Kurzanleitung

3D-Speaker-Tracking

Mathias Buder

Sprecherlokalisierung eines Sprechers im Raum unter Verwendung eines kugelförmigen Mikrofonarrays.

INFO

Dieses Dokument spezifiziert die Bedienungsvorschrift zur Sicherstellung einer fehlerfreien Programmfunktion. Im Folgenden werden die verwendeten Programme und deren Versionen für Simulation, DSP-Implementierung sowie Anzeige genannt. Anschließend erfolgt eine genau Erläuterung der zur Verfügung stehenden Programmoptionen.

1. Systemaufbau

Das in Abb. 1 dargestellte Blockschaltbild illustriert alle nötigen Komponenten des Mikrofonarraysystems. Dazu gehören:

1. Mikrofonarray mit acht Mikrofonkapseln
2. Mikrofonvorverstärker (inkl. Netzteil)
3. Digitaler Signalprozessor (D.Module.C6713 + D.Module.PCM3003 + JTAG-Adapter)
4. PC mit serieller Schnittstelle (bzw. USB2Serial-Adapter und max. 2m Datenleitung) inkl. der Anzeige GUI[1]. Diese befindet sich als ausführbare `jar-Datei` im Ordner `04_GUI/GUI_JAR` auf der mitgelieferten CD.

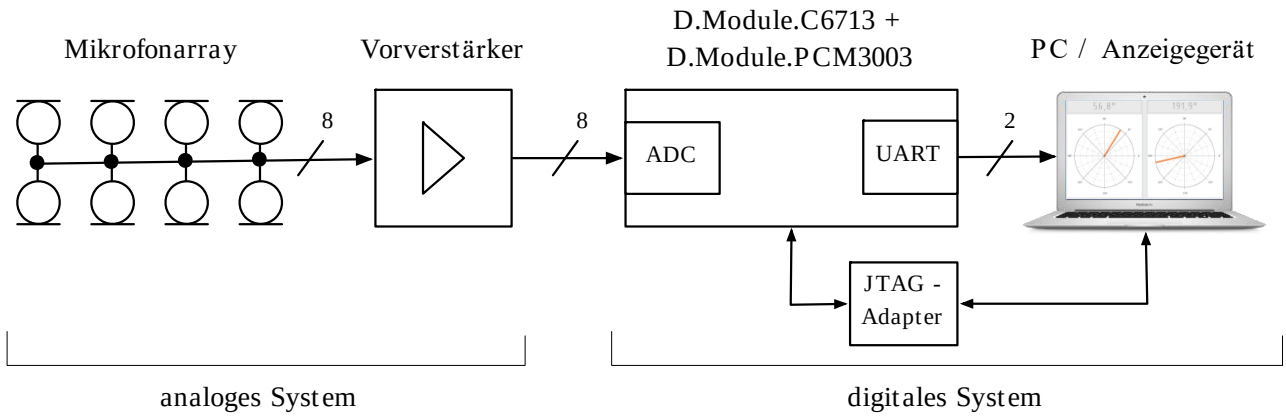


Abb. 1: **Komponenten des Systems**

2. Programmversion

Folgende Programme wurden für die Entwicklung der nötigen Algorithmen eingesetzt:

Typ	Programm	Version
Simulation	MATLAB	R2013a (8.1.0.604)
Echtzeit / DSP	Code Composer Studio	5.1.1.00031
Anzeige	Eclips Standard / SDK	Kepler (20130614-0229)

Tab. 1: verwendete Software

[1] Graphical User Interface (grafische Benutzeroberfläche)

3. Simulation (MATLAB)

Die Simulation in MATLAB gliedert sich in folgende zwei Teilgebiete:

- Erstellung synthetischer Signale
- Simulation des gesamten Algorithmus mit synthetischen oder realen Signalen

Hinweis: *Um eine fehlerfreie Programmfunