig für die Lokalisation ist, dass eine Nachführung des Beamformers ausschließlich in Sprachpausen vorgenommen wird. Sollte das keine Beachtung finden, kommt es zu linearen Verzerrungen und Dämpfungen des Sprechersignals.

3.2.6 Signalnachbearbeitung

Um die Ausgaben des Beamformers effektiv nutzen zu können, ist eine Nachbearbeitung notwendig. Beamforming bewirkt in einigen Fällen eine Dämpfung von verschiedenen Frequenzbändern in Abhängigkeit von der jeweiligen akustischen Situation. Gleiches gilt für eine Abnahme der Leistung durch eine Echounterdrückung. Dementsprechend ist eine Aufbereitung der Signaldaten in der Nachbearbeitung notwendig. Das gewünschte Ausgangssignal sollte in dem Zusammenhang keine weiteren Veränderungen oder Verzerrungen erfahren. Eine denkbare Realisierung wäre ein dynamischer Nachfilter, der die wesentlichen Spektralanteile bewertet.

Darüber hinaus kommt es vor, dass durch die vorgeschalteten Prozesse die Sprachsignale übermäßig vom Rauschen befreit wurden. Für diesen Fall wäre ein Hinzufügen vom sogenannten „Komfortrauschen“ denkbar.

Zur verbesserten Steuerung der Signalaufbereitung in der Nachbearbeitung wird zusätzlich angeregt, einen Rückkopplungszweig zum Beamformer mit in die Verarbeitungskette aufzunehmen. Grund dafür ist eine verbesserte adaptive Steuerung im Bezug auf den Eingangs-/Ausgangs-SNR.

Die hier aufgezeigten Ansätze sind ebenso nicht Bestandteil der vorliegenden Arbeit, dienen jedoch zur Orientierung für spätere Realisierungen.

4 Akustische Laufzeitschätzung und Quellenlokalisation

Ein Segment wurde im vorangegangen Abschnitt 3.2.2 bewusst ausgelassen - das der Schätzung von akustischen Laufzeiten und der daraus resultierenden Lokalisation der Schallquelle. Diese beiden Komponenten gelten als wichtigste Bestandteile des automatisierten Beamformings. Es stellt die Grundlage der Lenkvektoren bzw. Verzögerungsglieder für die einzelnen Mikrofonkanäle dar.
Grundsätzlich muss zwischen Lokalisation und Nachverfolgung (engl. Tracking) unterschieden werden. Ersteres zielt auf die Bestimmung der Position innerhalb eines Zeitintervalls ab. Im Gegensatz dazu berücksichtigt eine Nachverfolgung zuvor ermittelte Positionsinformationen. Eine effektive Steuerung und Nachführung des Beamformers kann nur durch eine Nachverfolgung realisiert werden. Zur späteren Vervollständigung wurde das Modul bereits in die Verarbeitungskette zur Laufzeitschätzung mit aufgenommen (siehe Abb. 7).

Die Bestimmung der ortsspezifischen Verzögerungsglieder wurde bisher durch eine punktuelle Flächenabtastung mit anschließender Leistungsberechnung am Ausgang des Beamformers realisiert. Das Konzept der Steered Response Power (SRP) ist durch einen hohen rechnerischen Aufwand gekennzeichnet. Für die Lokalisation mit der SRP-Methode mussten bisher 6.348 diskrete Raumpositionen für die Ortung verglichen werden (vgl. [Bir13]). Zusätzliche Optimierungen hinsichtlich der Körpergröße eines einzelnen Sprechers nach [Bir14] und der damit verbundenen Unterraumsuche brachten keine hinreichenden Verbesserungen in der Laufzeit. Infolgedessen wird ein neues, leistungsfähigeres System zur Sprecherlokalisation benötigt.

Das Problem der passiven Schätzung der Ankunftsrichtung (engl. Direction Of Arrival, kurz DOA) ist in der Literatur auf mannigfaltige Art und Weise beschrieben ([HR09], [BW01], [KMI$^+$07], [AM02]). Es existieren verschiedene Techniken zur Bestimmung. Einige dieser Ansätze sind für die dreidimensionale Lokalisation im verwendeten Mikrofonfeldverbund als ungeeignet einzustufen. Eine häufig verwendete Technik ist die Multiple Signal Classification (MUSIC). Sie arbeitet mit der Zerlegung der Eigenwertmatrix, die aufgrund der Orthogonalität von Signal- und Rauschprozessen eine Unterraumtrennung beinhaltet. MUSIC ist lediglich auf äquidistante, lineare Mikrofonanordnungen anwendbar (vgl. [KKB13], [Cha05]). Der korrespondierende ESPRIT-Algorithmus (Estimation Of Signal Parameters Via Rotational Invariance Techniques) nutzt ein ähnliches Korrelationssignalmodell. Er bietet durch die Interpolation eines linearen Mikrofonfeldes ebenso keine Abhilfe. Beide Algorithmen haben als Resultat einen Elevationswinkel, der für die dreidimensionale Ortung nicht ausreichend ist und erfahren keine Verwendung.
Die Literatur schlägt mit der MVDR-Technik (Minimum Variance Distortionless Response) einen weiteren Ansatz vor (vgl. [BW01], [Bit01]). Dieser stützt sich auf verschiedene sogenannte „superdirektive Beamformer“ und einer modifizierten Kohärenzfunktion. Das Ziel des MVDR ist es den Ausgang des Beamformers hinsichtlich der Varianz für ein spezielles diffuses Geräuschfeld zu optimieren. Bedingt durch eine fehlende Entscheidungsmöglichkeit zwischen zwei planaren Wellen, findet der MVDR-Ansatz in der Praxis selten Verwendung (vgl. [KKB13]).

Eine vielversprechendere Methode ist die Schätzung der relativen Verzögerungszeiten und der anschließenden Lokalisation unter Betrachtung eines möglichen Schätzfehlers (vgl. [Dre99]). Ein entsprechender zweistufiger Prozess wird in der Laufzeitschätzung im weiteren Verlauf vorgestellt.

Diese Art der Schätzung unterliegt keiner Restriktion hinsichtlich Aufbau und Größe des verwendeten Mikrofonfeldes und bietet damit die optimale Voraussetzung für die dreidimensionale Ortung.

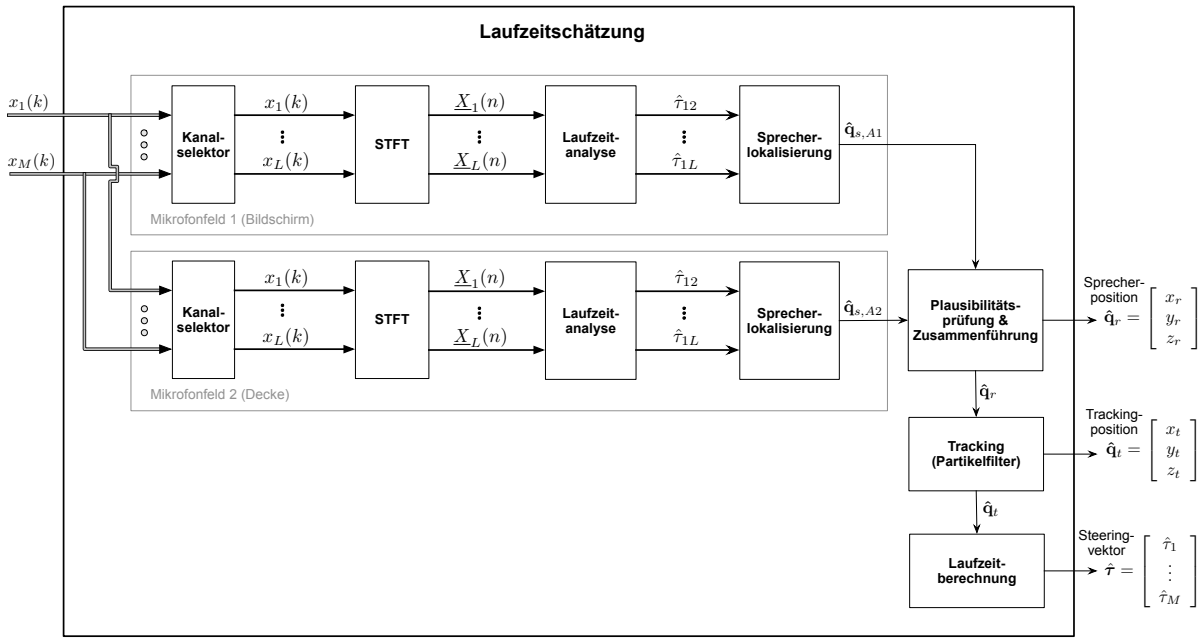


Abbildung 7: Verarbeitungskette für die Laufzeitschätzung (eigene Abb. nach [Dre99])

Zur Minimierung der Ausprägung von störenden Nebenkeulen bei Mikrofonfeldern ist zu empfehlen, den Inter-Mikrofonabstand gering zu halten (vgl. [McC01]). Als Konsequenz muss die Signalverarbeitung jeweils getrennt für die beiden Teilfelder erfolgen (siehe Abb. 7). Der geringere Mikrofonabstand sorgt für eine Reduktion von Mehrdeutigkeiten in der TDOA.

4.1 Kanalselektor

Eine Aufwandsreduktion kann durch einen Kanalselektor nach [Dre99] vorgenommen werden. Für diesen Zweck wird eine Auswahl von L Mikrofonen aus der Gesamtheit von M Mikrofonsensoren getroffen, unter der Beschränkung von $L \geq 4$ für die spätere Lokalisation. Eine derartige Restriktion ergibt sich aus dem Umstand, dass die Bestimmung der Quelle aus den Schnittpunkten von mindestens drei Hyperboloiden bestehen muss. Vergleichen lässt sich ein Hyperboloid mit einem unendlich langen, approximierten Kegel.
Die aus einer Schätzung hervorgehenden Verzögerungszeiten $\hat{\tau}_i$ für die Schnittpunktermittlung lassen sich aus $L-1$ Mikrofonpaaren bestimmen, mit $i = 1, \ldots, L-1$ (siehe Abb. 8).

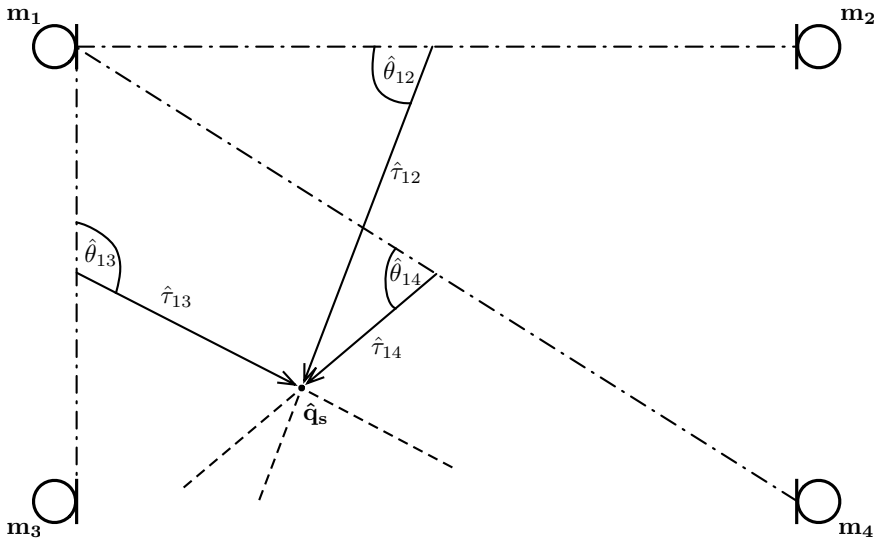


Abbildung 8: Visualisierung des Schnittpunktes für eine geschätzte Schallquelle $\hat{\mathbf{q}}_\mathbf{s}$ und einer rechteckigen Mikrofonfeldgeometrie mit $L = 4$ Mikrofonen

Eine weitere Einschränkung wurde hinsichtlich des Referenzmikrofones getroffen. Zur Vereinfachung von nachfolgenden Iterationsprozessen und ohne Verlust der Allgemeinheit wird das Mikrofon mit dem Index $i = 1$ für das jeweilige Teilfeld als Referenz festgelegt. Mit der Vereinbarung eines Referenzmikrofons folgt der Autor den Empfehlungen der Literatur (vgl. [HB04], [Dre99] oder [BAS97]).

Der durch [Dre99] eingeführte Kanalselektor ist das Resultat von verschiedenen Untersuchungen bezüglich des Lokalisationsfehlers und der gewählten Mikrofonanzahl. Eine zu dem Mikrofonfeld 1 (Abb. 20, Anhang A.1) vergleichbare Konstruktion findet genau dort Anwendung. Seinen Nachforschungen nach besitzt der verwendete Mikrofonfeldverbund mit $L \geq 16$ keine signifikante Verbesserungen hinsichtlich der Lokalisation. Im Gegenzug sollte die Anzahl nicht zu gering gewählt werden. In Bezug auf die eigene Realisierung wird eine Auswahl von $L = 22$ Mikrofone für das jeweilige Teilfeld getroffen. Zu beachten war eine vollständig Abdeckung des Mikrofonfeldes.

4.2 Laufzeitanalyse

Die Grundlage für die Laufzeitschätzung ist die spektrale Kreuzkorrelation (Abschnitt 2.4). Hinsichtlich der Detektion der zeitlichen Verzögerung wurde aufgezeigt, dass es zu Problemen mit Mehrdeutigkeiten kommt, die durch Nachhall oder Reflexionen entstehen. Eine durch [KC76] entwickelte Methode soll diesem Problem entgegenwirken. Sie sieht eine Generalisierung der Kreuzkorrelation (Generalized Cross-Correlation, kurz GCC) vor und wird im weiteren Verlauf ausführlich behandelt (vgl. [BAS97], [Mer02], [MA04], [Aar03]).

Zwei in (7) beschriebene diskrete, fouriertransformierte Eingangssignale lassen sich über das in (13) definierte Kreuzleistungsdichtespektrum $\underline{G}_{x_1x_2}(n)$ als

$$\underline{G}_{y_1y_2}(n) = \underline{H}_1(n)\underline{H}_2^*(n)\underline{G}_{x_1x_2}(n) \tag{31}$$

ausdrücken. Darin enthalten sind die Übertragungsfunktionen $\underline{H}_1(n)$ und $\underline{H}_2(n)$, die das Systemverhalten abbilden und die Kreuzleistungsdichte $\underline{G}_{y_1y_2}(n)$ vollständig beschreiben. Analog wird das Kreuzleistungsdichtesprektrum der beiden Systemfunktionen als allgemeine Filterfunktion

$$\underline{\Psi}(n) = \underline{H}_1(n)\underline{H}_2^*(n) \tag{32}$$

definiert. Es erfolgt eine Erweiterung der Kreuzkorrelationsfunktion (14) zu

$$\hat{R}_{x_1x_2}(\tau) = \sum_{n=0}^{N-1} \underline{\Psi}(n)\underline{G}_{x_1x_2}(n)\mathrm{e}^{\mathrm{j}2\pi\frac{kn}{N}}. \tag{33}$$

Für den Fall einer festen Belegung der Analysefunktion mit einem Skalar

$$\underline{\Psi} = 1 \tag{34}$$

ergibt sich die ungewichtete Kreuzkorrelation nach (14). Vergleichbare feststehende Filter finden wegen der nicht vorhandenen Verbesserung in Bezug auf die TDOA selten Anwendung.

Ein besseres Verhalten bietet eine Gruppe von adaptiven Filterfunktionen. In der Literatur häufig Verwendung findet die PHAT-Gewichtung (Phase Transform) (vgl. [BHC08], [MA04], [JCR13], [PA13], [AM02]):

$$\underline{\Psi}_{PHAT}(n) = \frac{1}{\left|\underline{G}_{x_1x_2}(n)\right|}. \tag{35}$$

Das Betragsspektrum einer gewichteten Kreuzkorrelation mit der PHAT-Filterfunktion wird zu Eins normiert, ohne eine Veränderung des Phasengangs vorzunehmen. Die Literatur bezeichnet das als ein „pre-whitening“ des Spektrums. Die PHAT-Gewichtung setzt eine geringe Nachhallzeit und ein großen SNR voraus.

Der Smoothed Coherence Factor (SCOT) ist ein weiterer Vertreter der GCC-Filterfunktionen. Er nutzt

die Korrelationseigenschaften der Kohärenz. Der SCOT-Filter

$$\underline{\Psi}_{SCOT}(n) = \frac{1}{\sqrt{G_{x_1x_1}(n)\,G_{x_2x_2}(n)}} \tag{36}$$

setzt sich analog zu der Kohärenz aus der Wurzel der Autoleistungsdichtespektren $G_{x_1x_1}(n)$ und $G_{x_2x_2}(n)$ zusammen. Folglich bildet dieser verbunden mit der spektralen Kreuzkorrelation die Kohärenzfunktion (15).

Ein weiterer Filter in der Gruppe ist die Maximum Likelihood (ML) Gewichtung:

$$\underline{\Psi}_{ML}(n) = \frac{|\underline{X}_1(n)|\,|\underline{X}_2(n)|}{|\underline{H}_1(n)|^2\,|\underline{X}_2(n)|^2 + |\underline{H}_2(n)|^2\,|\underline{X}_1(n)|^2}. \tag{37}$$

Der ML-Filter erzielt eine Gewichtung der Kreuzkorrelation durch die Betragsspektren $|\underline{X}_1(n)|$ und $|\underline{X}_2(n)|$, den Autoleistungsdichtespektren $|\underline{X}_1(n)|^2$ und $|\underline{X}_2(n)|^2$ nach (16) sowie den quadratischen Beträgen der Übertragungsfunktionen $|\underline{H}_1(n)|^2$ und $|\underline{H}_2(n)|^2$. Die Filterfunktion entspricht im Aufbau der Kohärenz (15). Eine Anwendung der MSC sorgt für eine äquivalente Definition als Hannon und Thomson (HT) Filterfunktion:

$$\underline{\Psi}_{HT}(n) = \frac{|\gamma_{x_1x_2}(n)|^2}{\left|\underline{G}_{x_1x_2}(n)\right|\,[1 - |\gamma_{x_1x_2}(n)|^2]}. \tag{38}$$

Für einen geringen SNR zeigen HT- und ML-Gewichtungen deutliche Schwächen auf (vgl. [AM02], [BAS97]).

Unter den vorgestellten Filterfunktionen kristallisiert sich in der Literatur der Phase Transform (PHAT)-Filter heraus. Im Vergleich der drei Filterfunktionen bietet diese PHAT-Gewichtung die präziseste Ausprägung des Maximums. Demnach wird die Schätzung aus (33) für die allgemeine Betrachtung zu

$$\hat{R}_{1i;PHAT}(\tau) = \sum_{n=0}^{N-1} \frac{1}{|\underline{X}_1(n)\underline{X}_i^*(n)|}\underline{X}_1(n)\underline{X}_i^*(n)\mathrm{e}^{\mathrm{j}2\pi\frac{kn}{N}} \tag{39}$$

$$= IDFT\left\{\frac{\underline{X}_1(n)\underline{X}_i^*(n)}{|\underline{X}_1(n)\underline{X}_i^*(n)|}\right\} \tag{40}$$

erweitert.

Die erneute Anwendung der Formel (11) gibt an, dass die gesuchten, paarweisen Verzögerungen

$$\hat{\tau}_{1i} = \arg\max_{\tau}\{\hat{R}_{1i;PHAT}(\tau)\}, \tag{41}$$

mit der größten Wahrscheinlichkeit, dem Argument des maximalen Funktionswertes entsprechen (siehe Bsp. Abb. 9). Die diskrete Schätzung von $\hat{\tau}_{1i}$ mit $i = 2, \ldots, L$ hat eine Betrachtung in Abtastwerten zur Folge.

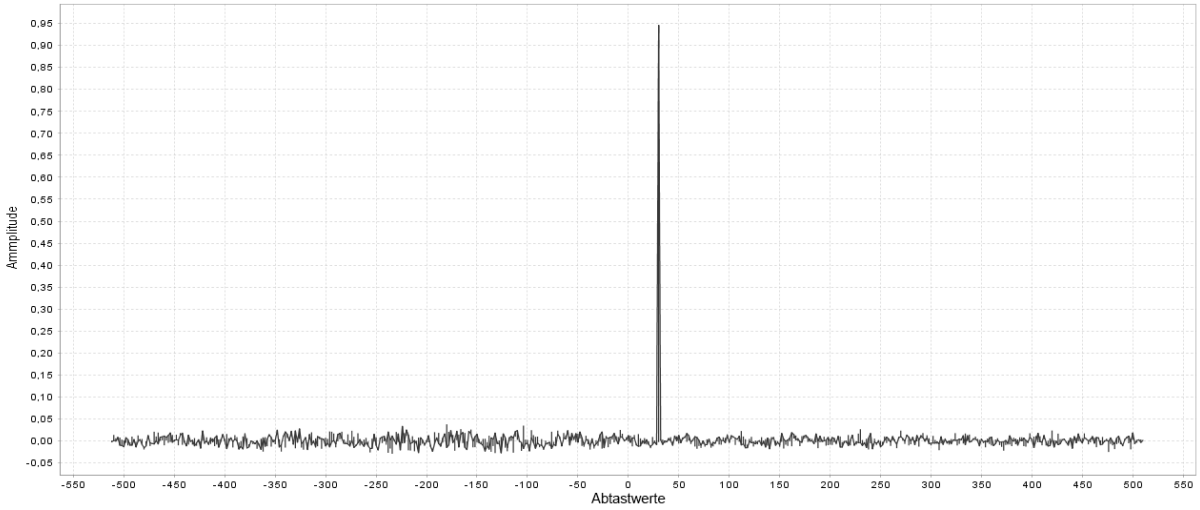


Abbildung 9: Simuliertes Beispiel: Kreuzkorrelation mit PHAT-Gewichtung zweier mit Rauschen überlagerter 1 kHz Sinussignale mit einer Verschiebung von $\tau = 30$ Abtastwerten (vgl. Abb. 3)

Im Vergleich zur Abbildung 3 ist mit der PHAT-Gewichtung eine deutliche Verbesserung in der Ausprägung des Maximalwertes ersichtlich. Zur Vermeidung von Mehrdeutigkeiten, wird weiterhin eine Suchoptimierung unter Verwendung des bekannten Mikrofonabstands d_{1i} getroffen:

$$\hat{\tau} \in \left[-d_{1i}\frac{f_a}{c}, d_{1i}\frac{f_a}{c} \right]. \tag{42}$$

Darin enthalten ist die Abtastfrequenz f_a und die Geschwindigkeit im Medium c (hier Luft mit $c = 343{,}2\,\frac{\mathrm{m}}{\mathrm{s}}$, bei $20\,^{\circ}\mathrm{C}$ Raumtemperatur). Diese Schranken berücksichtigen den Umstand, dass die Schallwelle die Distanz zwischen den Mikrofonen nur innerhalb eines bestimmten Zeitintervalls zurücklegen kann.

Für eine Stichprobe von K Abtastwerten besitzt jedes Mikrofonpaar nach Formal (39) und (41) einen Skalar $\hat{\tau}_{1i}$ für $i = 2, \ldots, L$. Zusammengefasst lässt sich der Vektor mit allen paarweisen Verzögerungszeiten

$$\hat{\boldsymbol{\tau}} = \begin{bmatrix} \hat{\tau}_{12} \\ \vdots \\ \hat{\tau}_{1L} \end{bmatrix} \tag{43}$$

angeben. Weiterhin kann keine explizite Aussage über die Position der Schallquelle mit dem Verzögerungsvektor $\hat{\boldsymbol{\tau}}$ getroffen werden. Für eine korrekte Ortung des Fokuspunktes muss eine anknüpfende Sprecherlokalisation stattfinden.

4.3 Sprecherlokalisation

Die aus dem Abschnitt 4.2 getroffenen Überlegungen haben als Resultat die paarweisen Verzögerungszeiten (43). Eine zielsichere Lokalisation oder Festlegung eines Lenkvektors ist bis zu diesem Punkt nicht möglich. Nachfolgend wird eine Lösung präsentiert, die alle fehlenden Informationen bereitstellt.

Für ein Mikrofonfeld mit M Mikrofonen sind die jeweiligen kartesischen Koordinaten definiert durch:

$$\mathbf{m_i} = [x_i, y_i, z_i]^\top, i = 1, \ldots, M. \tag{44}$$

$[\cdot]^\top$ entspricht der Transponierten des Vektors. Die Schallquellenposition kann weiterhin mit

$$\mathbf{q_s} = [x_s, y_s, z_s]^\top \tag{45}$$

beschrieben werden. Die Entfernung vom Ursprung zu der Quelle $\mathbf{q_s}$ ergibt sich über deren Betrag aus

$$\|\mathbf{q_s}\| = \sqrt{x_s^2 + y_s^2 + z_s^2}. \tag{46}$$

Analog dazu wird für die Mikrofonposition

$$\|\mathbf{m_i}\| = \sqrt{x_i^2 + y_i^2 + z_i^2}, i = 1, \ldots, M \tag{47}$$

angegeben. Ohne Verlust der Allgemeinheit, lassen sich mit dem Referenzmikrofon $L - 1$ reduzierte Paare im Mikrofonfeld bilden (siehe Kanalselektor Abschnitt 4.1). Jedes Mikrofonpaar besitzt zwei spezifische Raumkoordinaten: $\mathbf{m_1}$ für das Referenzmikrofon und $\mathbf{m_i}$ mit $i = 2, \ldots, L$ für ein weiteres konkretes Mikrofon. Der positionelle Zusammenhang ist in der Abbildung 10 dargestellt. Der Mittelpunkt

$$\mathbf{p}_{1i} = \frac{\mathbf{m_1} + \mathbf{m_i}}{2} \tag{48}$$

des Mikrofonpaares, gelegen auf der Achse

$$\mathbf{a_{1i}} = \frac{\mathbf{m_1} - \mathbf{m_i}}{\|\mathbf{m_1} - \mathbf{m_i}\|}, \tag{49}$$

kann nach *Brandstein et al.* [BAS95a] als Ursprung eines sich aufspannenden Hyperboloiden gekennzeichnet werden. Auf dessen Rotationsachse befindet sich in unbekannter Entfernung der gesuchte Zielpunkt $\mathbf{q_s}$.

Unter einer angenommenen Fernfeldbedingung für die Schallquelle und der damit verknüpften planaren Wellenfront lässt sich für jedes Mikrofonpaar ein Peilwinkel

$$\hat{\theta}_{1i} = \arccos\left(\frac{c\hat{\tau}_{1i}}{f_a d_{1i}}\right) \tag{50}$$

bestimmen ([BAS95b], S. 162). Wobei f_a für die gewählte Abtastfrequenz und c für die Geschwindigkeit im Medium stehen. Der Winkel $\hat{\theta}_{1i}$ kann für ein spezifisches $\hat{\tau}_{1i}$ Werte von 0° bis 180° annehmen. Die Festsetzung der Ankunftsrichtung ergibt sich aus dem Verhältnis der ermittelten Zeitdifferenz $\hat{\tau}_{1i}$ zum

paarigen Mikrofonabstand d_{1i}. Der Abstand zwischen den beiden Mikrofonen wird über die euklidische Norm

$$d_{1i} = \|\mathbf{m_1} - \mathbf{m_i}\| \tag{51}$$

$$= \sqrt{(x_1 - x_i)^2 + (y_1 - y_i)^2 + (z_1 - z_i)^2} \tag{52}$$

ermittelt. Durch die Beziehung vom Peilwinkel $\hat{\theta}_{1i}$ aus Gleichung (50) und Zeitdifferenz $\hat{\tau}_{1i}$ ergibt sich, dass die gezeigte DOA gleichbedeutend mit der TDOA ist.

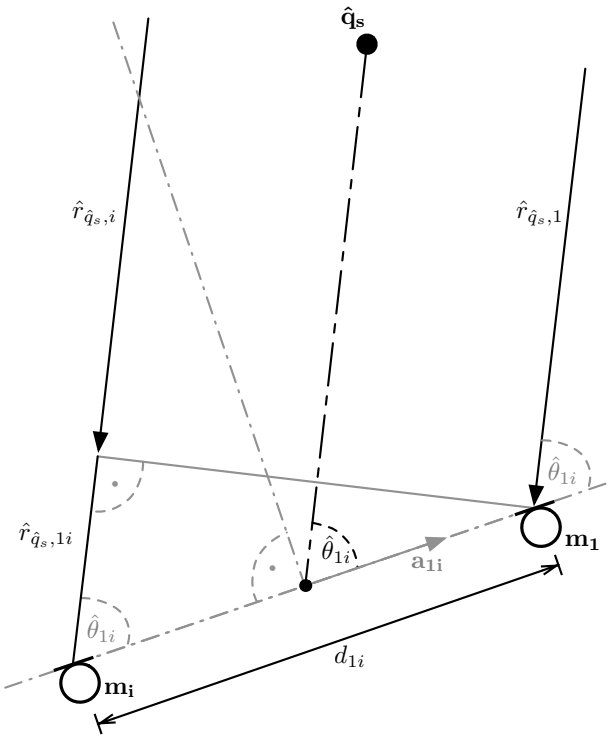


Abbildung 10: Geometrische Zusammenhänge zwischen Mikrofonsensoren und geschätzter Schallquellenposition $\hat{\mathbf{q}}_\mathbf{s}$

Die Festsetzung der paarweisen Peilwinkel bieten eine gute Möglichkeit zur lokalen Evaluierung der DOA für das jeweilige Mikrofonfeld.

Weitere Überlegungen der geometrischen Zusammenhänge lassen darauf schließen, dass für die ermittelten paarweisen Verzögerungsglieder τ_{1i} aus der Gleichung (41) die folgende Beziehung gilt (vgl. [Dre99], S. 93):

$$\tau_{1i} = \tau_1 - \tau_i. \tag{53}$$

Die darin enthaltenen Verzögerungen τ_1 und τ_i sind bislang unbekannt. Weitere Überlegungen sind dahingehend zu treffen.

In der Zeit τ_{1i} legt die Schallwelle die Entfernung

$$r_{q_s,1i} = \tau_{1i}\frac{c}{f_a} \tag{54}$$

zurück. Enthalten ist die Zeitdifferenz τ_{1i} aus der Schätzung (39), die Geschwindigkeit im Medium c und die Abtastfrequenz f_a. Aus (53) und (54) ergibt sich der aus Abbildung 10 gezeigte geometrische Zusammenhang

$$r_{q_s,1i} = r_{q_s,1} - r_{q_s,i}. \tag{55}$$

Die enthaltenen Entfernungen $r_{q_s,1}$ und $r_{q_s,i}$ können analog über die euklidische Norm aus (52) bestimmt werden:

$$r_{q_s,1} = d_{q_s,1} = \|\mathbf{q_s} - \mathbf{m_1}\|, \tag{56}$$

$$r_{q_s,i} = d_{q_s,i} = \|\mathbf{q_s} - \mathbf{m_i}\|, i = 2, \ldots, L. \tag{57}$$

Die Positionen $\mathbf{m_i}$ und $\mathbf{m_1}$ entsprechen den jeweiligen Mikrofonkoordinaten und $\mathbf{q_s}$ der gesuchten Zielschallquelle.

Durch die bisherige Annahme von einer planaren Wellenfront im Fernfeld und aus Mehrdeutigkeiten in Schätzung nach (39) ergibt sich unter Nahfeldbedingungen ein Fehler $J(\mathbf{q_s})$. Diese Abweichung von dem realen Positionsvektor $\mathbf{q_s}$ wird über die quadratische Fehlerbedingung

$$J(\mathbf{q_s}) = \sum_{i=2}^{M} \underbrace{\left[\hat{\tau}_{1i} - T(\{\mathbf{m_1}, \mathbf{m_i}\}, \mathbf{q_s})\right]}_{\Delta\mathbf{q_s}}^2 \tag{58}$$

mit

$$T\left(\{\mathbf{m_1}, \mathbf{m_i}\}, \mathbf{q_s}\right) = \frac{f_a}{c}\left(\|\mathbf{q_s} - \mathbf{m_1}\| - \|\mathbf{q}_s - \mathbf{m_i}\|\right) \tag{59}$$

ausdrückt (siehe Abb. 11) (Least Square (LS)-Estimator). Folglich ergibt sich die prognostizierte Zielposition aus der Fehlerfunktion $J(\mathbf{q_s})$ durch

$$\hat{\mathbf{q}}_\mathbf{s} = \arg\min_{\mathbf{q_s}}\{J(\mathbf{q_s})\}. \tag{60}$$

Die geschätzte Position $\hat{\mathbf{q}}_\mathbf{s}$ ergibt sich aus der Minimierung der Fehlerfunktion in Abhängigkeit des realen Quellenstandortes $\mathbf{q_s}$. Demnach muss eine Laufzeitdifferenz $\hat{\tau}_{1i}$ ermittelt werden, die mit hoher Wahrscheinlichkeit der wahren Zielposition $\mathbf{q_s}$ entspricht.

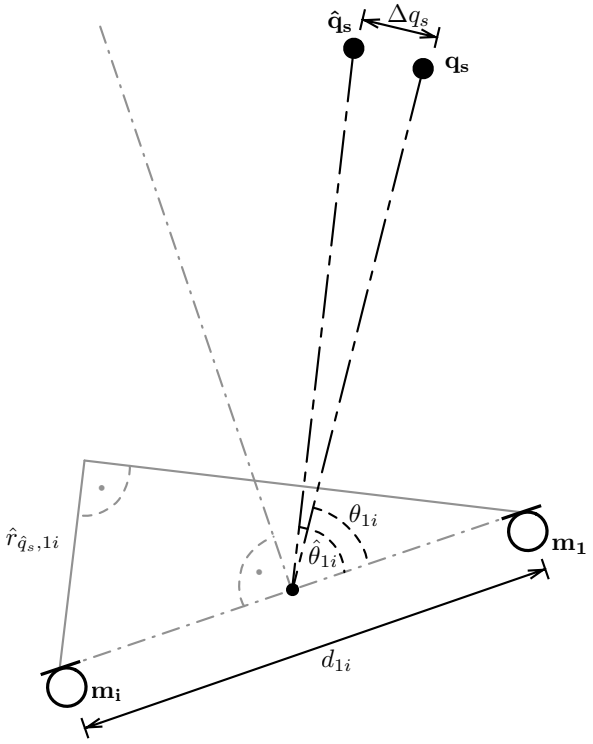


Abbildung 11: Darstellung des Positionsfehler zur Richtungsschätzung

Die Fehlerfunktion (58) verhält sich nicht linear zu der gesuchten Position und besitzt folglich kein triviales Ergebnis. Aus diesem Grund wird eine geschlossene Direction Of Arrival (DOA)-Lösung eingeführt. Entwickelt wurde die Methode von [CH94] und findet in [Dre99], [Dre96] und [TSA08] Anwendung. Ausgehend von einem möglichst kleinen Fehler in der Laufzeitschätzung von (41) ist der Ansatz auf die Fern- und Nahfeldbedingung anwendbar. Die Verfahrensweise ist eine iterativen Lösung und gilt als Approximation des LS-Schätzers (60).

Beginnend kann nach [Dre99] der positionelle Zusammenhang von einer möglichen Schallquelle zu allen relevanten Mikrofonsensoren über

$$\mathbf{G_x}\hat{\mathbf{q}}_\mathbf{s} - \mathbf{h} = \mathbf{0} \tag{61}$$

definiert werden. Darin enthalten ist der Vektor

$$\hat{\mathbf{q}}_\mathbf{s} = \begin{bmatrix} x_{q_s} \\ y_{q_s} \\ z_{q_s} \\ r_{q_s,1} \end{bmatrix} \tag{62}$$

mit der unbekannten Sprecherposition $\hat{\mathbf{q}}_\mathbf{s}$, erweitert um den Skalar $r_{q_s,1}$. Dieser Wert kommt der Distanz von Sprecherposition $\hat{\mathbf{q}}_\mathbf{s}$ zum Referenzmikrofon $\mathbf{m_1}$ gleich.

Darüber hinaus wird die Matrix

$$\mathbf{G_x} = \begin{bmatrix} x_1 - x_2 & y_1 - y_2 & z_1 - z_2 & r_{q_s,12} \\ x_1 - x_3 & y_1 - y_3 & z_1 - z_3 & r_{q_s,13} \\ \vdots & \vdots & \vdots & \vdots \\ x_1 - x_L & y_1 - y_L & z_1 - z_L & r_{q_s,1L} \end{bmatrix} \tag{63}$$

definiert. Die Gleichung enthält alle zur Lokalisation verwendeten Mikrofonkoordinaten aus $\mathbf{m_i}$ für $i = 2, \ldots, L$ sowie die paarweise Entfernungen aus Formel (54). Der zweite Vektor

$$\mathbf{h} = \frac{1}{2} \begin{bmatrix} r_{q_s,12}^2 + x_1^2 + y_1^2 + z_1^2 - x_2^2 - y_2^2 - z_2^2 \\ r_{q_s,13}^2 + x_1^2 + y_1^2 + z_1^2 - x_3^2 - y_3^2 - z_3^2 \\ \vdots \\ r_{q_s,1L}^2 + x_1^2 + y_1^2 + z_1^2 - x_L^2 - y_L^2 - z_L^2 \end{bmatrix} \tag{64}$$

in (61) beinhaltet die Differenz des quadratischen Fehlers aus den Entfernungs- bzw. Positionsdaten. (61) besitzt als Lösung einen Nullvektor.

Faktisch sind die Distanzen $r_{q_s,1i}$ Schätzwerte aus der Laufzeitschätzung (41) und besitzen im nicht rauschfreien Fall einen Fehler. In Folge dessen wird der Nullvektor in (61) durch einen Fehlervektor zu

$$\mathbf{G_x}\hat{\mathbf{q}}_\mathbf{s} - \mathbf{h} = \boldsymbol{\epsilon} \tag{65}$$

ersetzt. Die Fehlerfunktion aus (58) lässt sich zusammen mit (65) als

$$J(\mathbf{q_s}) = \left(\mathbf{h} - \mathbf{G_x}\hat{\mathbf{q}}_\mathbf{s}\right)^\top \mathbf{W}_\epsilon^{-\mathbf{1}} \left(\mathbf{h} - \mathbf{G_x}\hat{\mathbf{q}}_\mathbf{s}\right) \tag{66}$$

in eine Matrixschreibweise überführen, wobei $\mathbf{W}_\epsilon^{-\mathbf{1}}$ für die invertierte Kovarianzmatrix des Fehlervektors steht. Durch Minimierung des Fehlers nach (60) und unter der Annahme, dass alle Elemente im Lösungsvektor $\hat{\mathbf{q}}_\mathbf{s}$ linear unabhängig sind, wird die Vereinfachung

$$\hat{\mathbf{q}}_\mathbf{s} = \left(\mathbf{G}_\mathbf{x}^\top \mathbf{W}_\epsilon^{-\mathbf{1}} \mathbf{G_x}\right)^{-1} \mathbf{G}_\mathbf{x}^\top \mathbf{W}_\epsilon^{-\mathbf{1}} \mathbf{h} \tag{67}$$

getroffen. Die darin enthaltene Kovarianzmatrix $\mathbf{W}_\epsilon$ für den Fehlervektor $\boldsymbol{\epsilon}$ lässt sich, unter der Bedingung eines möglichst großen SNR, über die Beziehung

$$\mathbf{W}_\epsilon = \mathbf{B}\mathbf{W}_\tau\mathbf{B} \tag{68}$$

definieren (vgl. [TSA08], S. 43). Die Matrix $\mathbf{B}$ enthält die wahren Abstände $r_{q_s,i}$ zwischen der Quelle und dem i-ten Mikrofon für $i = 2, \ldots, L$. Alle Kovarianzen für die geschätzten Verzögerungszeiten $\hat{\tau}_{1i}$ sind in der Matrix $\mathbf{W}_\tau$ zusammengefasst.

Die gesuchte Diagonalmatrix ist

$$\mathbf{B} = \begin{bmatrix} r_{q_s,2} & 0 & \cdots & 0 \\ 0 & r_{q_s,3} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{q_s,L} \end{bmatrix}. \tag{69}$$

Die Matrix $\mathbf{B}$ ergibt sich aus der in Gleichung (55) gezeigten Beziehung, unter Verwendung von $r_{q_s,1}$ aus dem Lösungsvektor $\hat{\mathbf{q}}_\mathbf{s}$.

Für die erste Iteration wird eine Approximation getroffen, um die fehlenden wahren Abstände $r_{q_s,i}$ aus $\mathbf{B}$ zu kompensieren. Aus der Annahme, dass die Schallquelle sich im Fernfeld befindet und die gesuchten Abstände $r_{q_s,i}$ nicht fehlerbehaftet sind, kann (70) zu

$$\hat{\mathbf{q}}_\mathbf{s} \approx \left(\mathbf{G}_\mathbf{x}^\top \mathbf{W}_\tau^{-\mathbf{1}} \mathbf{G}_\mathbf{x}\right)^{-1} \mathbf{G}_\mathbf{x}^\top \mathbf{W}_\tau^{-\mathbf{1}} \mathbf{h} \tag{70}$$

approximiert werden. Weil die realen Entfernungen in $\mathbf{B}$ fehlen, wird die Kovarianzmatrix des Distanzfehlers $\mathbf{W}_\epsilon$ durch eine A-Priori-Kovarianzmatrix

$$\mathbf{W}_\tau = \begin{bmatrix} 1 & 0{,}5 & \cdots & 0{,}5 \\ 0{,}5 & 1 & \cdots & 0{,}5 \\ \vdots & \vdots & \ddots & \vdots \\ 0{,}5 & 0{,}5 & \cdots & 1 \end{bmatrix} \tag{71}$$

für den Schätzfehler der Verzögerung ersetzt. Die darin enthaltenen Kovarianzen entsprechen einer Funktion des emittierten Sprechersignals und der eingestreuten Rausch- und Störsignale (vgl. [Dre99], S. 96).

Zusammenfassend wurde hier eine zweistufige Lösung vorgestellt, die eine iterative Vorgehensweise beinhaltet. Dieser Prozess umfasst mit der ersten Schätzung nach Gleichung (70), unter Verwendung von (71) noch kein konkretes Ergebnis. Das im Lösungsvektor enthaltene $r_{q_s,1}$ wird benutzt, um die Matrix $\mathbf{B}$ über Gleichung (55) aufzustellen. In einer zweiten Stufe mit dem Gleichungssystem (67) wird $\mathbf{B}$ über (68) zugeführt.
Der zweite Schritt kann so lange wiederholt werden, bis die Lösung dem gesuchten Zielvektor entspricht. Untersuchungen von [Dre99] zeigten eine ausreichende Annäherung an den Lösungsvektor nach der ersten Iteration (vgl. [Dre99], S. 97).

4.4 Plausibilitätsprüfung

Durch die unterschiedliche Orientierung der beiden Teilfelder kann für das frontseitige Mikrofonfeld 1 (TV) nur die x- und z-Koordinate geschätzt werden. Analog dazu erfolgt eine Bestimmung von x- und y-Koordinaten für das zweite Mikrofonfeld an der Raumdecke. Unter der Annahme, dass für einen möglichen Sprecher die Blickrichtung überwiegend zum Bildschirm gerichtet ist, bekommt der x-Wert aus dem Mikrofonfeld 1 mehr Gewichtung.

Die Prüfung auf plausible Positionen in der Lokalisation ist einfach gehalten: Eine Orientierung anhand der vorgegebenen Raumgeometrie stellt eine Schranke für die jeweiligen Zielvektoren dar. Anders formuliert, kann sich eine Sprecherposition nicht hinter einer Wand, auf einem Tisch oder außerhalb der Fensterfront befinden. Das sorgt für eine Beschränkung auf die Grundfläche von $4{,}4 \times 4{,}4\,\mathrm{m}$ (vgl. [Bir13]). Aus dem Kapitel 3.2.1 wurde eine Anbringungshöhe von $2{,}35\,\mathrm{m}$ für das Mikrofonfeld 2 (Decke) bestimmt. Die ermittelte Höhe entspricht der oberen Schranke für die z-Koordinate.

Für die Zusammenlegung und Prüfung auf Plausibilität gilt der Zielvektor:

$$\hat{\mathbf{q}}_\mathbf{r} = \begin{bmatrix} x_{\hat{\mathbf{q}}_\mathbf{s},A1} | - 2{,}2 \geq x_{\hat{\mathbf{q}}_\mathbf{s},Array1} \leq 2{,}2 \\ y_{\hat{\mathbf{q}}_\mathbf{s},A2} | - 2{,}2 \geq y_{\hat{\mathbf{q}}_\mathbf{s},Array2} \leq 2{,}2 \\ z_{\hat{\mathbf{q}}_\mathbf{s},A1} | \quad 0 \geq z_{\hat{\mathbf{q}}_\mathbf{s},Array1} \leq 2{,}35 \end{bmatrix} \quad \forall\{x,y,z\} \in \mathbb{R}. \tag{72}$$

Innerhalb des beschränkten Messfeldes kann sich der Sprecher frei im Raum bewegen, ohne auf Hindernisse zu stoßen.

4.5 Laufzeitberechnung

Die für das Beamforming benötigten Laufzeiten in (6) können nachfolgend über die Entfernung zwischen geschätzter Zielposition $\hat{\mathbf{q}}_\mathbf{r}$ und den Positionsdaten für die einzelnen Mikrofone $\mathbf{m_i}$, mit $i = 1, \ldots, M$ über

$$\hat{\tau}_i = \frac{f_s}{c}(d_{max} - d_{\hat{\mathbf{q}}_\mathbf{r}\mathbf{m_i}}) \tag{73}$$

bestimmt werden. Dabei entspricht $d_{\hat{\mathbf{q}}_\mathbf{r}\mathbf{m_i}}$ der euklidischen Norm analog zu der Formel (52) und der größtmöglichen Distanz

$$d_{max} = \max(d_{\hat{\mathbf{q}}_\mathbf{r}\mathbf{m_1}}, d_{\hat{\mathbf{q}}_\mathbf{r}\mathbf{m_2}}, \ldots, d_{\hat{\mathbf{q}}_\mathbf{r}\mathbf{m_M}}) \tag{74}$$

zum Fokuspunkt $\hat{\mathbf{q}}_\mathbf{s}$. Das Distanzmaximum entspricht der Bestimmung des entferntesten Mikrofons von der Schallquelle. Folglich erfahren alle Eingangssignale $x_i(n)$ eine Verzögerung, bis die Schallwelle beim am weitesten entfernten Mikrofon eintrifft. Alle aus (73) ermittelten Lenkverzögerungen lassen sich im Lenkvektor

$$\hat{\boldsymbol{\tau}} = \begin{bmatrix} \hat{\tau}_1 \\ \hat{\tau}_2 \\ \vdots \\ \hat{\tau}_M \end{bmatrix} \tag{75}$$

zusammenfassen. Der Vektor $\hat{\boldsymbol{\tau}}$ sorgt für eine dreidimensionale Steuerung des Fokuspunktes vom Beamformer und bezieht sich auf alle $M = 64$ Mikrofone.

4.6 Diskussion und Weiterführendes

In dem vorangegangenen Kapitel wurde ausführlich auf die Lokalisation von akustischen Quellen eingegangen. Zur Verbesserung der Ortung sind einige Erweiterungen notwendig.

Ein Aspekt richtet sich an die Lokalisation innerhalb der Sprachpausen. Durch den fortwährenden Prozess der Erkennung werden ebenso Nicht-Sprachsegmente mit analysiert. Folglich ist eine Erweiterung der bestehenden Verarbeitungskette (Abb. 7) um eine Erkennung von Sprachaktivität erforderlich. Erst im Anschluss daran sollte eine Evaluierung der Ankunftsrichtung von Sprachsegmenten erfolgen.

Durch die Abhängigkeit von der Abtastfrequenz kann es zu einem Schätzfehler von der wahren Verzögerung kommen. Aus diesem Grund ist eine Interpolation von $\hat{\tau}$ zur Fehlerminimierung sinnvoll (siehe [Hof08], S. 21 und [Dre99], S. 91).

Ein weiterer Aspekt, der sich auf mögliche nachfolgende Abhandlungen und Implementierungen bezieht, ist der einer intervallübergreifenden Nachverfolgung (engl. Tracking). Die bisherige Verarbeitung von Informationen über ein Zeitintervall hinaus muss gesondert betrachtet werden. Grund dafür sind etwaige Daten aus einem menschlichen Bewegungsmodell und gleichzeitige Lokalisation von mehreren Sprechern. Zur Realisierung wurde bereits die Verarbeitungskette (Abb. 7) um das Modul eines möglichen Partikelfilters ergänzt ([Our07], [HR10], [WLW03], [SH14], [KMI$^+$07]).

5 Realisierung

Die Realisierung der in Kapitel 3 und 4 vorgestellten Verbesserungsschritte und des Lokalisationsalgorithmus erfolgte in der objektorientierten Programmiersprache Java. Als Entwicklungsumgebung (engl. Integrated Development Environment, kurz IDE) diente Eclipse unter der Verwendung des aktuellen Java-Entwicklungswerkzeug Java SE Development Kit (JDK) (Version 8, siehe Lst. 1).
Die implementierten Quellcodes sind von sekundärer Bedeutung. Der Leser der vorliegenden Arbeit soll in diesem Abschnitt über die Schnittstellen informiert werden, um bei Bedarf das CSL-Softwareprojekt weiterzuentwickeln. Darüber hinaus sind alle Quellcodes auf der beigefügten CD enthalten. Als alternative Bezugsquelle wird das zentrale Projektarchiv „Apache Subversion“ (SVN) von der Professur Kommunikationstechnik empfohlen.

Hinweis zur Notation: In den nachfolgenden Abschnitten wird für Klassen ein **dick** gedruckter und für Methoden ein *kursiver* Schriftschnitt verwendet. Klassenattribute sind in einem `schmalen geneigten` Stil dargestellt.

5.1 Refactoring von Geräteabhängigkeiten

Die Arbeiten zur Implementierung setzen ein ausführliches „Refactoring“ aller bestehenden geräteabhängigen Klassen voraus. Der Begriff des „Refactoring“ bezeichnet eine manuelle bzw. automatische Restrukturierung eines Softwareprojektes zur Verbesserung der Funktion, Verständlichkeit und Wartbarkeit (vgl. [Whi16]).

Bevor mit der eigentlichen Realisierung begonnen werden konnte, war eine umfangreiche Neuimplementierung verschiedener Klassen notwendig. Es fehlten Schnittstellen für Geräteabhängigkeiten und eine ausreichende Robustheit des Gesamtsystems. Betreffende reorganisierte Java-Klassen sind folgend aufgelistet oder können dem beigefügten Unified Modeling Language (UML) Klassendiagramm entnommen werden (Abb. 26, Anhang A.4):

- **HammerfallAudioDevice** als Ableitung der neuen abstrakten **APortAudioInputDevice**,
- **Trolley** mit den zugehörigen Kindklassen **Motor** und **Lasersensor**,
- **MicArrayState** und der dazu gehörigen **Steering**-Klasse,
- **DSBeamformer** sowie deren Vaterklasse **Beamformer3D**.

Zur Verarbeitung der diskreten Mikrofonsignale wurde eine neue stabile Geräteklasse **HammerfallAudioDevice** eingebunden, die ein Empfang und eine Verteilung der Audiodaten übernimmt. Sie ist eine abgeleitete Klasse der abstrakten Implementierung von **APortAudioInputDevice**. Die Wiederverwendung von Quellcode trägt zu einer besseren Strukturierung und Wartbarkeit von neuen gerätespezifischen Klassen und deren Implementierungen bei.

Alle konkreten Realisierungen eigenständiger Geräteklassen sind von **AAtomicHardware** abgeleitet. Analog dazu existiert die Klasse **ACompositeHardware** für einen Geräteverbund. Beide abstrakten Vorgaben leiten sich von der Klasse **AHardware** ab (siehe Abb. 25, Anhang A.4). Auf die konkrete

Umsetzung der allgemeinen Geräteklassen wird nicht weiter eingegangen. Für die entsprechende Dokumentation sei auf die *JavaDoc* in der Klasse **AHardware** verwiesen.

Zur **APortAudioInputDevice** und zum **HammerfallAudioDevice** ist zu erwähnen, dass für alle Audiokarten die Programmierschnittstelle (engl. Application Programming Interface (API)) JavaPortAudio (JPA) Verwendung findet. Audioparameter, wie Rahmengröße und Abtastfrequenz werden über die konkreten Ableitungen der abstrakten Realisierung definiert. Die in JPA intern laufenden Threads (dt. Fäden, leichtgewichtige Prozesse) übernehmen die Versorgung der Signalverarbeitungsklassen mit den Audiodaten. Dafür zuständig ist die

```
protected void callback(PaBuffer,PaBuffer,int){...}
```

Methode. Die Delegierung der Eingabedaten wird innerhalb der Methode vorgenommen („*Don't call us, we call you*"-Prinzip).

Die in Kapitel 3.2.3 vorgestellte Mikrofonfeldkalibrierung findet unmittelbar in der „*callback()*"-Methode Anwendung. Ausgelöst wird die Kalibrierung manuell über die Funktion „*calibrate()*". Nach einem Intervall von $4\,\mathrm{s}$ erfolgt die Ermittlung der Faktoren zur Anpassung der Amplituden innerhalb des **RmsLevelingCalibrator**. Die jeweiligen Mikrofonkanäle werden hierdurch mit einem Verstärkungsfaktor gewichtet. Somit ist für jede signalverarbeitende Klasse ein kalibriertes Mikrofonfeldsignal garantiert.

Ein weiteres Refactoring wurde bezüglich die Motorsteuerung und der in Verbindung stehenden Lasersensormessung realisiert. Die Modifikation für die Regelkreissteuerung betraf die Klasse **Trolley** zur variablen Positionsänderung des deckenseitigen Mikrofonfeldes. Weiterhin sind die Implementierungen der dazugehörigen Klassen **Motor** (Schrittmotorsteuerung) und **LaserSensor** (Distanzüberprüfung) neu implementiert worden. Fortwährende Programmabstürze aufgrund vielfältiger Fehler begründeten eine neue Umsetzung (siehe Abhandlung [Hes15]).

Neu ist die Objektklasse **MicArrayState**. Sie beinhaltet alle Informationen bezüglich des dreidimensionalen Mikrofonfeldes. Durch Aufruf der statischen, öffentlichen Methode

```
public static synchronized MicArrayState getCurrentState(){...}
```

können Benutzer oder andere Geräteklassen auf den aktuellen Zustand des Mikrofonfeldes zugreifen. Durch den Aufruf wird ein Zusammenführen von Daten aus unterschiedlichsten Klassen ausgelöst. Der Zugriff auf die Attribute der Objektinstanz ist statisch durchzuführen. Jede Klasse, die spezifische Informationen bezüglich des Mikrofonfeldes benötigt, kann durch ein **MicArrayState**-Objekt alle benötigten Parameter erhalten. In der Tabelle 2 ist eine Übersicht zu allen Attributtypen aufgelistet.

Attribut	**Datentyp**	**Ursprungsklasse**	**Verwendungsziel**
target	Point3d	**DoAEstimator**	Fokuspunkt des Mikrofonfeldes
positions	Point3d[]	**MicArrayViewer** &**MicArrayCeiling**	Positionsdaten der Mikrofone
trolleyPos	float	**MicArrayCeiling** (indirekt aus **Trolley/LaserSensor**)	Schienenposition des Deckenmikrofonfeldes
steerVec	float[]	**Steering**	Vektor mit relat. Verzögerungszeiten für 3D-DSB
delays	float[]	**Steering**	absolut. Verzögerungszeiten
gains	float[]	-	Eingangsverstärkung
activeMics	boolean[]	**MicArray3D**	aktivierte/deaktivierte Mikrofone
numberOfActiveMics	int	**MicArray3D**	Anzahl der aktiven Mikrofone

Tabelle 2: Zustandsattribute des **MicArrayState**-Objektes für das dreidimensionale Mikrofonfeld

Zum Tragen kommt das **MicArrayState**-Objekt unter anderem in der letzten Neuerung des Refactorings. Die Klasse **Beamformer3D** benötigt den Lenkvektor aus Gleichung (75), der in Abhängigkeit von *target* und *positions* in **Steering** generiert wurde. Weiterhin steht die Mikrofonfeldklasse **Beamformer3D** in direkter Verbindung mit **HammerfallAudioDevice**. Alle eingehenden Audiosignale werden unmittelbar nach der Kalibrierung an den Beamformer weitergeleitet (siehe Verarbeitungskette Kapitel 3.2.2). Die Realisierung der Signalzusammenlegung aus der Formel (6) geschieht für den DSB in der Basisklasse **DSBeamformer**. Das ermöglicht einen problemlosen Datenaustausch zwischen Eingabe, Verarbeitung (**DSBeamformer**) und Ausgabe auf einer weiteren auditiven Schnittstelle **MAudioDeviceLine34** (siehe UML-Klassendiagramm Abb. 26, Anhang A.4). Die verwendete Vererbungseinbindung ermöglicht einen reibungslosen Austausch der Methodik im Beamforming.

5.2 Realisierung der Quellenlokalisation

Eine wichtige Neuerung stellt der **DoAEstimator** mit allen dazugehörigen Unterklassen dar (siehe UML-Klassendiagramm in Abb. 27). Darin enthalten sind alle öffentlichen Benutzerschnittstellen.
Der Ablauf der internen Verarbeitung ist an die sequenziell arbeitende Laufzeitschätzung (Abb. 7, Kapitel 4.2) angelehnt. Die Klasse kann über das Entwurfsmuster „Singleton“ (dt. Einzelstück) direkt referenziert werden oder erfährt eine automatisierte Initialisierung zusammen mit **MicArray3D** (siehe UML-Klassendiagramm Abb. 27, Anhang A.4). Für weiterführende Informationen bezüglich der Entwurfsmuster von Softwareentwicklungen wird auf die Referenz [ES07] verwiesen.

Entworfen und implementiert wurde der **DoAEstimator** als geschlossenes System, um störende Einflüsse durch den Benutzer zu umgehen. Eine der wichtigsten Funktionen in der Klasse ist die Haltung der Information über den Zielpunkt des Beamformers. Die dafür zur Verfügung stehenden Methoden zum Setzen und Holen des Positionsvektors sind über

```
1 public synchronized Point3d getTargetSource() {...}
2 public synchronized void setTargetSource(Point3d target) {...}
```

implementiert. Das manuelle Festlegen eines Fokuspunktes ist nur möglich, wenn die Methode *isAutoMode()* einen „FALSE“-Wert zurückliefert. Die damit verbundene automatische Laufzeitschätzung ist für diesen Fall deaktiviert. Korrespondierend dazu existiert zur Aktivierung die öffentliche Methode *setAutoMode(boolean)*.

Damit sind die Möglichkeiten zur benutzerseitigen Steuerung bereits erschöpft. Die tatsächliche Implementierung ist weitgehend verborgen. Zur Dokumentation wird weiterhin auf die interne Verarbeitung eingegangen.
Wie bereits in Abschnitt 5.1 beschrieben, übernimmt das **HammerfallAudioDevice** die Zulieferung von Audiodaten. Die Speicherung der Daten erfolgt direkt in einer Queue (dt. Warteschlage) aus *callback()* heraus (siehe [Ora15b]). Der Aufruf ist über die Methode

```
1 public void addAudioDataToQueue(float[][] framesPerChannel){
2     audioDataQueue.add(framesPerChannel);
3 }
```

realisiert. Die Notwendigkeit einer Warteschlange entstand durch die begrenzten Rechenkapazitäten und der nicht vorhandenen Möglichkeit zur unmittelbaren Verarbeitung von Audiodaten. Die Verwendung einer spezifischen „LinkedBlockingQueue“ gestattet es, ein sicheres „Leser-Schreiber-Problem“ zu modellieren. Die Verarbeitung der eingehenden Daten übernimmt ein überwachender, leichtgewichtiger Prozess (engl. thread), der in einem Intervall die Queue überprüft. Im Fall von existierenden Daten erfolgt ein Aufruf der nicht öffentlichen Methode *runDoAEstimator()*.

Die sequenzielle Verarbeitung wird in den folgenden Schritten durchgeführt:

(I) Entnahme der Audiodaten aus der Warteschlage,
(II) Verdoppelung der Audiorahmen,
(III) Auswahl der Mikrofonkanäle,
(IV) Schätzung der relativen Laufzeiten,
(V) Lokalisation auf der Grundlage der geschätzten Laufzeiten,
(VI) Plausibilitätsprüfung und Zusammenführung.

Anknüpfend sind die einzelnen Phasen ausführlicher erläutert. Dabei werden die Schritte **(III)** bis **(VI)** für jedes Teilmikrofonfeld separat betrachtet und verarbeitet. Die Art der Ausführung bleibt mit den vorgestellten Phasen gleich. Lediglich eine Unterscheidung in den Daten wird getroffen.

Die in Schritt **(I)** durchgeführte Entnahme der Audiorahmen aus der „LinkedBlockingQueue“ dient nur der Vermeidung eines Speicherüberlaufes. Die Ausführung verläuft stetig, selbst über eine deaktivierte Lokalisation hinaus.

Auf der Stufe **(II)** wird eine Verdoppelung der Rahmengröße zur Analyse durchgeführt. Standardisiert sind 512 Abtastwerte für die Eingabepuffer der Audiokarte vorgesehen. Bei einer Abtastfrequenz von $44100\,\mathrm{Hz}$ entspricht das $11{,}6\,\mathrm{ms}$ an Audiodaten. Ungeachtet der kurzen Stationarität von Sprache, weisen unter anderem [JCR13] und [LZBK11] nach, dass für die Laufzeitschätzung eine Rahmengröße von mindestens 1024 Abtastwerten sinnvoll ist. Dem entspricht die hier implementierte Konkatenation.

Der nachfolgende Schritt **(III)** verwendet die Klasse **ChannelSelector** für eine Auswahl der notwendigen Mikrofone und deren korrespondierenden Signale zur Analyse und Lokalisation (siehe Kanalselektor Abschnitt 4.1). Zur Verbesserung der iterativen Abläufe in den Feldern wurde für den Zweck der temporären Datenhaltung der Datentyp „HashMap“ eingeführt (vgl. [Ora15a]):

```
1 public void fillAudioHM(float[][] framesPerChannel, ConcurrentHashMap<Integer,
      AudioData> audioDataHM, Point3d[] positions, MicArray array) {...}
```

Dieser assoziative Speicher beinhaltet zwei Parameter: Einen identifizierenden Schlüsselwert vom Typ Integer und ein neu definiertes Datenobjekt vom Typ **AudioData** (siehe UML-Klassendiagramm Abb. 27, Anhang A.4). Das neue Objekt besitzt die Möglichkeit des Vorhaltens der Mikrofonpositionsdaten, deren Eingangssignale und die entsprechenden fouriertransformierten Spektralverteilungen. Das bietet die Option eines fortwährenden Zugriffs auf einzelne mikrofonspezifische Informationen über den gesamten Verarbeitungsprozess hinweg. Darüber hinaus wird eine variable Wahl des Referenzmikrofons zur Paarbildung unterstützt (siehe Kapitel 4.1). Im Gegensatz zu herkömmlichen Felddatenstrukturen können die Mikrofone in der HashMap leichter identifiziert werden. Die Mikrofonauswahl ist im **ChannelSelector** weiterführend klassenintern verborgen. Ein Abruf der Informationen ist dennoch über zwei Methoden möglich:

```
1 public boolean[] getChannelSelection(MicArray array){...}
2 public boolean[] getTotalChannelSelection() {...}
```

Zur Unterscheidung der Informationen aus den jeweiligen Teilmikrofonfeldern dient die Aufzählung

MicArray in den beiden voranstehenden Programmauszügen (siehe UML-Klassendiagramm, Abb. 27, Anhang A.4).

Stufe **(IV)** beschäftigt sich mit der Schätzung von relativen Laufzeiten in Klasse **DelayAnalysis**. Zur Übergabe der HashMap wurde die Methode

```
1 public void run(ConcurrentHashMap<Integer, AudioData> audioDataHM, ConcurrentHashMap
      <String, TauData> tauDataHM, int refMicId) {...}
```

implementiert. Sie bekommt über die **AudioData**-Objekte und dem Index des Referenzmikrofons hinaus eine weitere HashMap übergeben. Diese „tauDataHM“ genannte Entität vom Typ **TauData** beinhaltet die Rückgabewerte um **DelayAnalysis**, die im nachfolgenden Schritt (V) Anwendung findet (siehe Abb. 27, Anhang A.4).
Die Datenverarbeitung innerhalb der Klasse **DelayAnalysis** entspricht der im Abschnitt 4.2 geschilderten Vorgehensweise. Es erfolgt zu Beginn eine Transformation der zeitdiskreten Eingangssignale in den Frequenzbereich, realisiert durch die Klasse **FFT**. Sie dient als Verpackung für die von der *Apache Software Foundation* implementierten Klasse „FastFourierTransformer“. Der Rückgabewert entspricht einem Feld vom Typ „Complex“, der in die HashMap vom Typ **AudioData** abgelegt wird. Die komplexen Kurzzeitspektren werden mit einem Bandpassfilter gewichtet (siehe Abschnitt 5.3).
Zusammen mit den fouriertransformierten Signalen kann im Anschluss die Berechnung der Zeitverzögerung nach Gleichung (39) und (41) erfolgen. Die Rückgabe ist mit der **TauData**-HashMap sichergestellt.

Der Teilschritt **(V)** in der Abfolge beschäftigt sich mit der Positionsbestimmung unter Zuhilfenahme der geschätzten relativen Verzögerungszeiten. Die in Abschnitt 4.3 vorgestellte iterative, geschlossene Lösung zur Lokalisation wird an dieser Stelle realisiert. Für diesen Zweck ist die Klasse **Localizers** vom **DoAEstimator** über die Methode

```
1 public Matrix int refMicId, ConcurrentHashMap<String, TauData> tauHashMap, MicArray
      array) {...}
```

aufzurufen. Die Umsetzung erforderte eine Einbindung der *JAMA* (MathWorks®) API. Sie stellt alle Funktionalitäten bezüglich der vektoriellen Verarbeitung zur Verfügung, die im nativen Leistungsumfang von Java fehlen.
Im Anschluss an den Iterationsprozess enthält der Rückgabevektor den Zielpunkt für das jeweilige Teilmikrofonfeld.

Beide Vektoren werden im finalen Schritt **(VI)** zerlegt und zu einem neuen Ausgabevektor zusammengefügt (siehe Abschnitt 4.4). Die Plausibilitätsprüfungen übernimmt die Klasse **PlausibilityChecker** und kontrolliert die Vektoren auf Einhaltung der vorgegebenen Schrankenbestimmungen.

Schlussendlich wird der Zielvektor in einer Variable gespeichert, auf die andere Programmteile mit *getTargetSource()* zugreifen können. Die Visualisierung in Kapitel 6 nutzt diese Information gleichermaßen, um das dort dargestellte Messfeld zu aktualisieren.

5.3 Entwurf eines feststehenden Bandpassfilters

Zur Verbesserung der Richtungsschätzung innerhalb der Klasse **DelayAnalysis**, ist ein festehender Bandpassfilter vorgesehen. Aufgrund dessen musste ein phasenstabiler, komplexwertiger Filter entworfen werden. Die Restriktion der Stabilität in der Phase ergibt sich daraus, dass nur die Phaseninformationen innerhalb der GCC nach (39) von Bedeutung sind. Gleichwohl sollte dieser Filter nicht für alle Arten der GCC dienlich sein.
Durch die Beschränkung auf eine stabile Phase kam der Finite Impulse Response (FIR) Filter zum Einsatz. Ein vergleichbarer Infinite Impulse Response (IIR) Filter wäre durchaus stabil gewesen, würde jedoch im Gegenzug eine Nicht-Linearität im Phasengang aufweisen.

Die Berechnung der Filterkoeffizienten und das Design wurde mittels der numerisch-ausgelegten Programmierumgebung MATLAB® (MathWorks®) durchgeführt. Die Eingabeparameter für das gewünschte Filterdesign wurden wie folgt festgelegt:

Filterparameter

Abtastrate:	$44100\,\mathrm{Hz}$
Erste Sperrfrequenz:	$90\,\mathrm{Hz}$
Erste Durchlaßfrequenz:	$110\,\mathrm{Hz}$
Zweite Durchlaßfrequenz:	$3980\,\mathrm{Hz}$
Zweite Sperrfrequenz:	$4000\,\mathrm{Hz}$
Erste Sperrdämpfung:	$0{,}00031622776602$
Durchlaßbandwelligkeit:	$0{,}28012999961$
Zweite Sperrdämpfung:	$0{,}0001$
Dichtefaktor	20

Die untere Grenzfrequenz ist durch die tiefe männliche Stimme definiert. Diese liegt bei $125\,\mathrm{Hz}$ für den Grundton und wäre infolgedessen die niedrigste wahrzunehmende Frequenz. Die obere Grenzfrequenz von $4000\,\mathrm{Hz}$ erschließt sich aus den Abständen der Mikrofone mit der Gleichung (5) zusammen mit dem kleinsten ermittelten Mikrofonabstand von $d_{min} = 0{,}04\,\mathrm{m}$ (für die Mikrofone m_5 und m_6).

Die Generierung des Filterdesigns erfolgte über interne Funktion „*FIRPM*“ in MATLAB® („Equiripple Filter“). Für die oben definierten Parameter ergibt sich ein phasenstabiler Filter (siehe Abb. 12 und Abb. 13).

Durch die feste Fensterbreite von $N = 1024$ Abtastwerte in der Verarbeitung kann eine entsprechende Einschränkung der diskreten Koeffizienten getroffen werden. Dies erfolgt durch die *freqz()*-Funktion. Die Trennung von Real- und Imaginärteil sorgen für Java-kompatible Feldstrukturen.

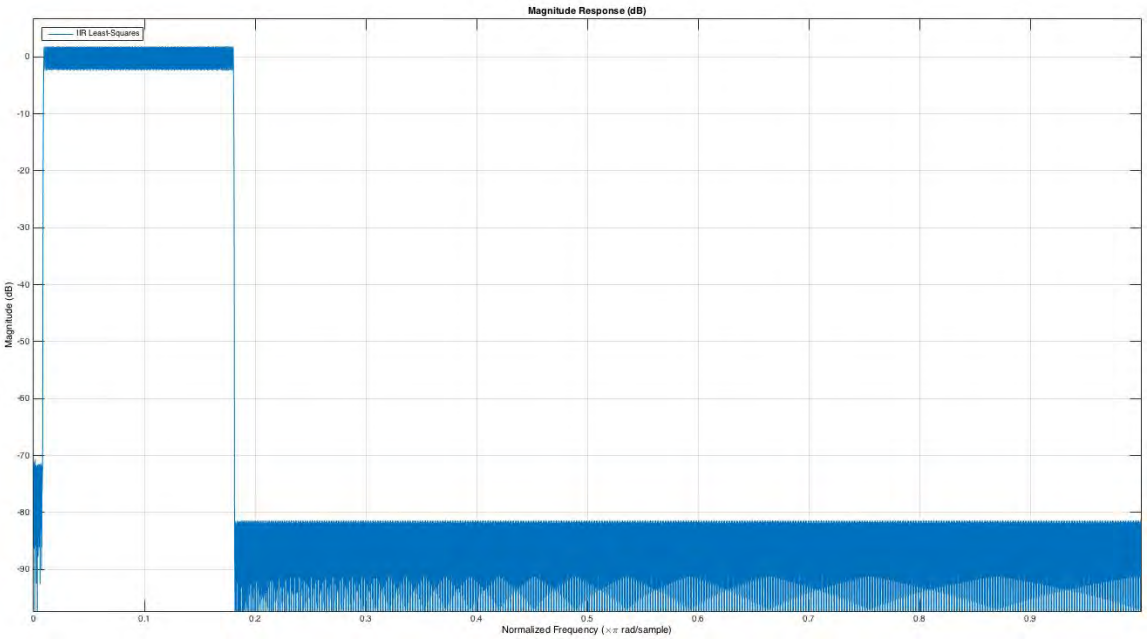


Abbildung 12: Normalisierter Amplitudengang des FIR-Bandpassfilters

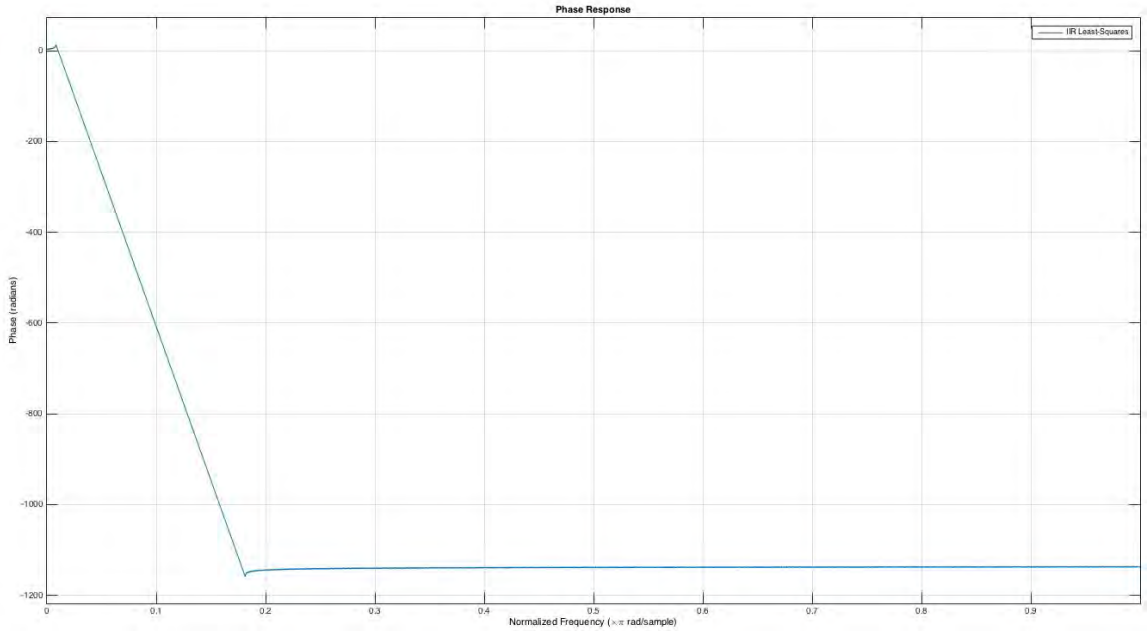


Abbildung 13: Normalisierter Phasengang des FIR-Bandpassfilters

Die Verwendung des Bandpassfilters über die statische Klasse **StaticBandpassFIR** wird zur Laufzeit referenziert. Der feststehende Filter findet nachfolgend für die jeweiligen Kurzzeitspektren Anwendung. Zur Vollständigkeit befindet sich im unteren Teil der Klasse **StaticBandpassFIR** der entsprechende MATLAB®-Code in einer auskommentierten Zeichenkette.

5.4 GCC-PHAT Testumgebung

Zur besseren Evaluierung des eingesetzten GCC-Vorfilters wurde eine entsprechende Testklasse implementiert (siehe UML-Klassendiagramm Abb. 29, Anhang A.4). Die Klasse **MicrophoneGCCTester** soll dazu dienen, Fehler zu entdecken und die Laufzeitschätzung hinsichtlich des SNR-Verhaltens zu optimieren. Das Testprogramm wurde als autonome Software entwickelt und bedient sich der im Abschnitt 5.2 vorgestellten Teilimplementierungen aus der Klasse **DelayAnalysis**. Wie bereits in Kapitel 5.1 vorgestellt, erfolgt die Zulieferung mit Audiodaten für die Testumgebung über **HammerfallAudioDevice**. Im Gegensatz zu der Realisierung von **DoAEstimator** ist das Programm keine Echtzeitanwendung. Zur Prüfung des Algorithmus aus der Gleichung (41) wird ein unveränderliches Audiosignal erzeugt. Der Nutzer der Testumgebung muss dafür einen Lautsprecher im Raum positionieren und diesen mit der entsprechenden Audiokarte des Industry Personal Computers (IPCs) verbinden (Regelung des Ausgabekanals über das Betriebssystem). Der Standpunkt ist abhängig von den zu untersuchenden kohärenten bzw. inkohärenten Bedingungen für den Schalleinfall. Zur Ausführung der Testklasse wird die *main()*-Methode von **MicrophoneGCCTester** gestartet.

Das Testsignal (hier Bandpassrauschen) wird über eine Zeit von $30\,\mathrm{s}$ ausgegeben. Die Handhabung von Laufzeitkomponenten sorgt für eine Verzögerung, die lediglich ein Zeitintervall von $14\,\mathrm{s}$ zulässt. Das bietet jedoch eine ausreichend große Stichprobe. Alle in dem Intervall gemessenen Daten werden für die Zeit der Wiedergabe in einer Warteschlage gespeichert. Eine Analyse und Auswertung ergibt sich erst nach Abschluss der Signalwiedergabe.

Die Analyse berechnet für beide Mikrofonfelder die geschätzten $\hat{\tau}_{1i}$-Werte in der Generalized Cross-Correlation (GCC). Die Korrelationsanalyse findet für alle gespeicherten Audiorahmen statt und wird darüber hinaus in einem Feld gespeichert. Für alle gesammelten Daten erfolgt anschließend eine Varianzanalyse, die darüber Aufschluss gibt, ob die verwendeten GCC-Methode ausreichend robust für die jeweiligen Mikrofonpaare ist. Die Ergebnisse werden zur Betrachtung auf der Textkonsole ausgegeben. Ein geringer Varianzwert entspricht einer günstigen Eigenschaft zur Lokalisation der Schallquelle. Analog dazu ist für den Fall von hohen Varianzen eine Lokalisation ausgeschlossen. Alle ermittelten Daten speichert das Testprogramm auf der Festplatte zur weiteren (tabellarischen) Analyse.

6 Visualisierung

Eine weitere Aufgabenstellung innerhalb der vorliegenden Arbeit ist die Entwicklung einer geeigneten Bedienoberfläche für das CSL-Projekt. Diese grafische Benutzerschnittstelle (engl. Graphical User Interface (GUI)) sollte alle Komponenten aus früheren Arbeiten enthalten und darüber hinaus Steuerungsmöglichkeiten sowie Veranschaulichungen für das Beamforming bieten. Bereits in der Arbeit [Bir13] hat der Autor eine 3D-Anwendung, beruhend auf der *Java3D*-Klassenbibliothek entwickelt, die der Visualisierung der Zielkoordinaten des räumlichen Suchalgorithmus dient. Dieser Quader repräsentierte eine dreidimensionale Darstellung des CSLs mit den darin enthalten Mikrofonfeldern (siehe Abschnitt 6.4). Im Konferenzbeitrag [Bir14] wird ein zweidimensionaler Querschnitt der Sensitivitätsverteilung aus Abbildung 1 vorgestellt. Mit dem Hintergrund der zwei und den dreidimensionalen Darstellungen ist ein erster Entwurf für das GUI entstanden.

6.1 Entwurf einer grafischen Benutzeroberfläche

Der Entwurf des GUI-Designs sollte sich in die bisherige Umgebung des CSL-Softwareprojektes mit eingliedern. Alle neu entstehenden Darstellungen, Schaltflächen und sonstige Interaktionselemente müssen den Designkriterien des Library Computer Access and Retrieval System (LCARS) gerecht werden. Das LCARS ist eine fiktive Benutzerschnittstelle für die Steuerkonsolen der Raumschiffe aus der U.S.-Serie *Star Trek*™. Diese Zukunftsvision einer GUI war im Jahr 1987 in „Star Trek™: The Next Generation“ erstmals zu sehen (Abb. 14).

Abbildung 14: LCARS-basierte Benutzerschnittstelle in „Star Trek™: The Next Generation“-TV-Serie, entwickelt von Michael Okuda[1]

Der eigene Entwurf dient vorrangig zur Orientierungshilfe für die Anordnung der einzelnen Objekte. Farbgebungen, Implementierungskomponenten, weitere Funktionalitäten oder exakte LCARS-Designdetails standen nicht im Fokus. Ein gewisser Funktionsumfang wurde beim Konzept mit berücksichtigt. Es galt in erster Linie die vorgegebene Auflösung von 1920×1080 Bildpunkte optimal räumlich

[1] Bildquelle: `https://chasingatlantis.wordpress.com/2013/12/09/michael-okuda-star-trek-graphic-art-designer-north-america-tour-2-finale/`

auszunutzen. Es entstand ein erster Grobentwurf des Steuerpults (Abb. 15). Die Realisierung des fertigen Panels (dt. Tafel) für das Beamforming wird in Abschnitt 6.2 genauer erläutert.

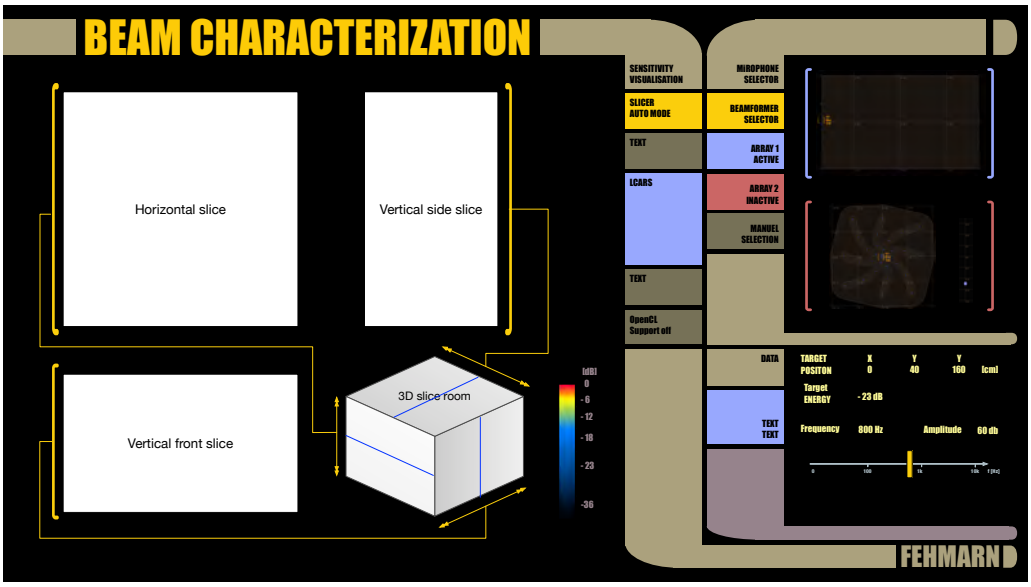


Abbildung 15: Grobentwurf LCARS-Konsole für das Beamforming

Im Konzept 15 werden alle Platzhalter für die bereits angesprochenen Elemente ersichtlich, darunter die Querschnittsflächen des Raummodelles und die dreidimensionale Darstellung des Raumkoordinatensystems. Zusätzlich wurde in das 3D-Modul die Ebenen eingebracht, die später als räumliche Orientierung der Querschnitte dienen sollten. Gekennzeichnet sind sie durch Schieberegler, die zur Auswahl der Ebene dienen. Darüber hinaus ist auf der rechten Seite eine Visualisierung der beiden zweidimensionalen Mikrofonfelder eingefügt. Diese bieten dem Benutzer die Möglichkeit, sich über den Status der einzelnen Mikrofone zu informieren. Der übrige Raum in der unteren rechten Ecke steht für zusätzliche Informationen zur Verfügung. Eine Veränderung der frequenzabhängigen Raumquerschnitte ist über den vorgesehenen Schieberegler denkbar.

6.2 Realisierung einer Benutzerschnittstelle auf der Basis von LCARS

Im vorangegangenen Kapitel 6.1 ist der Grobentwurf eines möglichen LCARS-Panels präsentiert worden. Für die Realisierung sollte keine Benutzeroberfläche aus den Standardbibliotheken von Java verwendet werden. Zum Einsatz kam das GUI LCARS Widget Toolkit (LCARSWT) nach [Wol15b]. Diese auf rein Java-basierende API dient dazu, die grafische Benutzeroberfläche einer Anwendung dem in *Star Trek*™-Serien gezeigten Schaltpulten nachzuempfinden.

Die Implementierung erfolgte in der Klasse **BeamformerPanel**, die namensgebend für die Benutzeroberfläche ist. Das Panel verfügt über verschiedenartige Elemente, die jeweils eine spezifische x- und y-Koordinate auf dem Bildschirm besitzen. Zur Orientierung und Positionsfestlegung dienen die Koordinaten aus dem Entwurf 15. Die Definition und Festlegung der Objekte findet in der überschriebenen Methode *init()* statt. Es existieren eine Reihe von Standardobjekten, die in der eigenen Implementierung Verwendung finden (vgl. [Wol15a]):

- **ERect**: ein etikettiertes Rechteck,
- **EValue**: eine Zahlenwertanzeige mit zusätzlichem Etikett,
- **ELabel**: ein Text,
- **EElbo**: ein ellbogenförmiges etikettiertes Objekt,
- **ESector**: ein sektorförmiges etikettiertes Objekt,
- **EPositioner**: dynamischer Schieberegler und
- **EImage**: ein Bitmap-Bild.

Aus der Auflistung ist ersichtlich, dass für die eigene Realisierung weitere Objektklassen implementiert bzw. adaptiert werden mussten. Dies galt insbesondere für die modifizierbaren zwei- und dreidimensionalen Darstellungen, die auf dem Panel eingebettet wurden. Es existierte durchaus eine Möglichkeit, Grafiken einzubinden, jedoch nur für statische, externe Bilddateien, die nicht veränderbar sind (**EImage**). Aufgrund dessen entstanden die Klassen **EBufferedImage** und **ESensitivityPlot**. Ersteres entspricht einem unveränderlichen Objekt mit Erzeugung zur Laufzeit. Der verwendete Name schließt auf eine Abstract Window Toolkit (AWT) Anwendung, die durch Transformation in ein Standard Widget Toolkit (SWT) Objekt überführt wird. Verwendung findet diese Art der Repräsentation für die Farbskalen zur Evaluation der Sensitivitätsverteilungen (siehe Abb. 17). Die zweite neu geschaffene grafische Klasse **ESensitivityPlot** kommt bei der eigentlichen Darstellung der Sensitivitätsverteilung zum Tragen. Die vorliegende Bildklasse dient als Schnittstelle zwischen der bildgebenden Implementierung aus Kapitel 6.3.3 und dem erläuterten bilddarstellenden Panel.

Ein weiteres Beispiel ist der Schieberegler, welcher die gewählte Frequenz der Sensitivitätsverteilungen darstellt. Mit der Klasse **EPositioner** existierte bereits eine solche Lösung. Sie besitzt jedoch keine gehörrichtige, logarithmische Skaleneinteilung. Infolgedessen findet eine Adaption des linear arbeitenden **EPositioner** in die logarithmische Repräsentation durch die Klasse **EPositionerFreq** statt.

Die Schwierigkeit in der entsprechenden Implementierung lag in der Abbildung von einem linearen in ein logarithmisches Regelverhalten.

Die Benutzeroberfläche besitzt mit der dreidimensionalen Raumdarstellung zusätzliches interaktives Objekt. Zur konkreten Realisierung wird auf das Kapitel 6.4 verwiesen. Zu erwähnen ist, dass eine neue Schnittstellenklasse geschaffen wurde. Mit der **ECanvas** ist es möglich, eine dreidimensionale Darstellung, Canvas (dt. Leinwand) genannt, in das Panel einzubinden.

Genaue Betrachtungen von **BeamformerPanel** zeigen weitere öffentliche Methoden (siehe UML-Klassendiagramm Abb. 31, Anhang A.4). Es handelt sich um eine klasseninterne Implementierung des Entwurfsmusters „Observer“. Es sei erwähnt, dass die interne Realisierung mangels der Möglichkeit einer multiplen Klassenerweiterung direkt eingebettet wurde. Das „Observer“-Entwurfsmuster dient an dieser Stelle zur asynchronen Kommunikation zwischen Panel und weiteren Objekten. Das gestattet eine gleichzeitige Veränderung der Grafiken und Neupositionierung der Raumquerschnitte in der dreidimensionalen Darstellung. Als Kommunikationseinheit dienen verschiedenartige Nachrichtenobjekte, die sich von **IBeamformerMsg** ableiten.

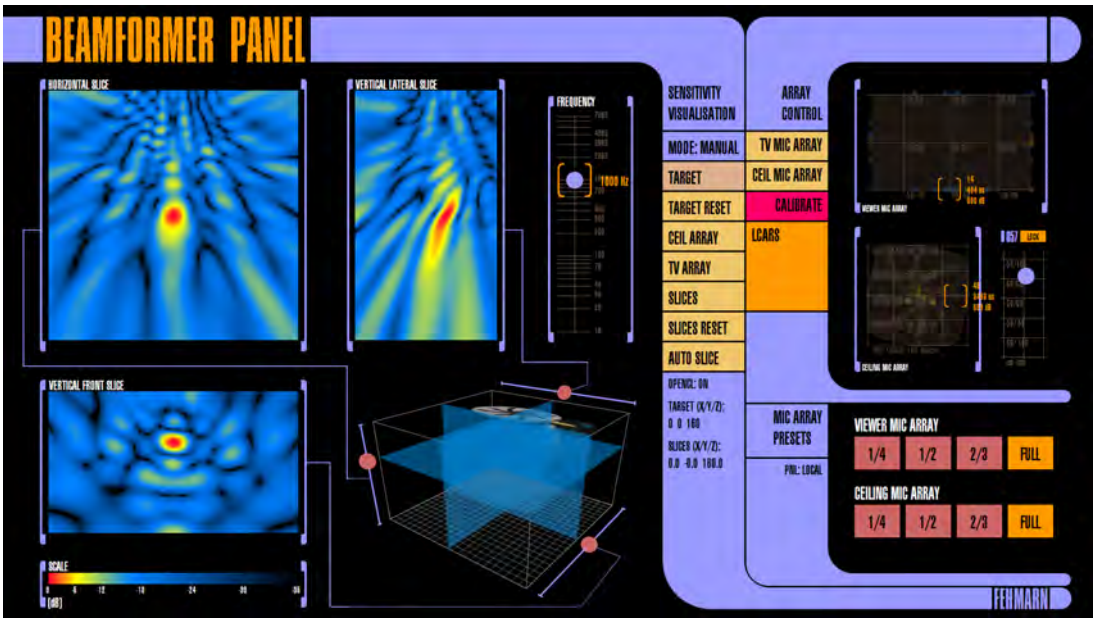


Abbildung 16: Bildschirmkopie vom LCARS **BeamformerPanel**

Das fertiggestellte Panel (Abb. 16) besitzt drei optische Unterteilungen. Auf der linken Seite sind die unterschiedlichen Ansichten der Raumquerschnitte und deren Lage im Raummodell ersichtlich. Darüber hinaus kann durch den logarithmischen Schieberegler auf die gewählte Frequenz Einfluss genommen werden. Die Regler an der Raumdarstellung sind für die Wahl repräsentierten Querschnittsfläche dienlich. Durch Betätigen eines der Sensitivitätsverteilungsplots ist es möglich manuell Zielpunkt des Beamformers zu verändern.

In der Mitte der Szene befindet sich ein zweigeteiltes Auswahlmenü. Die linke Seite beinhaltet entsprechende Befehle zur Modifikation der dreidimensionalen Raumdarstellung. Es bietet die Möglichkeit des Ein- oder Ausblendens von Objekten (*„TARGET“*, *„TV ARRAY“*, *„CEIL ARRAY“*). Für das Zurücksetzen des gewählten Zielpunktes ist die Schaltfläche *„TARGET RESET“* dienlich. Analog dazu steht ein Schalter zum Zurücksetzen der Raumquerschnitte mit *„SLICE RESET“* zur Verfügung. Durch Betätigen von *„SLICE“* wird eine weitere Funktion freigeschaltet: Manuelle Rotation der dreidimensionalen Raumdarstellung. Benutzer können folglich händisch Einfluss auf die Orientierung der Illustration nehmen. Die oberste Schaltfläche mit dem nicht selektierten Status *„MODE: MANUAL“* repräsentiert den verwendeten Modus zur Nachführung und Steuerung des Beamformers. Durch Aktivierung der Schaltfläche wird der Modus *„AUTO“* ausgelöst, der eine automatische Lokalisation des Sprechers vornimmt. Parallel wird die manuelle Auswahl in den Raumquerschnittsflächen deaktiviert.
Die rechte Menüseite bietet weniger interaktive Elemente. Durch Auswahl von *„TV MIC ARRAY“* oder *„CEIL MIC ARRAY“* steht dem Benutzer eine Deaktivierung der beiden Teilmikrofonfelder zur Verfügung. Darüber hinaus kann mittels *„CALIBRATE“* eine manuelle Kalibrierung der Mikrofonamplituden vorgenommen werden. Die letzte eingebundene Schaltfläche ist das *„LCARS“*, über die das Hauptmenü des CSL-Softwareprojektes erreichbar ist.

Rechtsseitig auf dem Panel befinden sich zwei zusätzliche Visalisierungs- und Steuereinheiten. Im oberen Teil sind beide Teilfelder mit den entsprechenden Mikrofonen dargestellt, die über Leuchtpunkte deren Aktivität widerspiegeln. Ein zusätzlicher Regler dient der variablen Positionsveränderung des deckenseitigen Mikrofonfeldes. Im unteren Teil sind Schaltflächen platziert, die es ermöglichen eine Auswahl von Mikrofonen im jeweiligen Teilmikrofonfeld zu deaktivieren.

Abschließend sei zur Gesamtvisualisierung erwähnt, dass die überwiegenden Aktualisierungen über periodisch aufgerufene Funktionen gesteuert werden. Die entsprechenden Methoden *fps1()*, *fps10()* und *fps25()* sind hierfür vom **MainPanel** abgeleitet.

6.3 Generierung der Raumquerschnitte zur Sensitivitätsverteilung

Für die Raumquerschnitte, folgend *„Slices“* (dt. Scheiben) genannt, wurde eine bereits bestehende Implementierung verwendet (vgl. [Bir14]). Die Raumquerschnitte der Sensitivitätsverteilung bezeichnen eine zweidimensionale Visualisierung der orts- und frequenzabhängigen Dämpfung eines Beamformers (siehe Abb. 17). Die Darstellung der Werte ist infolgedessen logarithmisch und erfolgt unter Angabe einer beigestellten dB-Skala.

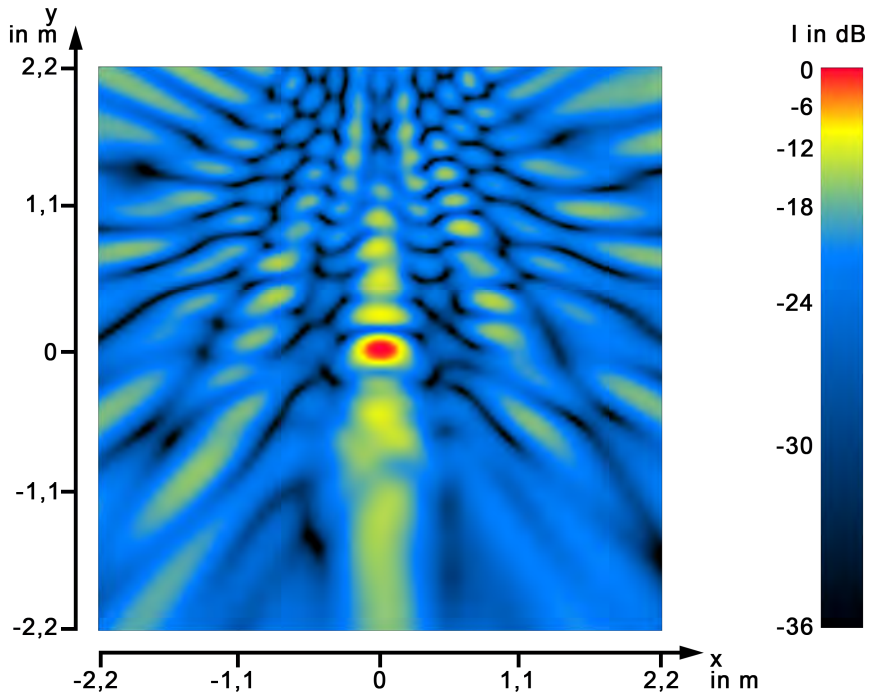


Abbildung 17: Raumquerschnitte der Sensitivitätsverteilung ($f = 1000\,\mathrm{Hz}$, $N = 64$ Mikrofone, Fokus auf Messfeldmitte bei $1,6\,\mathrm{m}$ Höhe, Deckenmikrofonfeld über dem Nullpunkt)

Die existierende Implementierung des verwendeten Algorithmus wies äußerste Defizite in der Laufzeit auf. Im Mittel von 100 Messungen ergab die Ausführung $2{,}2\,\mathrm{s}$ für die Anwendung (siehe Tabelle 3). Diese erste Revision war in ein reines Java-Programm auf einem Industrie-PC (kurz IPC) (Konfigurations- und Laufzeitumgebung im Anhang A.3). Der ermittelte Leistungswert ist für eine Echtzeitanwendung inakzeptabel und wurde bereits in einer früheren Version durch eine Realisierung auf der Basis von *OpenCL* verbessert.

6.3.1 OpenCL als Algorithmenbeschleuniger

Grafikkarten sind für parallelisierbare Berechnungen konzipiert worden. Für diesen Zweck stellen sie den General Purpose GPU (GPGPU) als arithmetisch-logische Einheit für allgemeine Berechnungen zur Verfügung. Eine Reihe von Programmierschnittstellen bedienen sich einer derartigen Technik und bieten folglich erhebliche Potentiale in der Leistungsverbesserung. Eine universell einsetzbare Schnittstelle ist die C-basierte API *OpenCL*. Sie bietet die Chance, nicht-grafische Programme auf dem GPGPU auszuführen.

6.3.2 OpenCL mittels JavaCL

Durch die Restriktion der Verwendung von rein Java-basierten Schnittstellen galt es eine Verpackung zu finden, mit der es möglich ist, *OpenCL* unter Java einzusetzen. Die von [Cha15] bereitgestellte API *JavaCL* entsprach dieser Bedingung. Sie stellt alle nativen OpenCL-Bibliotheksfunktionen zur Verfügung (vgl. [MGM+12], [Khr14]).
Die Arbeit von [Dom12] verdeutlichte, dass *JavaCL* gegenüber dem konkurrierenden APIs *JOGAMP* und *JOCL* nicht nur erhebliche Vorteile in der Komplexität des Aufbaus bietet, sondern bessere Dokumentationen und Entwicklungszeiten zulässt. Das bedeutendste Argument ist im Vergleich von den Verarbeitungstechniken die Laufzeit. So überzeugt *JavaCL* mit $42\,\mathrm{ms}$ gegenüber dem *JOGAMP* $(46\,\mathrm{ms})$, *JOCL* $(97\,\mathrm{ms})$ und dem CPU $(4970\,\mathrm{ms})$ bei einer Vektormultiplikation von der Länge 10000000.

6.3.3 JavaCL Implementierung

Eine erste Implementierung zur Generierung von Plots der Sensitivitätsverteilung durch *JavaCL* wurde bereits frühzeitig in das CSL-Softwareprojekt eingebunden. Eine kontinuierliche Verwendung war mangelst Schnittstellen und Programmstruktur nicht möglich. Ein ausführliches Refactoring sollte diesen Umstand ausgleichen (siehe UML-Klassendiagramm Abb. 30, Anhang A.4).

Die Java-Klasse **SensitivityPlot** stellt für diesen Zweck die Hauptklasse dar, worüber die Plotgenerierung ausgeführt wird. Diese Klasse besitzt keinen öffentlichen Konstruktor und wird über „Singleton"-Entwurfsmuster instanziiert. Es existiert neben der Standardmethode „getInstance()" eine zweite Methode „getInstance(CLState)". Sie ermöglicht eine Steuerung der Generierungsart, um eine nicht vorhandene GPGPU-Schnittstellen zu kompensieren. In der Aufzählung **CLState** sind drei verschiedene Variationen definiert (siehe Abb. 30, Anhang A.4). Die in dem Zusammenhang stehende Initialisierung der Klasse nimmt nachfolgend eine Überprüfung des gewählten Statusindikators vor, um sicher zu stellen, dass die Ausführungsplattform die gewünschte Mindestanforderung erfüllt. Sollte seitens der verwendeten Computer keine OpenCL-Unterstützung existieren, wird automatisch die CPU-basierte Implementierung verwendet (`NO_CL`). Als Konsequenz hat das erhebliche Einbußen auf die Leistung der Benutzeroberfläche.

Über `NO_CL` hinaus sind noch zwei weitere Statusindikatoren definiert: `CL_BUFFER` und `CL_IMAGE_2D`. Für beide Varianten gelten unterschiedliche Benutzerschnittstellen und Methoden, um auf generische Typen verzichten zu können.

Die Umsetzung von `CL_BUFFER` ist eine Weiterentwicklung der bereits vorhandenen Plotimplementierung und liefert dem Benutzer über die Methoden

```
1 public int[] getHorizontalSensitivityIntArray(MicArrayState state, Point3d slicePos,
      float frequency, int width, int height){...}
2
3 public int[] getVerticalFrontSliceIntArray(MicArrayState state, Point3d slicePos,
     float frequency, int width, int height){...}
4
5 public int[] getVerticalLateralSliceIntArray(MicArrayState state, Point3d slicePos,
     float frequency, int width, int height){...}
```

jeweils einen horizontalen, vertikal-frontseitigen und vertikal-seitlichen zweidimensionalen Sensitivitätsverteilungsplot. Die Daten sind in Feldern vom Typ Integer mit 32-Bit und einem BGRA-Farbschemata kodiert. Die gewählte Farbkodierung ist konform mit der aktuell verwendeten grafischen Programmbibliothek SWT. Die SWT-Bibliothek setzt in diesem Zusammenhang auf „Image“-Objekte anstatt der in AWT bisher verwendeten „BufferedImage“-Instanzen.

Der zweite Statusindikator *CL_IMAGE_2D* gilt für die ARGB-Farbkodierung (AWT Umgebungen). Analog zum Quellcodeauszug 6.3.3 bietet es mit

```
1 public BufferedImage getHorizontalSliceImage(MicArrayState state, Point3d slicePos,
      float frequency, int width, int height){...}
2
3 public BufferedImage getVerticalFrontSliceImage(MicArrayState state, Point3d
      slicePos, float frequency, int width, int height){...}
4
5 public BufferedImage getVerticalLateralSliceImage(MicArrayState state, Point3d
      slicePos, float frequency, int width, int height){...}
```

drei Methoden zum Erstellen der jeweiligen Plots. Erarbeitet wurde dieses Konzept auf der Suche nach den maximalen Möglichkeiten einer *OpenCL*-Implementierung. Präzisiert ausgedrückt sorgten genauere Betrachtungen der gerätespezifischen *OpenCL*-Grafikkartenspezifikationen im Programmcodeabschnitt 3 dafür, dass die eingesetzte Grafikkarte (*NVIDIA GeForce GTX 660*) die Unterstützung von direkten Bildberechnungen über *OpenCL* gewährleistet (Lst. 3 Anhang A.3, Zeile 39). Alle Berechnungen können so unmittelbar durch den GPGPU durchgeführt werden. Die bisherige Verwendung der OpenCL-Technik im AWT-Bereich zielte darauf hin, die Basisdaten von 440×440 Bildpunkte zu liefern. Das beinhaltete für jeden Bildpunkt eine Gleitkommazahl (dB-Wert), die im weiteren Verlauf vom Prozessor auf eine Farbe übertragen werden musste. Erst nach der Transformation entstand das fertige Bild. Durch die *Image2D*-Unterstützung wurde auf eine Umrechnung verzichtet. Umsetzung und Implementierung dieser Technik zeigten weitere Optimierungen hinsichtlich der Laufzeit. Die veraltete auf AWT basierende Implementierung von $40{,}3\,\mathrm{ms}$ wurde auf $33{,}78\,\mathrm{ms}$ verbessert (siehe Tabelle 3). Der Einsatz von SWT brachte eine erneute Optimierung auf $28{,}79\,\mathrm{ms}$.

Java (CPU exklusiv)	JavaCL Buffer (AWT)	JavaCL Image2D	JavaCL Buffer (SWT)
$\varnothing t$ in ms			
2216,79	40,30	33,87	28,79

Tabelle 3: Durchschnittliche Zeit von 100 Messungen zur Generierung eines Raumquerschnittes der Sensitivitätsverteilung mit 440×440 Bildpunkte (siehe Abb. 17)

Die Initialisierung des Kernels wird über die Klasse **SensitivityKernels** realisiert, die beide Implementierungsvarianten von *OpenCL* einschließt. Der Quellcode für den Kernel befindet sich in der dazugehörigen Datei „SensitivityKernels.cl“. Der darin befindliche Text ist reiner C-Code und wird einmalig in Form einer Zeichenkette in *JavaCL* geladen. Die Nutzbarmachung des Kernels findet durch Instanzen

von der **SensitivityEntity** statt. Das schließt die abstrakten Klasse **ASensitivityEntity** aller Methoden zum Erstellen der Grafiken mit ein. Jede Ausführung beinhaltet dabei eine Aktualisierung der für die Generierung wichtigen Parameter im Kernel. Wichtig ist zu beachten, dass für jede Ploterzeugung eine eigenständige Instanz von **SensitivityEntity** erstellt werden muss. Das ist der Tatsache geschuldet, dass die Implementierung von *JavaCL* für jede Ausführung des Kernels ein eigenständiges C-Objekt erstellt. Eine Löschung der Instanz kann nur erfolgen, wenn das korrespondiere Objekt in der *JavaCL*-Umgebung zerstört wird (siehe [Cha14], „Garbage Collection vs. Manual Release“). In Java übernimmt die automatisierte Handhabung zur Speicherbereinigung von Objektinstanzen der „Garbage Collector“. Eine exklusive Zerstörung ist nicht möglich. Wird dies nicht beachtet, kommt es zu Speicherüberläufen und damit in Verbindung stehende Abstürze der Java Virtual Machine (JVM).

Die Verarbeitung der Klasse **SensitivityEntity** wird über die interne Methoden in **ASensitivityEntity** realisiert. Innerhalb dieser befindet sich die Aktualisierung des Kernelparameters sowie die Erzeugung des eigentlichen Plots. Die jeweils betreffenden Kernel-Objekte werden zur Ausführung statisch von der Klasse **SensitivityKernels** bereitgestellt, um eine erneute Instanziierung zu vermeiden.

Über die bereits vorgestellten öffentlichen Schnittstellen hinaus, bietet die **SensitivityPlot**-Klassen einige weitere Methoden, die kurz Erwähnung finden. Die aktuelle Position der drei Raumquerschnitte ist in der Objektinstanz von **SensivitiyPlot** gespeichert und kann über *getSlicePos()* ausgelesen werden. Weiterhin geben *getCLState()*, sowie *isOpenCLActive()* darüber Auskunft, welche bzw. ob eine Verwendung von *OpenCL* vorliegt. Zur Evaluierung der kodierten Farben eines Plots kann über die Methode *getVerticalScaleImage()* ein vertikale Farbskala generiert werden. Analog dazu existiert für eine horizontale Darstellung *getHorizontalScaleImage()*.

6.4 Dreidimensionale Visualisierung

Die Auswahl Querschnitte für die Sensitivitätsverteilung sollte einfach gestaltet sein. Hierzu wurde eine Steuerung gewählt, die sich einer existierenden Java3D™-Realisierung bediente, in der eine dreidimensionale Visualisierung des Experimentierfeldes zu sehen ist (Abb. 18). Eine Einbettung in das **BeamformerPanel** war bisher nicht möglich, begründet durch das Fehlen entsprechender Kommunikationsschnittstellen. Das bereits vorgestellte und implementierte Entwurfsmuster des *Observers* bietet diese Funktionalität. Die Versorgung der Grafikeinheit mit den entsprechenden Nachrichten aus den abgeleiteten Instanzen der **IBeamformerMsg** erfolgt über die Methode:

```
1 public void update(IObservable o, Object arg){...}
```

Eine Erweiterung des **Cube3d** wurde hinsichtlich der Komponenten der drei Raumquerschnitte vorgenommen: Die horizontale „Z-Slice“ und die beiden vertikalen „X-“ und „Y-Slices“. Ein entsprechender grafischer Schieberegler neben jeder der zweidimensionalen Abbildungen innerhalb des **BeamformerPanel** wäre für den Benutzer nicht intuitiv genug. Das begünstigte die Entscheidung der Einbettung in die dreidimensionale Darstellung (Abb. 18). Die konkrete Position der Slices lässt sich über die öffentliche Methode *getSlicePosition()* ermitteln.

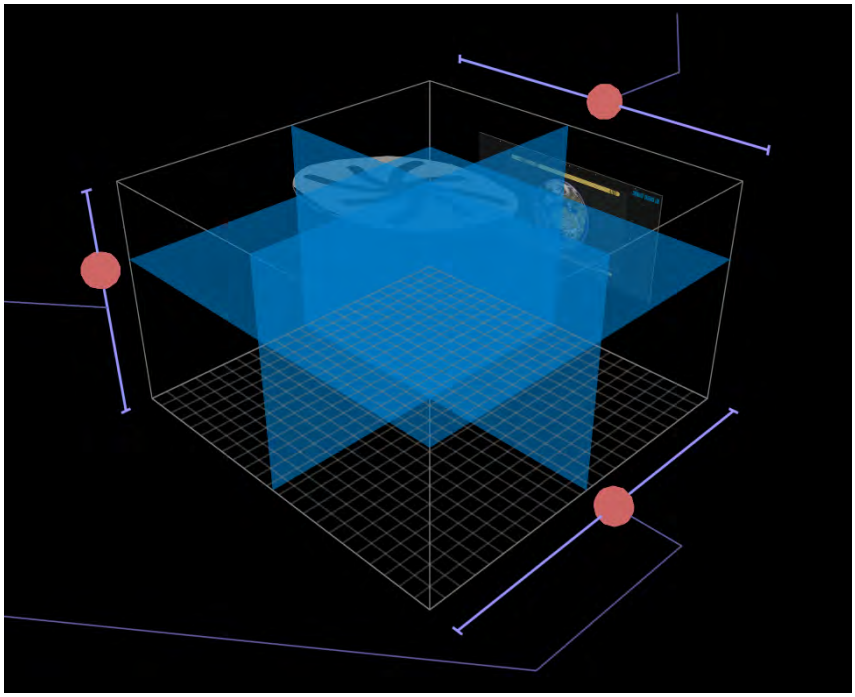


Abbildung 18: Dreidimensionale Messfeldvisualisierung

Ein Einschluss der Schieberegler in der virtuellen Umgebung hat eine explizite Behandlung von Mausbewegungen und entsprechenden Translationen zur Folge. Üblicherweise erfolgt die Rotation in *Java3D*™ mit der linken und die Translation mit der rechten Maustaste. Für den Einsatz eines Touchscreens steht ausschließlich die linke Taste zur Verfügung. Dies hatte zur Folge, dass die nativen **MouseEventListener** nicht erweitert werden konnten. Für den gewünschte Handhabung wurden die originale Implementierung weitgehend umprogrammiert (Abb. 33, Anhang A.4).

Im Programm der Klassen **MouseTranslateNew** wird mit Codeauszug 6.4 auf Maustastendruck und -ereignis geprüft. Das Ausrufezeichen in dieser entsprechenden Zeile kennzeichnet eine Überprüfung auf die linke Maustaste.

```
1 if ((id == MouseEvent.MOUSE_DRAGGED) && !evt.isAltDown()
```

Darüber hinaus sind Eingaben mit der Maustaste für eine lokale zweidimensionale Position auf dem Bildschirm definiert. Diese müssen für eine entsprechende räumliche Darstellung folgend in die virtuellen, dreidimensionalen Koordinaten überführt werden. Das geschieht unter Berücksichtigung der Mausbewegungsrichtung und der aktuellen Transformation der virtuellen Welt in der Methode *doProcess(MouseEvent evt)* von **MouseTranslateNew**.

Die Visualisierung besitzt weitere öffentliche Verbindungen. Sie ermöglichen eine Ausgabe eins bewegliches Canvas3d-Objektes durch die Methode:

```
1 public Canvas3D getCanvas3d(Vector3f viewPoint, float angle, boolean movable)
```

Weiterhin ist eine Änderung des Status des angezeigte Objektes durch *setVisible(boolean)* verfügbar. Die korrespondierende Abfrage wird von *isVisible()* zur Verfügung gestellt. Für die allgemeinere Verwenbarkeit, Wartbarkeit und Codeübersicht von Objekten wurden Deklaration vieler Objekte von der Klasse **Cube3d** in die abstrakte Vaterklasse **Cube3dObject** ausgelagert. Mit dieser Implementierung wurde ein ganzheitlicher Ansatz zur Lösung des Problem der räumlichen Darstellung von den Raumquerschnitten geschaffen.

7 Zusammenfassung und Ausblick

Mit der vorliegenden Arbeit präsentiert der Autor ein umfangreiches System zur Signalverarbeitung durch Mikrofonfelder.
Über die einleitenden theoretischen Betrachtungen hinaus erfolgte eine ausführliche Analyse von möglichen Fehlerquellen, die Einfluss auf eine robuste Sprecherlokalisierung haben könnten. Nachfolgend wurde auf die Beseitigung der Fehler eingegangen, um eine Steigerung der Robustheit der Mikrofonfeldverarbeitung zu erzielen. In diesem Zusammenhang wurde darüber hinaus eine neuartige Mikrofonfeldkalibrierung vorgestellt, die eine automatisierte Anpassung der Amplituden gestattet.

Im Hauptteil wird eine Lösung zur akustischen Laufzeitschätzung und Sprecherlokalisation für ein dreidimensionales Mikrofonfeld vorgestellt. Folglich konnte mit der zweistufigen Ortung ein Verfahren entwickelt werden, dass die rechnerisch aufwendige SRP-Methode ersetzt.

Im ersten Schritt der Laufzeitschätzung fand der eingesetzte PHAT-Filter eine besondere Beachtung. Die verwendete Gewichtung ist in der Literatur durchaus umstritten und wird durch unterschiedlichste Meinungen kommentiert (vgl. [KG13], [BAS97], [AM02], [SH14]). Grund dafür ist die Effizienz hinsichtlich des Nachhalls und des Ansprechverhaltens bei einem niedrigen SNR. Gleichwohl einer optimalen Voraussetzung durch Einsatz von Schallabsorber im Laborbereich stellte sich heraus, dass diese Eigenschaft ebenso negative Auswirkungen auf die implementierten Algorithmen hat. Die praktische Umsetzung zeigte, dass der verwendete Filter zu stark vom SNR abhängt. Es wurde rein subjektiv festgestellt, dass es zu erheblichen Schwankungen der Erkennungsrate der Schallankunftsrichtung kommt, wenn sich die Mikrofone im diffusen Schallfeld befinden. Dieser Fall tritt für eine Signalquelle unterhalb des Mikrofonfeldes ein, bei der eine Bündelung der Schallwellen in Richtung des Hauptbildschirmes erfolgt.
Unterschiedliche Testimplementierungen von Teilgewichtungen des modifizierten PHAT-Filters nach [AMP12] brachten keine Verbesserungen. Eine Frage, die weiterhin empirischer Untersuchungen bedarf, ist inwiefern sich dieses Filterverhalten durch die Erweiterung mit zusätzlichen Vorfiltern verbessern ließe. Eine Lösung für das Problem der Erkennerrate liefert zum Beispiel [LK10] mit einem „Wavelet“-Vorfilter. Um weitere Aussagen zur Verbesserung der Sprecherlokalisation treffen zu können, bedarf es weiterer Implementierungen und einer entsprechenden Simulation unter realen Bedingungen.
Der zweite Schritt zur Lokalisation mit einer geschlossenen Lösung nach [CHB05] zeigt, für den ungestörten Fall durchaus akzeptable Ergebnisse. Für die robuste Sprecherlokalisation in einem dreidimensionalen Mikrofonfeld stellt das eine solide Grundlage dar.

Die Realisierung umfasste alle Verarbeitungsschritte von der Signalerfassung, über die Weiterverarbeitung, bis hin zur Visualisierung und Ausgabe der Daten. Im Rahmen der Arbeit wurden 11.761 reine Codezeilen in 63 Klassen neu geschaffen oder bearbeitet. Die zusätzlichen Dokumentationszeilen sind in der Statistik nicht inbegriffen. Der überwiegende Teil des geschaffenen Programmcodes bietet für den Nutzer neue Funktionalitäten. Allerdings ist zu beachten, dass der geschaffene Funktionsumfang einiger Erweiterungen bedarf. Durch die Beschränkung in der Aufgabenstellung sind mögliche Ausbaustufen für spätere Abhandlungen in den Diskussionsteilen und dem analysierenden Abschnitt 3.1 angemerkt. Abgesehen von einem Erkenner für Sprachsegmente sind mit der Verarbeitungskette 3.2.2 die Verbindungen zu weiteren Verarbeitungseinheiten hinreichend dokumentiert. Die explizite Einarbeitung einer

Voice Activity Detection (VAD) wird bewusst offen gelassen, um ausreichend Spielraum für eine Erweiterung zu bieten.

Die mit der Aufgabenstellung geforderte Visualisierung besitzt über die algorithmische Realisierung hinaus einen weitreichenden Funktionsumfang. Die grafische Benutzerschnittstelle beinhaltet zahlreiche ansehnliche Darstellungen und Interaktionsmöglichkeiten für den Beamformer. Die Implementierung der Grafikelemente war nicht immer problemlos. Häufige Systemabstürze, verursacht durch die Generierung der Sensitivitätsverteilungsplots und der damit in Verbindung stehenden falschen Handhabung von JavaCL, konnten in dem Zusammenhang behoben werden.

Das vorgestellte und realisierte System stellt zusammenfassend die Grundlage für weitere Forschungsarbeiten im Rahmen des Cognitive Systems Lab dar. Insofern besitzen die neuen Verarbeitungs- und Visualisierungseinheiten zahlreiche Möglichkeiten, die ihr Potential erst in der zukünftigen Verwendung zeigen.

8 Quellenverzeichnis

[Aar03] AARABI, Parham: The Fusion of Distributed Microphone Arrays for Sound Localization. In: *EURASIP J. Appl. Signal Process.* 2003 (2003), Januar, 338–347. http://dx.doi.org/10.1155/S1110865703212014. – DOI 10.1155/S1110865703212014. – ISSN 1110–8657

[AM02] AARABI, Parham ; MAHDAVI, Albarz: The relation between speech segment selectivity and source localization accuracy. In: *ICASSP*, IEEE, 2002. – ISBN 0–7803–7402–9, 273-276

[AMP12] AGUILAR, Cosmellerena ; MOHINO, Inma ; PEREZ, Lorena A.: A comparative study of time-delay estimation techniques for convolutive speech mixtures. In: YENDURI, Sumanth (Hrsg.) ; WSEAS European Computing Conference (Veranst.): *Advances in Computer Science* WSEAS European Computing Conference, 2012, S. 291–296

[Aro92] ARONS, Barry: A Review of the Cocktail Party Effect. In: *Journal of the American Voice I/O Society* 12 (1992), 35–50. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.4.1283

[BAS95a] BRANDSTEIN, Michael S. ; ADCOCK, John E. ; SILVERMAN, Harvey F.: *Microphone-Array Localization Error Estimation with Application to Sensor Placement*. 1995

[BAS95b] BRANDSTEIN, Michael S. ; ADCOCK, John E. ; SILVERMAN, Harvey F.: A practical time-delay estimator for localizing speech sources with a microphone array. In: *Computer Speech & Language* 9 (1995), Nr. 2, 153 - 169. http://dx.doi.org/http://dx.doi.org/10.1006/csla.1995.0009. – DOI http://dx.doi.org/10.1006/csla.1995.0009. – ISSN 0885–2308

[BAS97] BRANDSTEIN, M.S. ; ADCOCK, J.E. ; SILVERMAN, H.F.: A closed-form location estimator for use with room environment microphone arrays. In: *Speech and Audio Processing, IEEE Transactions on* 5 (1997), Jan, Nr. 1, S. 45–50. http://dx.doi.org/10.1109/89.554268. – DOI 10.1109/89.554268. – ISSN 1063–6676

[Bau05] BAUMANN, Wolf: *Optimierung frequenzvarianter Nullbeamformer für akustische Signale mittels Statistik höherer Ordnung - Anwendungen im Kfz und in Büroräumen*, Technischen Universität Berlin, Doktorarbeit, 2005

[BHC08] BENESTY, Jacob ; HUANG, Yiteng ; CHEN, Jingdong: *Microphone Array Signal Processing*. Springer, 2008

[Bir13] BIRTH, Martin: *Sprecherlokalisierung mit einem 3D-Mikrofonfeld*, Brandenburgische Technische Universität Cottbus - Senftenberg, Bachelorarbeit, 2013

[Bir14] BIRTH, Martin: Sprecherlokalisierung mit einem 3D-Mikrofonfeld. In: *Konferenz Elektronische Sprachsignalverarbeitung (ESSV)*, TUDpress, 2014, S. 233–239

[Bit01] BITZER, Jörg: *Mehrkanalige Geräuschunterdrückungssysteme*, Universität Bremen, Dissertation, 2001

[BW01] BRANDSTEIN, Prof. M. ; WARD, Dr. D.: *Microphon Arrays: Signal Processing Techniques and Applications*. Springer, 2001

[CBH06] CHEN, Jingdong ; BENESTY, Jacob ; HUANG, Yiteng: Time Delay Estimation in Room Acoustic Environments: An Overview. In: *EURASIP J. Appl. Signal Process.* 2006 (2006), Januar, 170–170. http://dx.doi.org/10.1155/ASP/2006/26503. – DOI 10.1155/ASP/2006/26503. – ISSN 1110–8657

[CH94] CHAN, Y.T. ; HO, K.C.: A simple and efficient estimator for hyperbolic location. In: *Signal Processing, IEEE Transactions on* 42 (1994), Aug, Nr. 8, S. 1905–1915. http://dx.doi.org/10.1109/78.301830. – DOI 10.1109/78.301830. – ISSN 1053–587X

[Cha05] CHANDRAN, Sathish: *Advances in Direction-of-Arrival Estimation*. Norwood : Artech House, 2005 https://cds.cern.ch/record/995264

[Cha14] CHAFIK, Oliver: *ReadFromAndWriteToBuffersAndImages*. Website. https://code.google.com/p/javacl/wiki/ReadFromAndWriteToBuffersAndImages. Version: May 2014. – Zugriff: 25.12.2015

[Cha15] CHAFIK, Oliver: *javaCL, OpenCL bindings for Java*. https://github.com/nativelibs4java/JavaCL. Version: 2015. – Zugriff: 27. August 2015

[CHB05] CHEN, Jingdong ; HUANG, Y.A. ; BENESTY, J.: Time delay estimation via multichannel cross-correlation [audio signal processing applications]. In: *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on* Bd. 3, 2005. – ISSN 1520–6149, S. iii/49–iii/52 Vol. 3

[Dom12] DOMINGUEZ, Raquel M.: *Evaluating different Java bindings for OpenCL*, University Carlos III of Madrid, University of Seoul, Bachelor Thesis, 2012. http://e-archivo.uc3m.es/bitstream/handle/10016/17183/finalversionPFC_Raquel_Medina.pdf

[Dre96] DREWS, Martin: Speaker Localization and Its Application to Time Delay Estimators for Multi-Microphone Speech Enhancement Systems. In: *Proc. Of VIII European Signal Processing Conference* 1 (1996), September, Nr. 8, 483-486. http://www.eurasip.org/Proceedings/Eusipco/1996/paper/sp_7.pdf

[Dre99] DREWS, Martin: *Mikrofonarrays und mehrkanalige Signalverarbeitung zur Verbesserung gestörter Sprache*, Technische Universität Berlin, Dissertation, 1999

[ES07] EILEBRECHT, Karl ; STARKE, Gernot: *Patterns kompakt - Entwurfsmuster für effektive Software-Entwicklung*. Bd. 2. Spektrum Akademischer Verlag, 2007. – ISBN 978–3–8274–1591–2

[HB04] HUANG, Yiteng ; BENESTY, Jacob: *Audio Signal Processing, For Next-Generation Multimedia Communication Systems*. Kluwer Academic Publishers, 2004

[Hes15] HESSE, F.: *Anpassung einer vorhandenen Positioniereinrichtung*, BTU Cottbus, Bachelorarbeit, 2015

[Hof08] HOFER, Daniel: *Implementierung eine 64 Kanal Mikrofonarrays*, Universität für Musik und Darstellende Kunst Graz, Technische Universität Graz, Dimplomarbeit, 2008

[HR09] HAYKIN, Simon ; RAYLIU, K. J.: *Handbook on array processing and sensor networks*. Wiley, 2009

[HR10] HABIB, Tania ; ROMSDORFER, Harald: COMPARISON OF SRP-PHAT AND MULTIBAND-POPI ALGORITHMS FOR SPEAKER LOCALIZATION USING PARTICLE FILTERS. In: *Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx-10)* Bd. n/a, 2010, S. n/a – n/a

[HW14] HOFFMANN, Rüdiger ; WOLFF, Matthias: *Intelligente Signalverarbeitung 1*. Springer-Verlag, 2014

[JCR13] JILLINGS, Nicholas ; CLIFFORD, Alice ; REISS, Joshua D.: *Performance Optimization of GCC-PHAT for Delay and Polarity Correction under Real World Conditions*. 20013

[KC76] KNAPP, C. ; CARTER, G.Clifford: The generalized correlation method for estimation of time delay. In: *Acoustics, Speech and Signal Processing, IEEE Transactions on* 24 (1976), Aug, Nr. 4, S. 320–327. http://dx.doi.org/10.1109/TASSP.1976.1162830. – DOI 10.1109/TASSP.1976.1162830. – ISSN 0096–3518. – Zugriff: 16.11.2015

[KG13] KUMAR, Sumit ; GUPTA, Rajat: Estimating Time Delay using GCC for Speech Source Localisation. In: *International Journal of Application or Innovation in Engineering & Management (IJAIEM)* 2 (2013), May, Nr. 5, S. 24–30

[Khr14] KHRONOS OPENCL WORKING GROUP: *The OpenCL Specification*. Version: 2.0, Document Revision: 26. Beaverton, USA, 2014. https://www.khronos.org/registry/cl/specs/opencl-2.0.pdf

[KKB13] KRISHNAVENI, V ; KESAVAMURTHY, T ; B, Aparna.: Article: Beamforming for Direction-of-Arrival (DOA) Estimation-A Survey. In: *International Journal of Computer Applications* 61 (2013), January, Nr. 11, 4-11. http://research.ijcaonline.org/volume61/number11/pxc3884758.pdf. – Full text available

[KMI+07] KAWANISHI, M. ; MARUTA, R. ; IKOMA, N. ; KAWANO, H. ; MAEDA, H.: Sound target tracking in 3D using particle filter with 4 microphones. In: *SICE, 2007 Annual Conference*, 2007, S. 1427–1430

[KMR12] KUMATANI, K. ; MCDONOUGH, J. ; RAJ, B.: Microphone Array Processing for Distant Speech Recognition: From Close-Talking Microphones to Far-Field Sensors. In: *Signal*

Processing Magazine, IEEE 29 (2012), Nov, Nr. 6, 127-140. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6296525. – ISSN 1053–5888

[LK10] LIU, Di ; KHONG, Andy Wai H.: *A WAVELET-BASED GCC PREFILTERING ALGORITHM FOR SPEECH DOA ESTIMATION*. School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, 2010

[LW10] LIU, Wei ; WEISS, Stephan: *Wideband Beamforming: Concepts and Techniques*. Wiley Publishing, 2010. – ISBN 0470713925, 9780470713921

[LZBK11] LOMBARD, A. ; ZHENG, Yuanhang ; BUCHNER, H. ; KELLERMANN, W.: TDOA Estimation for Multiple Sound Sources in Noisy and Reverberant Environments Using Broadband Independent Component Analysis. In: *Audio, Speech, and Language Processing, IEEE Transactions on* 19 (2011), Aug, Nr. 6, S. 1490–1503. http://dx.doi.org/10.1109/TASL.2010.2092765. – DOI 10.1109/TASL.2010.2092765. – ISSN 1558–7916

[MA04] MUNGAMURU, B. ; AARABI, P.: Enhanced Sound Localization. In: *Trans. Sys. Man Cyber. Part B* 34 (2004), Juni, Nr. 3, 1526–1540. http://dx.doi.org/10.1109/TSMCB.2004.826398. – DOI 10.1109/TSMCB.2004.826398. – ISSN 1083–4419

[McC01] MCCOWAN, Iain: *Microphone Arrays, A Tutorial*. https://www.idiap.ch/~mccowan/arrays/tutorial.pdf. Version: 2001. – Zugriff: 13.04.2015

[Mer02] MERIMAA, Juha: Applications of a 3-D microphone array. In: *in Preprint 112th*, 2002, S. 1–6

[MGM+12] MUNSHI, Aaftab ; GASTER, Benedict R. ; MATTSON, Timothy G. ; FUNG, James ; GINSBURG, Dan: *OpenCL Programming Guide*. Pearson Education, Inc., 2012 http://ptgmedia.pearsoncmg.com/images/9780321749642/samplepages/0321749642.pdf. – Zugriff: 05.06.2015

[Moe09] MOESER, Michael: *Technische Akustik*. Springer, 2009. http://dx.doi.org/10.1007/978-3-540-89818-4. http://dx.doi.org/10.1007/978-3-540-89818-4

[MOK12] MOHAMED-OMAR, F. ; KNÖRZER, C.: *Virtuelle Surround-Systeme*. https://www.hdm-stuttgart.de/~curdt/Virtuelle%20Surround-Systeme.pdf. Version: 2012. – Zugriff: 25.08.2015

[Ora15a] ORACLE: *Class HashMap<K,V>*. https://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html. Version: 2015. – Zugriff: 10.11.2015

[Ora15b] ORACLE: *Class LinkedBlockingQueue<E>*. https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/LinkedBlockingQueue.html. Version: 2015. – Zugriff: 10.11.2015

[Our07] OURSLAND, David: *Particle Filter Demo*. http://www.oursland.net/projects/particlefilter/. Version: 01 2007. – Zugriff: 30.12.2015

[PA13] POURMOHAMMAD, Ali ; AHADI, SeyedMohammad: N-dimensional N-microphone sound source localization. In: *EURASIP Journal on Audio, Speech, and Music Processing* 2013 (2013), Nr. 1. http://dx.doi.org/10.1186/1687-4722-2013-27. – DOI 10.1186/1687–4722–2013–27

[Sar13] SARRADJ, Ennes: *Schallmesstechnik - Kurzskript zur Vorlesung*. 2013

[Sen02] SENGPIEL, Eberhard: Laufzeitdifferenz und Phasenverschiebung / UdK Berlin. Version: 2002. http://www.sengpielaudio.com/LaufzeitdifferenzUndPhasenverschiebung.pdf. 2002. – Forschungsbericht

[SH14] SEGURA, Carlos ; HERNANDO, Javier: 3D Joint Speaker Position and Orientation Tracking with Particle Filters. In: *Sensors* 14 (2014), Nr. 2, 2259. http://dx.doi.org/10.3390/s140202259. – DOI 10.3390/s140202259. – ISSN 1424–8220

[TSA08] THAI, Derek Z. ; SCIENCE, Defence ; (AUSTRALIA)., Technology O.: *Speaker localisation using time difference of arrival*. DSTO Edinburgh, S. Aust, 2008 http://pandora.nla.gov.au/pan/24592/20080827-1348/DSTO-TR-2126%20PR.pdfhttp://www.dsto.defence.gov.au/publications/scientific_record.php?record=9298

[Whi16] WHILER, Oliver: *Refactoring Java Code*. http://www.methodsandtools.com/archive/archive.php?id=4. Version: 2016. – Zugriff: 15.01.2016

[WLW03] WARD, Darren B. ; LEHMANN, E. A. ; WILLIAMSON, R. C.: Particle filtering algorithms for tracking an acoustic source in a reverberant environment. In: *Speech and Audio Processing, IEEE Transactions on* 11 (2003), November, Nr. 6, 826–836. http://dx.doi.org/10.1109/tsa.2003.818112. – DOI 10.1109/tsa.2003.818112. – ISSN 1063–6676

[Wol15a] WOLFF, Matthias: *How to Program an LCARS Panel*. https://github.com/matthias-wolff/LCARSWT/wiki/How-to-Program-an-LCARS-Panel. Version: 2015. – Zugriff: 29.12.2015

[Wol15b] WOLFF, Matthias: *LCARS Widget Toolkit*. https://github.com/matthias-wolff/LCARSWT. Version: 2015. – Zugriff: 28.12.2015

[ZGET00] ZHENG, Y.R. ; GOUBRAN, Rafik A. ; EL-TANANY, Mohamed: A broadband adaptive beamformer usind nested arrays and multirate techniques. In: *Proc. IEEE DSP Workshop*, 2000, 1-6

A Anhang

A.1 Dreidimensionales Mikrofonfeld

Abbildung 19: Cognitive Systems Lab mit einem dreidimensionalen Mikrofonfeld (Teilmikrofonfeld 1 um 72" Touchscreen und Teilmikrofonfeld 2 an der Raumdecke)

A.1.1 Teilmikrofonfeld 1 (TV)

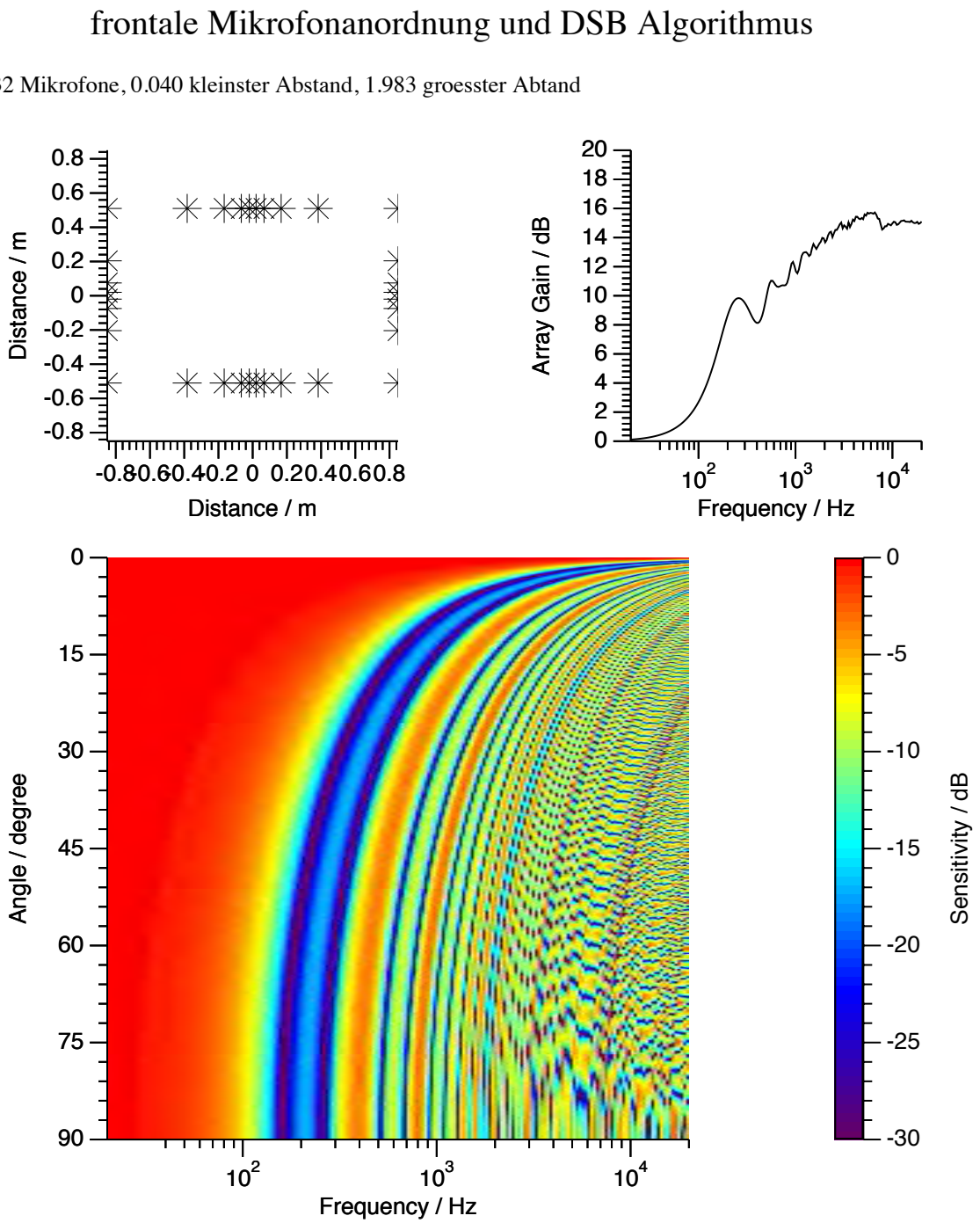


Abbildung 20: Datenblatt von Teilmikrofonfeld 1 (Quelle: Dipl.-Ing. Thomas Fehér, TU Dresden)

A.1.2 Teilmikrofonfeld 2 (Decke)

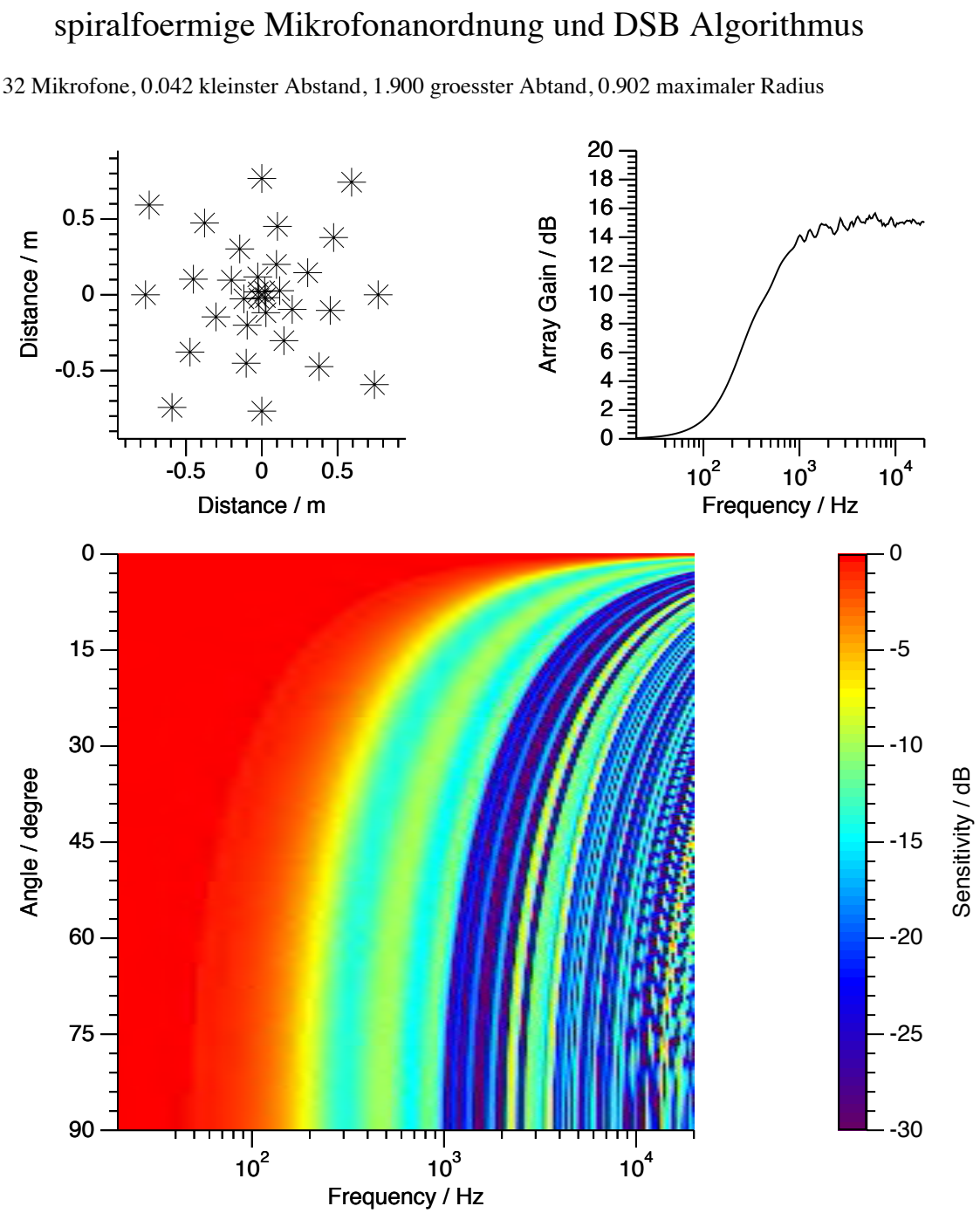


Abbildung 21: Datenblatt von Teilmikrofonfeld 2 (Quelle: Dipl.-Ing. Thomas Fehér, TU Dresden)

A.1.3 Relative Mikrofonpositionsdaten

Nr	x in m	y in m
1	-0,850	2,050
2	-0,383	2,050
3	-0,176	2,050
4	-0,066	2,050
5	-0,020	2,050
6	0,020	2,050
7	0,066	2,050
8	0,167	2,050
9	0,383	2,050
10	0,850	2,050
11	-0,850	1,030
12	-0,383	1,030
13	-0,167	1,030
14	-0,066	1,030
15	-0,020	1,030
16	0,020	1,030
17	0,066	1,030
18	0,167	1,030
19	0,383	1,030
20	0,850	1,030
21	-0,850	1,336
22	-0,850	1,465
23	-0,850	1,520
24	-0,850	1,560
25	-0,850	1,615
26	-0,850	1,744
27	0,850	1,336
28	0,850	1,465
29	0,850	1,520
30	0,850	1,560
31	0,850	1,615
32	0,850	1,744

Tabelle 4: Relative Mikrofonpositionen von Mikrofonfeld 1 (TV)

Nr	x in m	y in m
33	0,023	0,019
34	0,118	-0,027
35	0,097	-0,200
36	-0,146	-0,303
37	-0,451	-0,103
38	-0,474	0,378
39	0,000	0,767
40	0,743	0,592
41	0,019	-0,023
42	-0,027	-0,118
43	-0,200	-0,097
44	-0,303	0,146
45	-0,103	0,451
46	0,378	0,474
47	0,767	0,000
48	0,592	-0,743
49	-0,023	-0,019
50	-0,118	0,027
51	-0,097	0,200
52	0,146	0,303
53	0,451	0,103
54	0,474	-0,378
55	0,000	-0,767
56	-0,743	-0,592
57	-0,019	0,023
58	0,027	0,118
59	0,200	0,097
60	0,303	-0,146
61	0,103	-0,451
62	-0,378	-0,474
63	-0,767	0,000
64	-0,592	0,743

Tabelle 5: Relative Mikrofonpositionen von Mikrofonfeld 2 (Decke)

A.1.4 Messung der Positionsdaten

Mic Nr.	ID im CSL	Koordianten x	Y	z	Distanz SOLL (errechnet) in cm	Distanz IST (gemessen) (1) in cm	(2) in cm	(3) in cm	Distanz IST (Mittel) in cm	Abweichung SOLL-IST in cm
I/01	0	-85,60	217,50	205,10	310,97	311,1	310,8	310,7	310,87	0,10
I/02	1	-38,30	218,20	205,20	301,97	301,9	302,0	302,2	302,03	0,06
I/03	2	-16,70	218,00	205,30	299,92	299,8	300,0	300,0	299,93	0,02
I/04	3	-6,60	218,10	205,30	299,60	299,7	299,9	299,3	299,63	0,03
I/05	4	-2,00	218,40	205,30	299,75	299,3	299,8	300,1	299,73	0,02
I/06	5	2,00	218,80	205,20	299,97	300,2	300,3	299,9	300,13	0,16
I/07	6	6,60	218,80	205,10	299,97	299,7	299,8	300,4	299,97	0,01
I/08	7	16,70	218,50	205,10	300,15	300,4	299,9	300,1	300,13	0,01
I/09	8	38,30	218,80	205,00	302,27	302,4	302,2	302,3	302,30	0,03
I/10	9	85,60	220,10	204,80	312,59	312,6	312,4	312,7	312,57	0,03
I/11	10	-85,60	218,10	103,20	256,02	255,9	255,9	256,2	256,00	0,02
I/12	11	-38,40	218,60	102,90	244,64	244,2	244,7	245,0	244,63	0,01
I/13	12	-16,70	218,90	102,70	242,37	242,4	242,2	242,5	242,37	0,00
I/14	13	-6,60	219,20	102,70	242,16	242,2	242,0	242,3	242,17	0,01
I/15	14	-2,00	219,20	102,70	242,07	241,8	242,2	242,1	242,03	0,04
I/16	15	2,00	219,10	102,60	241,94	241,9	242,0	241,8	241,90	0,04
I/17	16	6,60	219,40	102,60	242,29	242,4	242,2	242,2	242,27	0,03
I/18	17	16,70	219,50	102,50	242,83	242,8	242,9	242,9	242,87	0,04
I/19	18	38,30	220,00	102,30	245,63	245,7	245,7	245,6	245,67	0,04
I/20	19	85,60	220,60	102,50	257,87	257,9	257,8	257,9	257,87	0,01
I/21	20	-85,60	217,60	134,00	269,51	269,5	269,6	269,3	269,47	0,04
I/22	21	-85,60	217,50	146,90	276,07	276,2	276,1	275,9	276,07	0,00
I/23	22	-85,60	217,50	152,40	279,03	279,0	279,1	279,1	279,07	0,03
I/24	23	-85,60	217,30	156,10	280,92	280,9	281,1	280,8	280,93	0,02
I/25	24	-85,60	217,30	161,60	284,01	283,9	283,9	284,1	283,97	0,04
I/26	25	-85,60	217,30	174,50	291,54	291,8	291,5	291,6	291,63	0,09
I/27	26	85,60	220,50	133,20	271,46	271,4	271,4	271,5	271,43	0,03
I/28	27	85,60	220,80	146,10	278,25	278,4	278,2	278,2	278,27	0,01
I/29	28	85,60	220,90	151,50	281,21	281,2	281,1	281,2	281,17	0,04
I/30	29	85,60	220,60	155,30	283,04	283,0	283,0	283,0	283,00	0,04
I/31	30	85,60	220,60	160,90	286,15	286,1	286,1	286,3	286,17	0,02
I/32	31	85,60	220,30	173,70	293,31	293,2	293,5	293,2	293,30	0,01
II/33	32	7,60	1,88	234,72	234,85	234,8	234,8	234,7	234,77	0,08
II/34	33	17,10	-2,67	235,40	236,03	235,9	236,2	236,0	236,03	0,00
II/35	34	15,00	-19,78	237,95	239,24	238,8	238,8	238,8	238,80	0,44
II/36	35	-9,30	-29,97	239,46	241,51	241,3	241,2	241,2	241,23	0,28
II/37	36	-39,80	-10,19	236,52	240,06	240,1	240,1	239,9	240,03	0,03
II/38	37	-42,10	37,39	229,43	236,24	236,0	236,1	236,2	236,10	0,14
II/39	38	5,30	75,86	223,70	236,28	236,1	235,9	235,8	235,93	0,34
II/40	39	79,60	58,55	226,28	246,92	247,5	247,4	245,0	246,63	0,28
II/41	40	7,20	-2,27	235,34	235,46	235,3	235,3	235,4	235,33	0,13
II/42	41	2,60	-11,67	236,74	237,04	236,8	237,0	236,9	236,90	0,14
II/43	42	-14,70	-9,59	236,43	237,08	237,1	237,0	236,9	237,00	0,08
II/44	43	-25,00	14,44	232,85	234,63	234,6	234,5	234,4	234,50	0,13
II/45	44	-5,00	44,61	228,36	232,73	232,4	232,3	232,3	232,33	0,39
II/46	45	43,10	46,88	228,02	236,74	237,3	236,6	236,7	236,87	0,12
II/47	46	82,00	0,00	235,00	248,90	248,8	249,4	249,1	249,10	0,20
II/48	47	64,50	-73,49	245,94	264,67	264,7	265,1	265,4	265,07	0,40
II/49	48	3,00	-1,88	235,28	235,31	235,4	235,3	235,2	235,30	0,01
II/50	49	-6,50	2,67	234,60	234,71	234,5	234,7	234,5	234,57	0,14
II/51	50	-4,40	19,78	232,05	232,94	232,8	232,7	232,6	232,70	0,24
II/52	51	19,90	29,97	230,54	233,33	233,4	233,3	233,3	233,33	0,01
II/53	52	50,40	10,19	233,48	239,08	239,6	239,5	239,5	239,53	0,46
II/54	53	52,70	-37,39	240,57	249,09	249,6	249,4	249,6	249,53	0,44
II/55	54	5,30	-75,86	246,30	257,77	258,1	258,3	258,2	258,20	0,43
II/56	55	-69,00	-58,55	243,72	259,98	259,9	260,2	259,9	260,00	0,02
II/57	56	3,40	2,27	234,66	234,70	234,7	234,6	234,7	234,67	0,03
II/58	57	8,00	11,67	233,26	233,69	233,6	233,6	233,5	233,57	0,12
II/59	58	25,30	9,59	233,57	235,13	235,2	235,1	235,3	235,20	0,07
II/60	59	35,60	-14,44	237,15	240,24	240,4	240,6	240,5	240,50	0,26
II/61	60	15,60	-44,61	241,64	246,22	245,8	245,9	245,9	245,87	0,35
II/62	61	-32,50	-46,88	241,98	248,61	248,5	248,4	248,3	248,40	0,21
II/63	62	-71,40	0,00	235,00	245,61	245,8	245,7	245,4	245,63	0,03
II/64	63	-53,90	73,49	224,06	241,88	241,7	241,7	241,7	241,70	0,18
								Mittlerer Fehler:		0,11
								Standardabweichung:		0,13
								Varianz:		0,02

Abbildung 22: Tabellarische Darstellung der neu ermittelten Mikrofonkoordinaten im Vergleich zu den gemessenen Distanzwerten

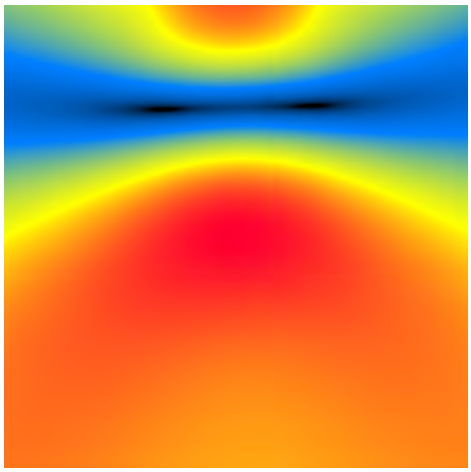
A.1.5 Disparität der Mikrofonkoordinaten

Mic Nr.	ID im CSL	Koordianten ALT x	y	z	Distanz ALT (errechnet) in cm	Koordianten NEU x	y	z	Distanz NEU (errechnet) in cm	Dis.-Abw. NEU-ALT in cm
I/01	0	-85	220	205	312,49	-85,60	217,5	205,10	310,97	1,52
I/02	1	-38,3	220	205	303,14	-38,30	218,2	205,20	301,97	1,17
I/03	2	-16,7	220	205	301,17	-16,70	218,0	205,30	299,92	1,25
I/04	3	-6,6	220	205	300,78	-6,60	218,1	205,30	299,60	1,18
I/05	4	-2	220	205	300,71	-2,00	218,4	205,30	299,75	0,96
I/06	5	2	220	205	300,71	2,00	218,8	205,20	299,97	0,74
I/07	6	6,6	220	205	300,78	6,60	218,8	205,10	299,97	0,81
I/08	7	16,7	220	205	301,17	16,70	218,5	205,10	300,15	1,03
I/09	8	38,3	220	205	303,14	38,30	218,8	205,00	302,27	0,87
I/10	9	85	220	205	312,49	85,60	220,1	204,80	312,59	0,10
I/11	10	-85	220	103	257,36	-85,60	218,1	103,20	256,02	1,34
I/12	11	-38,3	220	103	245,92	-38,40	218,6	102,90	244,64	1,28
I/13	12	-16,7	220	103	243,49	-16,70	218,9	102,70	242,37	1,12
I/14	13	-6,6	220	103	243,01	-6,60	219,2	102,70	242,16	0,85
I/15	14	-2	220	103	242,93	-2,00	219,2	102,70	242,07	0,85
I/16	15	2	220	103	242,93	2,00	219,1	102,60	241,94	0,98
I/17	16	6,6	220	103	243,01	6,60	219,4	102,60	242,29	0,71
I/18	17	16,7	220	103	243,49	16,70	219,5	102,50	242,83	0,66
I/19	18	38,3	220	103	245,92	38,30	220,0	102,30	245,63	0,29
I/20	19	85	220	103	257,36	85,60	220,6	102,50	257,87	0,51
I/21	20	-85	220	133,6	271,06	-85,60	217,6	134,00	269,51	1,56
I/22	21	-85	220	146,5	277,65	-85,60	217,5	146,90	276,07	1,58
I/23	22	-85	220	152	280,59	-85,60	217,5	152,40	279,03	1,55
I/24	23	-85	220	156	282,77	-85,60	217,3	156,10	280,92	1,86
I/25	24	-85	220	161,5	285,84	-85,60	217,3	161,60	284,01	1,84
I/26	25	-85	220	174,4	293,33	-85,60	217,3	174,50	291,54	1,78
I/27	26	85	220	133,6	271,06	85,60	220,5	133,20	271,46	0,40
I/28	27	85	220	146,5	277,65	85,60	220,8	146,10	278,25	0,61
I/29	28	85	220	152	280,59	85,60	220,9	151,50	281,21	0,62
I/30	29	85	220	156	282,77	85,60	220,6	155,30	283,04	0,26
I/31	30	85	220	161,5	285,84	85,60	220,6	160,90	286,15	0,30
I/32	31	85	220	174,4	293,33	85,60	220,3	173,70	293,31	0,02
II/33	32	2,3	1,9	234,6	234,62	7,60	1,88	234,72	234,85	0,23
II/34	33	11,8	-2,7	235,1	235,41	17,10	-2,67	235,40	236,03	0,62
II/35	34	9,7	-20,0	236,7	237,74	15,00	-19,78	237,95	239,24	1,50
II/36	35	-14,6	-30,3	237,7	240,07	-9,30	-29,97	239,46	241,51	1,44
II/37	36	-45,1	-10,3	235,8	240,30	-39,80	-10,19	236,52	240,06	0,24
II/38	37	-47,4	37,8	231,2	239,02	-42,10	37,39	229,43	236,24	2,78
II/39	38	0	76,7	227,5	240,08	5,30	75,86	223,70	236,28	3,81
II/40	39	74,3	59,2	229,2	248,11	79,60	58,55	226,28	246,92	1,19
II/41	40	1,9	-2,3	235	235,02	7,20	-2,27	235,34	235,46	0,44
II/42	41	-2,7	-11,8	235,9	236,21	2,60	-11,67	236,74	237,04	0,83
II/43	42	-20	-9,7	235,7	236,75	-14,70	-9,59	236,43	237,08	0,33
II/44	43	-30,3	14,6	233,4	235,81	-25,00	14,44	232,85	234,63	1,18
II/45	44	-10,3	45,1	230,5	235,10	-5,00	44,61	228,36	232,73	2,37
II/46	45	37,8	47,4	230,3	238,15	43,10	46,88	228,02	236,74	1,40
II/47	46	76,7	0,0	234,8	247,01	82,00	0,00	235,00	248,90	1,89
II/48	47	59,2	-74,3	241,9	259,89	64,50	-73,49	245,94	264,67	4,78
II/49	48	-2,3	-1,9	235	235,02	3,00	-1,88	235,28	235,31	0,29
II/50	49	-11,8	2,7	234,6	234,91	-6,50	2,67	234,60	234,71	0,20
II/51	50	-9,7	20,0	232,9	233,96	-4,40	19,78	232,05	232,94	1,02
II/52	51	14,6	30,3	231,9	234,33	19,90	29,97	230,54	233,33	1,00
II/53	52	45,1	10,3	233,8	238,33	50,40	10,19	233,48	239,08	0,75
II/54	53	47,4	-37,8	238,4	245,99	52,70	-37,39	240,57	249,09	3,11
II/55	54	0	-76,7	242,1	253,96	5,30	-75,86	246,30	257,77	3,81
II/56	55	-74,3	-59,2	240,5	258,58	-69,00	-58,55	243,72	259,98	1,40
II/57	56	-1,9	2,3	234,6	234,62	3,40	2,27	234,66	234,70	0,08
II/58	57	2,7	11,8	233,7	234,01	8,00	11,67	233,26	233,69	0,32
II/59	58	20	9,7	233,3	234,36	25,30	9,59	233,57	235,13	0,78
II/60	59	30,3	-14,6	236,2	238,58	35,60	-14,44	237,15	240,24	1,66
II/61	60	10,3	-45,1	239,1	243,53	15,60	-44,61	241,64	246,22	2,69
II/62	61	-37,8	-47,4	239,3	246,86	-32,50	-46,88	241,98	248,61	1,75
II/63	62	-76,7	0,0	234,8	247,01	-71,40	0,00	235,00	245,61	1,40
II/64	63	-59,2	74,3	227,7	246,72	-53,90	73,49	224,06	241,88	4,84
								Mittlerer Fehler:		1,26
								Standardabweichung:		1,03
								Varianz:		1,09

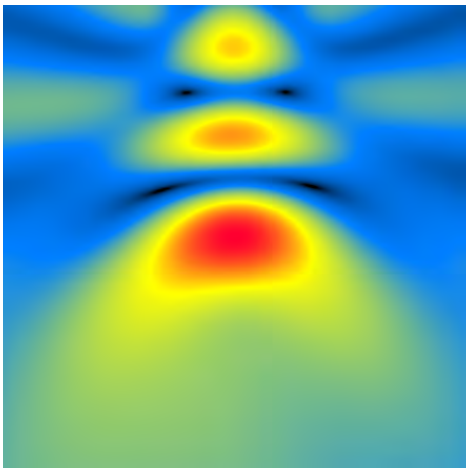
OFFSET NEU	
y	220
z	153
OFFSET ALT	
y	220
z	154
OFFSET NEU	
x	5,3
z	235
ROTATION	
alpha	-8,47
radiant	-0,15
cos	0,99
sin	-0,15
OFFSET ALT	
x	0
z	235

Abbildung 23: Vergleich von bestehenden und neuen Mikrofonpositionen

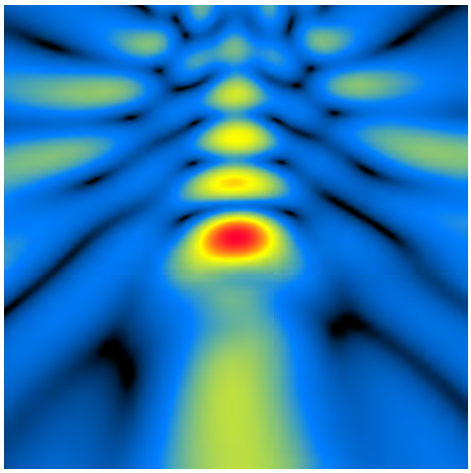
A.2 Frequenzabhängige Raumquerschnitte der Sensitivitätsverteilung



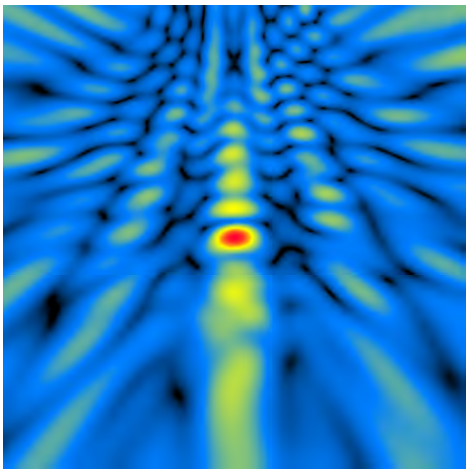
$f = 100\,\mathrm{Hz}$



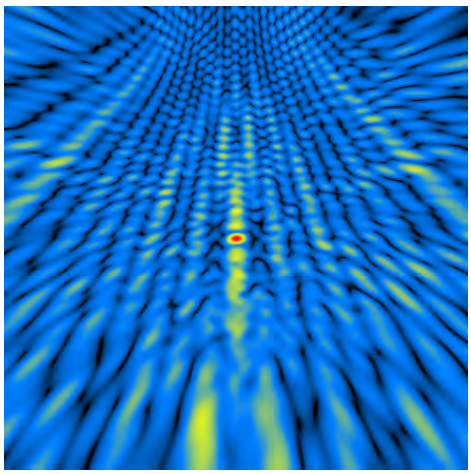
$f = 250\,\mathrm{Hz}$



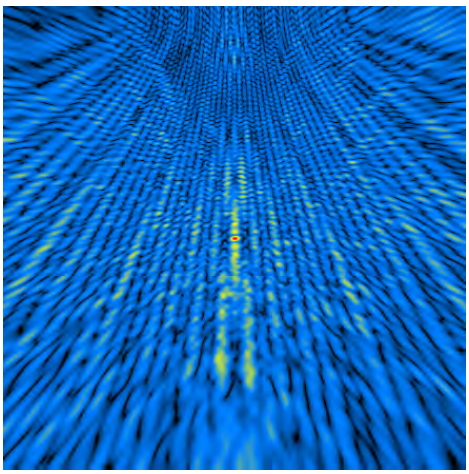
$f = 500\,\mathrm{Hz}$



$f = 1000\,\mathrm{Hz}$



$f = 2500\,\mathrm{Hz}$



$f = 5000\,\mathrm{Hz}$

Abbildung 24: Frequenzabhängigkeit der Raumquerschnitte aus der Sensitivitätsverteilung für $N = 64$ Mikrofone, Zielposition $q_s = (0; 0; 1,6)$ und Mikrofonfeld 2 bei $p = 0,57\,\mathrm{m}$

A.3 Konfiguration des Industrie-PCs

A.3.1 CPU Eigenschaften

```
==================================[ System / CPU ]
- CPU Name: Intel(R) Core(TM)2 Quad CPU Q9550 @ 2.83GHz
- CPU Core Speed: 2826 MHz
- CPU logical cores: 4
- Family: 6 - Model: 7 - Stepping: 10
- Physical Memory Size: 8192 MB
- Operating System: Windows 7 64-bit build 7601 [Service Pack 1]
- PhysX Version: 9140702
- Java(TM) SE Runtime Environment (build 1.8.0_25-b17)
- Java HotSpot(TM) 64-Bit Server VM (build 25.25-b02, mixed mode)
```

Listing 1: CPU Eigenschaften des Industrie-PCs

A.3.2 GPU Eigenschaften

```
==================================[ Graphics Adapters / GPUs ]
- Current Display Mode: 1920x1080 @ 60 Hz - 32 bpp
- Num GPUs: 1

- GPU 1
  - Name: NVIDIA GeForce GTX 660
  - GPU codename: GK106
  - Device ID: 10DE-11C0
  - Subdevice ID: 1458-354E
  - Driver: 9.18.13.4752 (R347.52)
  - Branch: r346_00-321
  - Bus Id: 1
  - Shader cores: 960
  - Texture units: 80
  - ROP units: 24
  - TDP: 140W
  - BIOS version: 80.06.10.00.3d
  - Memory size: 2047MB
  - Memory type: GDDR5
  - Memory bus width: 192-bit
  - GPU base clock: 1032 MHz
  - GPU boost clock: 1097 MHz
  - GPU power target: 100 % TDP
  - GPU min power target: 46 % TDP
  - GPU max power target: 110 % TDP
```

Listing 2: GPU Eigenschaften des Industrie-PCs

A.3.3 OpenCL Eigenschaften

```
==================================[ OpenCL Capabilities ]
- Num OpenCL platforms: 1
- CL_PLATFORM_NAME: NVIDIA CUDA
- CL_PLATFORM_VENDOR: NVIDIA Corporation
- CL_PLATFORM_VERSION: OpenCL 1.1 CUDA 7.0.23
- CL_PLATFORM_PROFILE: FULL_PROFILE
- Num devices: 1

    - CL_DEVICE_NAME: GeForce GTX 660
    - CL_DEVICE_VENDOR: NVIDIA Corporation
    - CL_DRIVER_VERSION: 347.52
    - CL_DEVICE_PROFILE: FULL_PROFILE
    - CL_DEVICE_VERSION: OpenCL 1.1 CUDA
    - CL_DEVICE_TYPE: GPU
    - CL_DEVICE_VENDOR_ID: 0x10DE
    - CL_DEVICE_MAX_COMPUTE_UNITS: 5
    - CL_DEVICE_MAX_CLOCK_FREQUENCY: 1097MHz
    - CL_NV_DEVICE_COMPUTE_CAPABILITY_MAJOR: 3
    - CL_NV_DEVICE_COMPUTE_CAPABILITY_MINOR: 0
    - CL_NV_DEVICE_REGISTERS_PER_BLOCK: 65536
    - CL_NV_DEVICE_WARP_SIZE: 32
    - CL_NV_DEVICE_GPU_OVERLAP: 1
    - CL_NV_DEVICE_KERNEL_EXEC_TIMEOUT: 1
    - CL_NV_DEVICE_INTEGRATED_MEMORY: 0
    - CL_DEVICE_ADDRESS_BITS: 32
    - CL_DEVICE_MAX_MEM_ALLOC_SIZE: 524288KB
    - CL_DEVICE_GLOBAL_MEM_SIZE: 2048MB
    - CL_DEVICE_MAX_PARAMETER_SIZE: 4352
    - CL_DEVICE_GLOBAL_MEM_CACHELINE_SIZE: 128 Bytes
    - CL_DEVICE_GLOBAL_MEM_CACHE_SIZE: 80KB
    - CL_DEVICE_ERROR_CORRECTION_SUPPORT: NO
    - CL_DEVICE_LOCAL_MEM_TYPE: Local (scratchpad)
    - CL_DEVICE_LOCAL_MEM_SIZE: 47KB
    - CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE: 64KB
    - CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS: 3
    - CL_DEVICE_MAX_WORK_ITEM_SIZES: [1024 ; 1024 ; 64]
    - CL_DEVICE_MAX_WORK_GROUP_SIZE: 1024
    - CL_EXEC_NATIVE_KERNEL: 17152448
    - CL_DEVICE_IMAGE_SUPPORT: YES
    - CL_DEVICE_MAX_READ_IMAGE_ARGS: 256
    - CL_DEVICE_MAX_WRITE_IMAGE_ARGS: 16
    - CL_DEVICE_IMAGE2D_MAX_WIDTH: 32768
    - CL_DEVICE_IMAGE2D_MAX_HEIGHT: 32768
    - CL_DEVICE_IMAGE3D_MAX_WIDTH: 4096
    - CL_DEVICE_IMAGE3D_MAX_HEIGHT: 4096
    - CL_DEVICE_IMAGE3D_MAX_DEPTH: 4096
    - CL_DEVICE_MAX_SAMPLERS: 32
    - CL_DEVICE_PREFERRED_VECTOR_WIDTH_CHAR: 1
    - CL_DEVICE_PREFERRED_VECTOR_WIDTH_SHORT: 1
```

```
50    - CL_DEVICE_PREFERRED_VECTOR_WIDTH_INT: 1
51    - CL_DEVICE_PREFERRED_VECTOR_WIDTH_LONG: 1
52    - CL_DEVICE_PREFERRED_VECTOR_WIDTH_FLOAT: 1
53    - CL_DEVICE_PREFERRED_VECTOR_WIDTH_DOUBLE: 1
54    - CL_DEVICE_EXTENSIONS: 16
55    - Extensions:
56        - cl_khr_byte_addressable_store
57        - cl_khr_icd
58        - cl_khr_gl_sharing
59        - cl_nv_compiler_options
60        - cl_nv_device_attribute_query
61        - cl_nv_pragma_unroll
62        - cl_nv_d3d9_sharing
63        - cl_nv_d3d10_sharing
64        - cl_khr_d3d10_sharing
65        - cl_nv_d3d11_sharing
66        - cl_nv_copy_opts
67        - cl_khr_global_int32_base_atomics
68        - cl_khr_global_int32_extended_atomics
69        - cl_khr_local_int32_base_atomics
70        - cl_khr_local_int32_extended_atomics
71        - cl_khr_fp64
```

Listing 3: OpenCL Eigenschaften der *NVIDIA GeForce GTX 660* Grafikkarte

A.4 UML-Klassendiagramme

A.4.1 HammerfallAudioDevice UML-Klassendiagramm

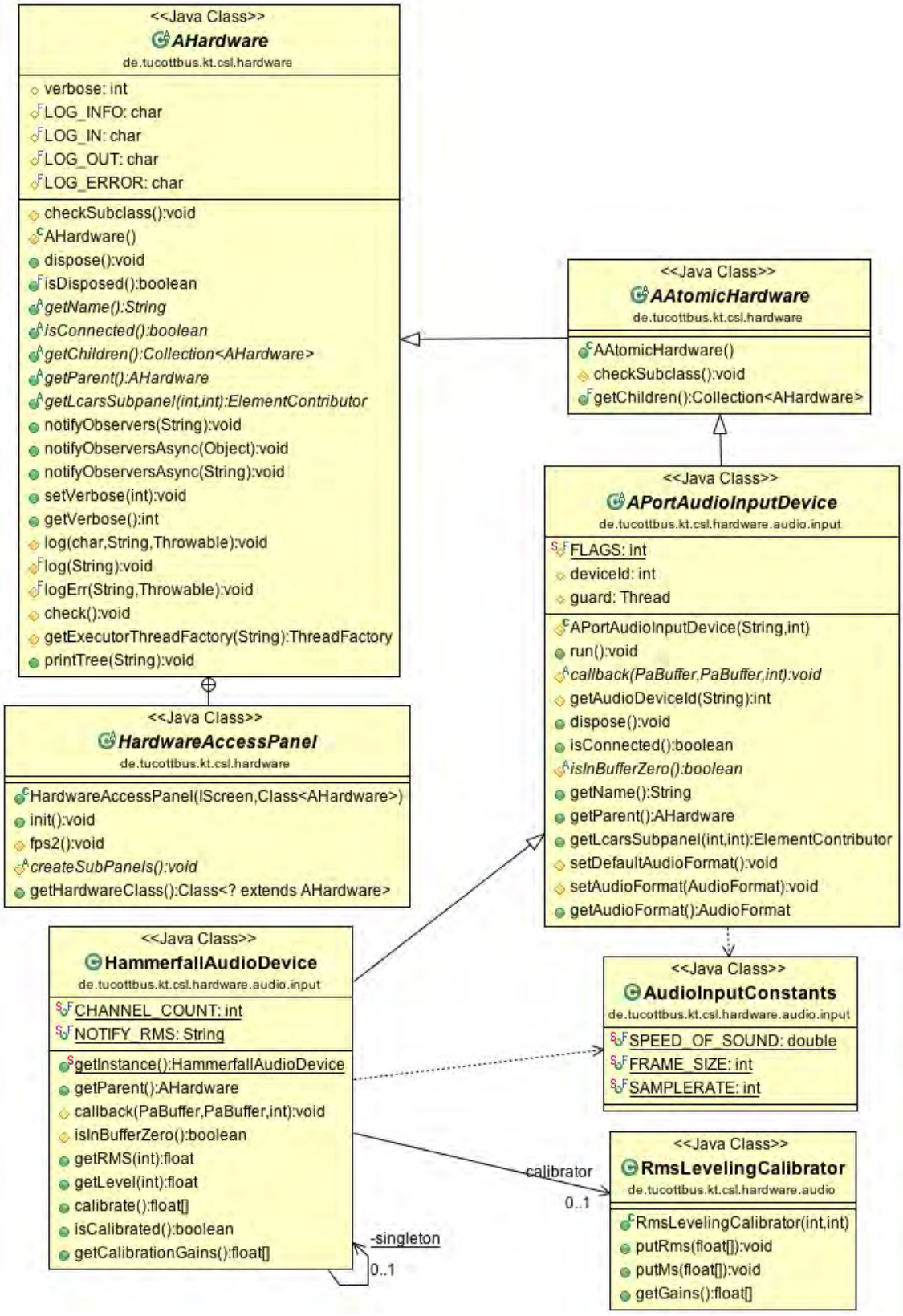


Abbildung 25: UML-Klassendiagramm des abgeleiteten **HammerfallAudioDevice**

A.4.2 Beamformer UML-Klassendiagramm

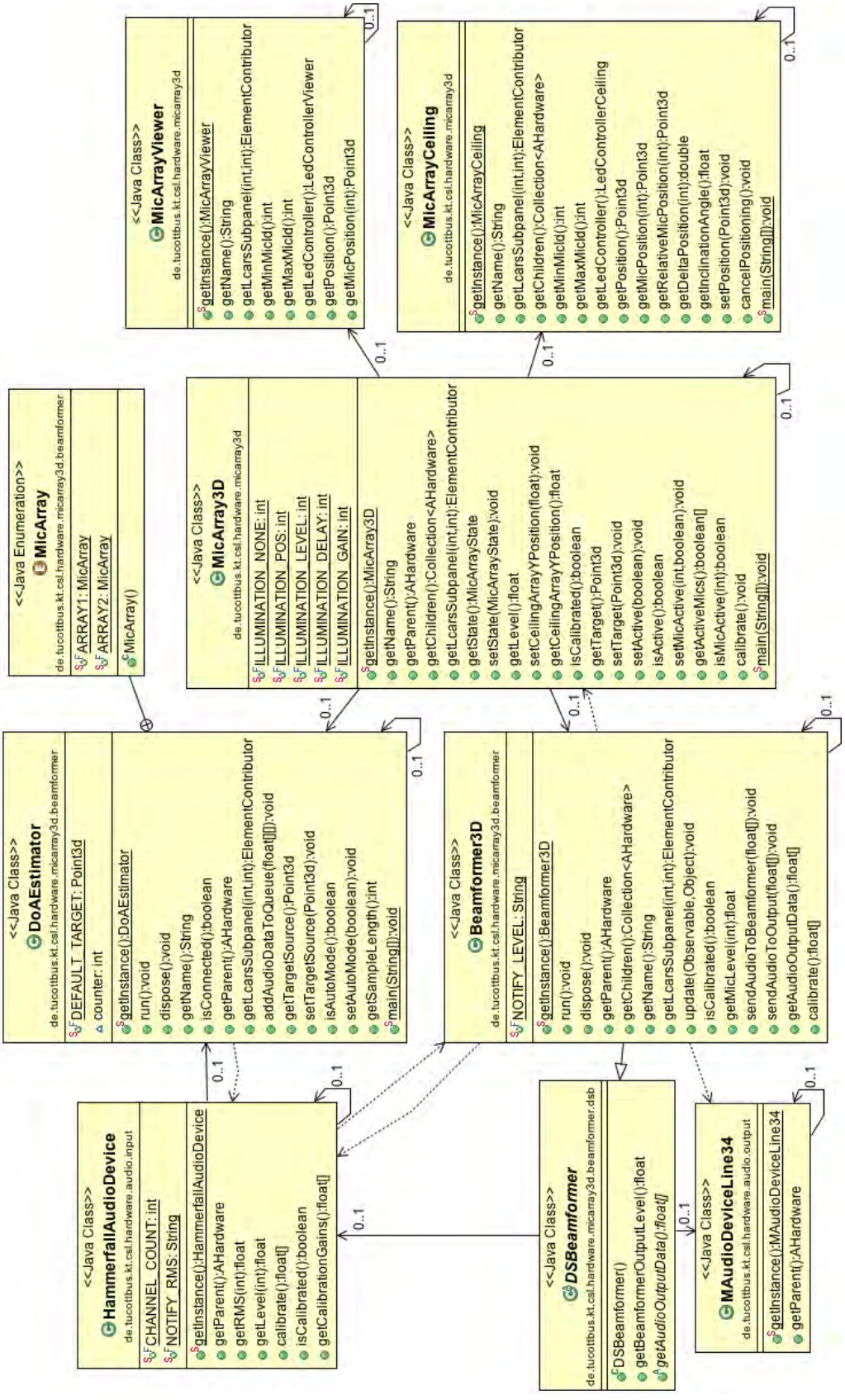


Abbildung 26: UML-Klassendiagramm von **Beamformer3D** und den dazugehörenden Klassen

A.4.3 DoAEstimator UML-Klassendiagramm

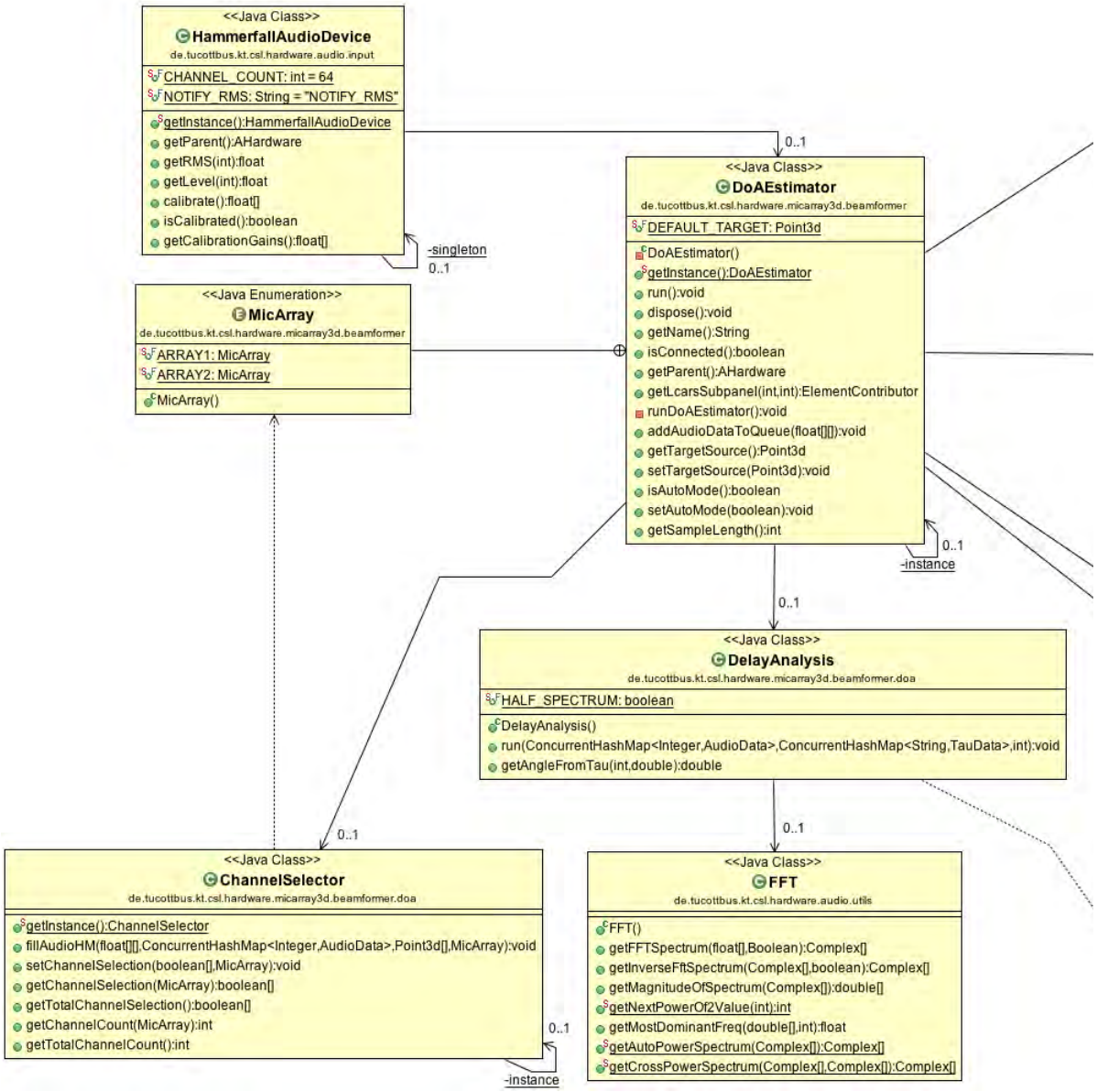


Abbildung 27: UML-Klassendiagramm der Klasse **DoAEstimator** und allen dazugehörenden Unterklassen (Teil 1)

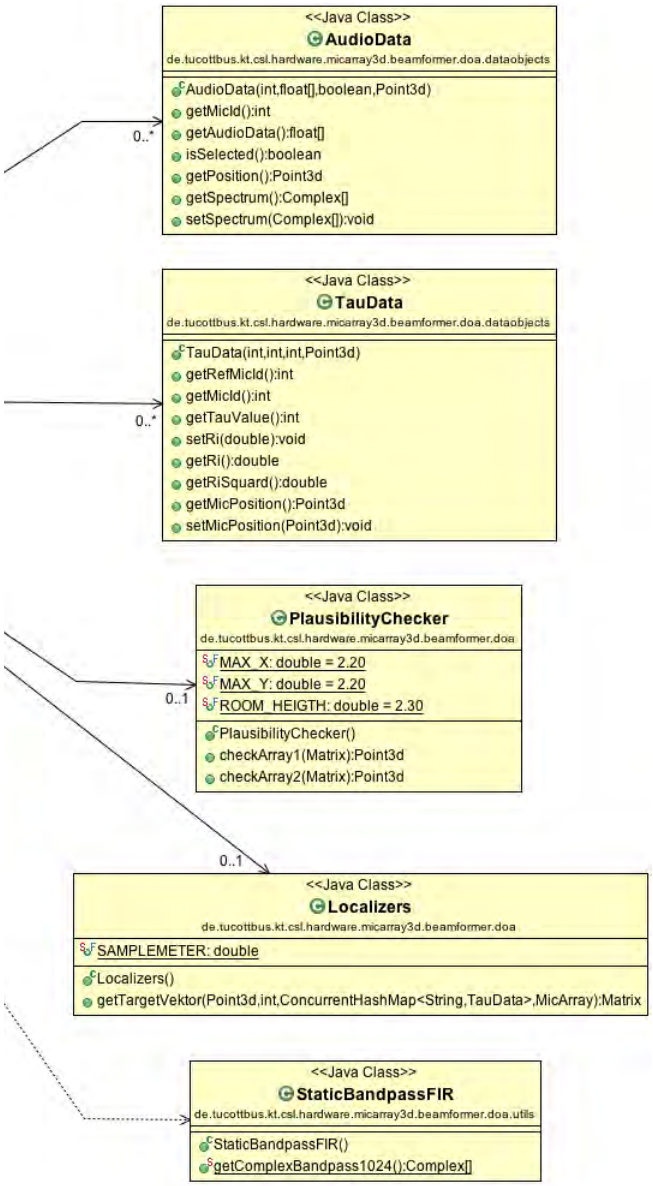


Abbildung 28: UML-Klassendiagramm der Klasse **DoAEstimator** und allen dazugehörenden Unterklassen (Teil 2)

A.4.4 UML-Klassendiagramm des GCC Testprogramms

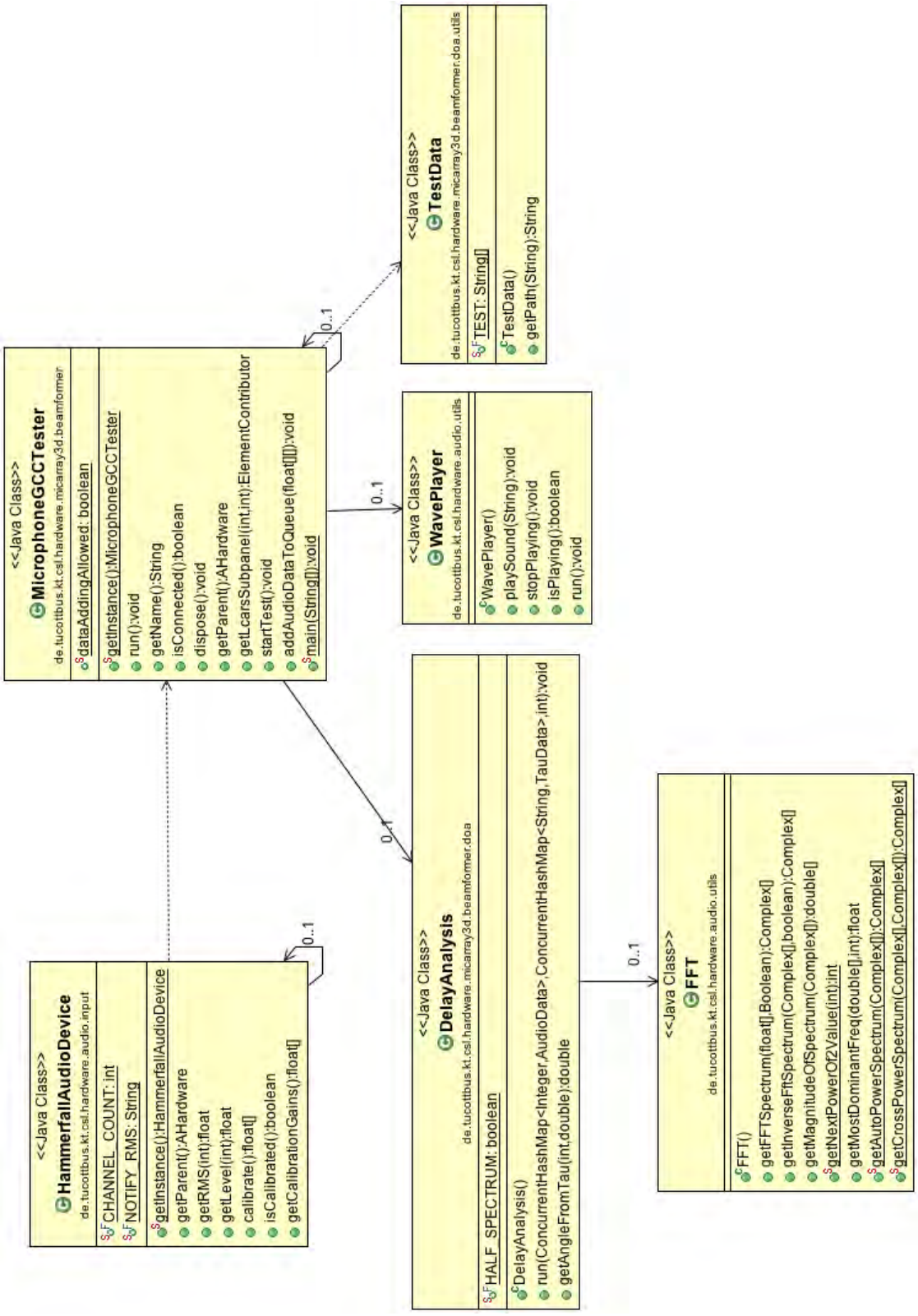


Abbildung 29: UML-Klassendiagramm vom GCC Testprogramm **MicrophoneGCCTester** und den dazugehörenden Unterklassen

A.4.5 SensitivityPlot UML-Klassendiagramm

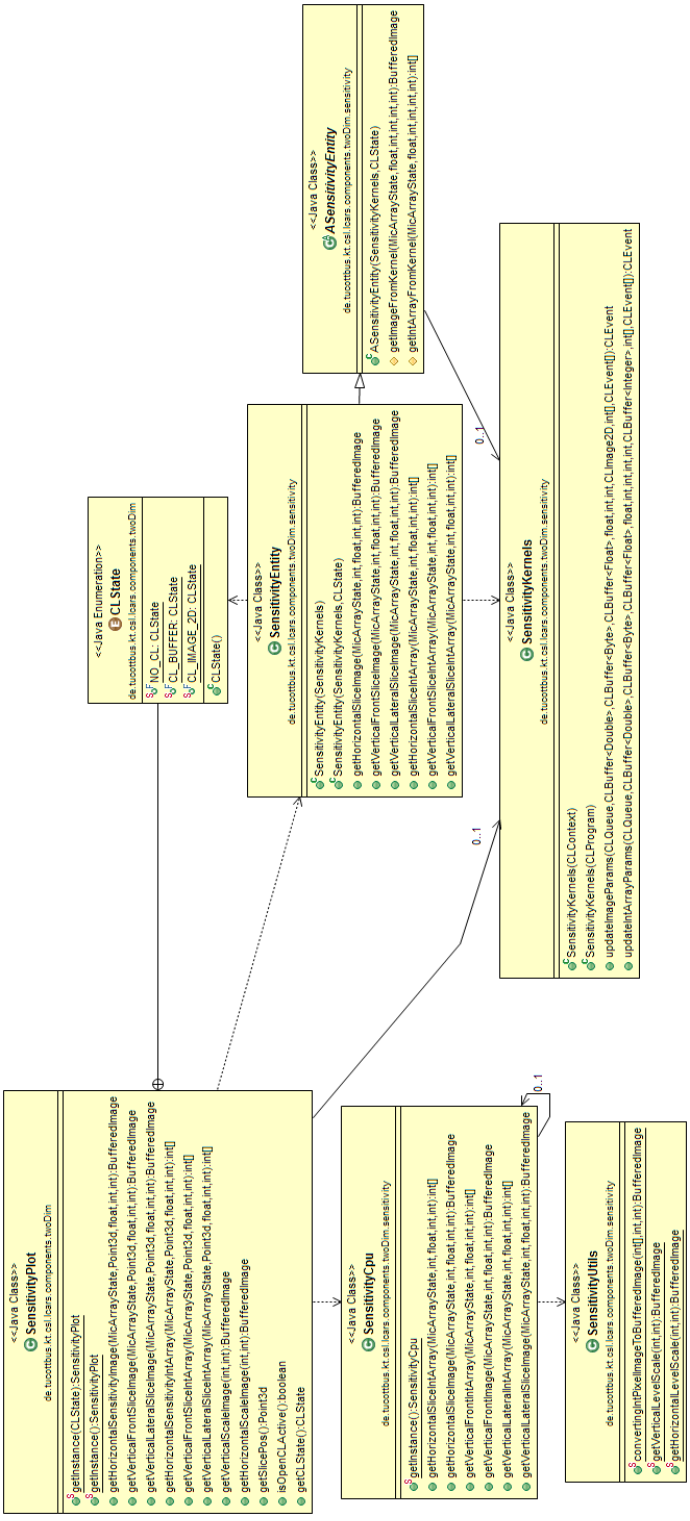


Abbildung 30: UML-Klassendiagramm der Klasse **SensitivityPlot** und allen dazugehörenden Unterklassen

A.4.6 BeamformerPanel UML-Klassendiagramm

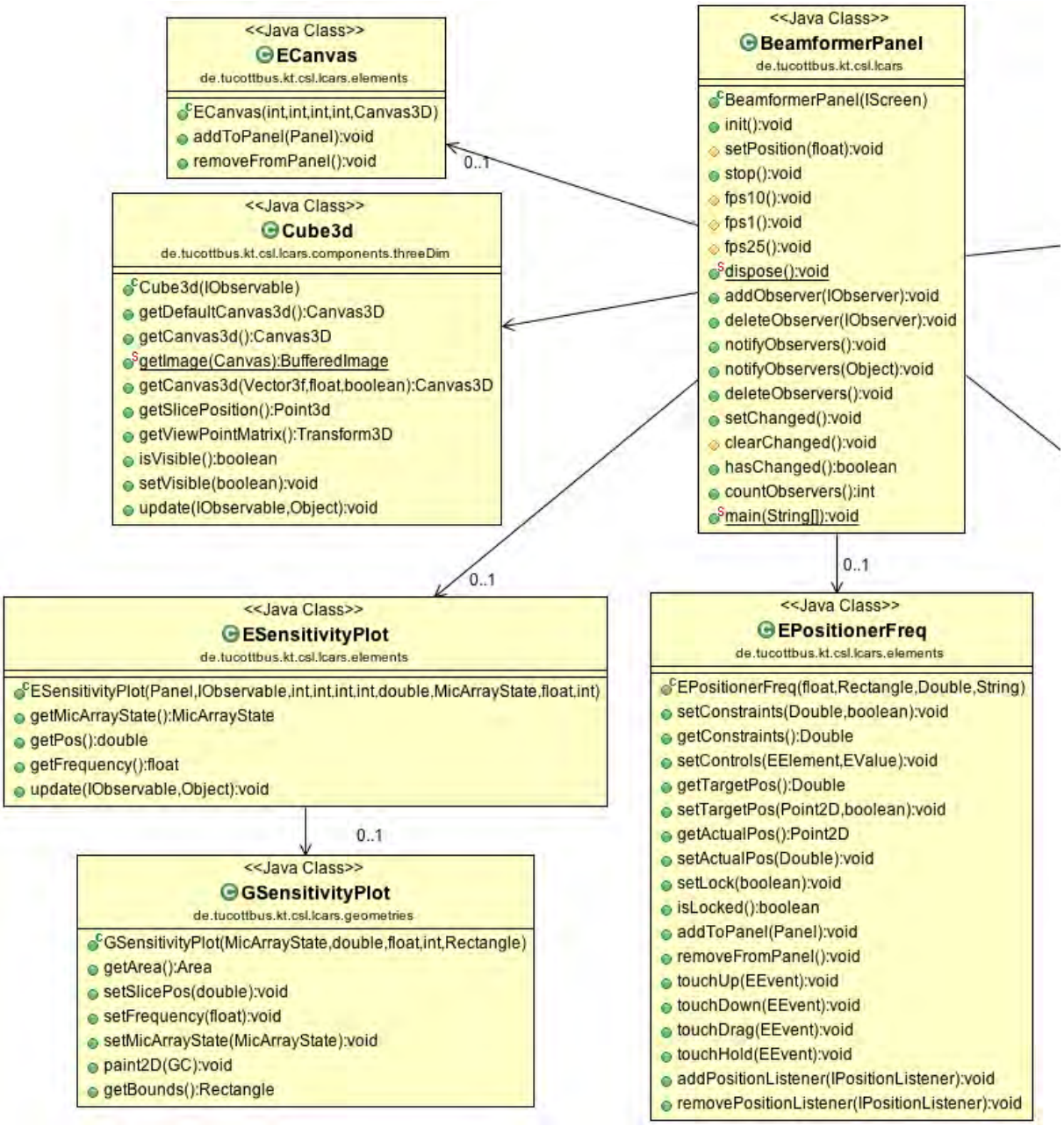


Abbildung 31: UML-Klassendiagramm des **BeamformerPanels** und den zugehörigen Unterklassen (Teil 1)

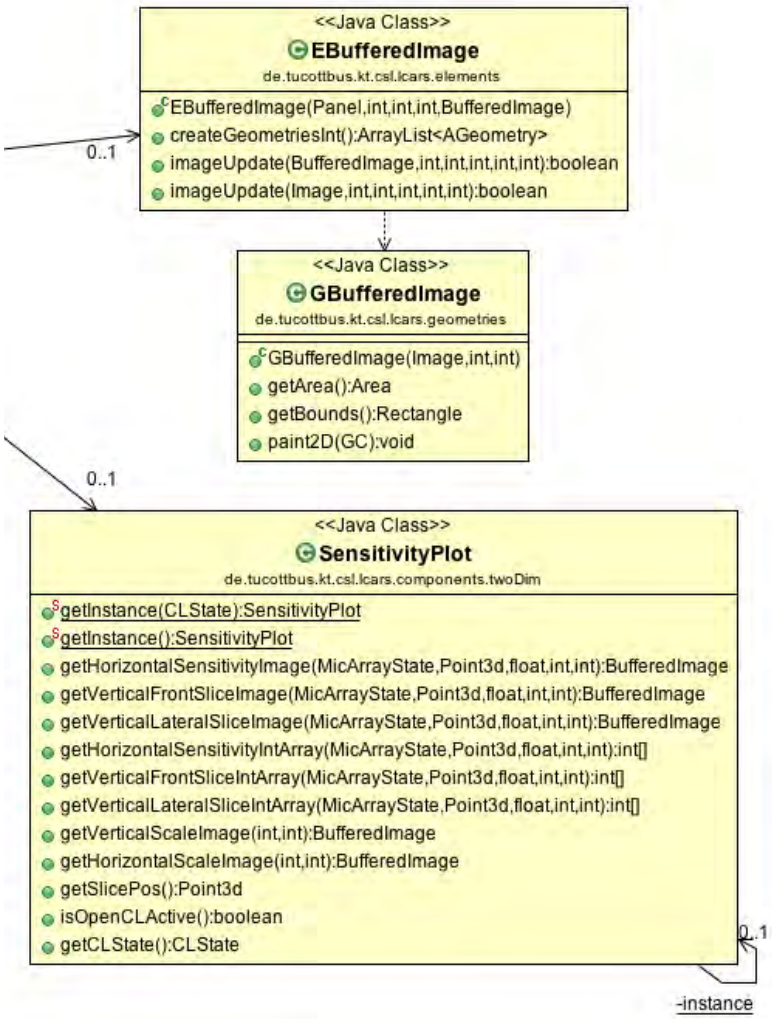


Abbildung 32: UML-Klassendiagramm des **BeamformerPanels** und den zugehörigen Unterklassen (Teil 2)

A.4.7 Cube3d UML-Klassendiagramm

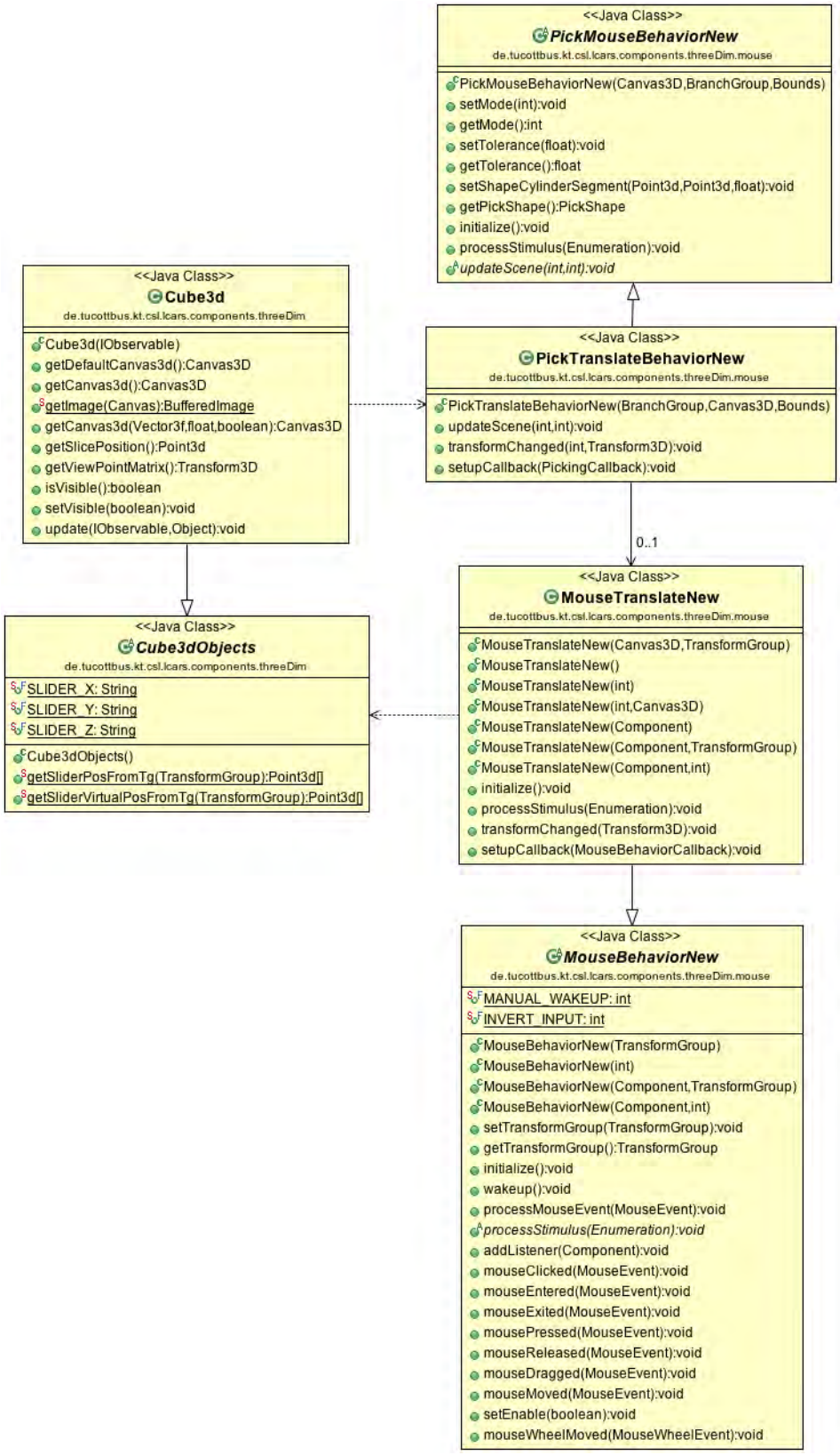


Abbildung 33: UML-Klassendiagramm der **Cube3d**-Hauptklasse und allen dazugehörenden Unterklassen

Selbstständigkeitserklärung

Der Verfasser erklärt, dass er die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt hat. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Cottbus, den 12. März 2016 .....

(Unterschrift)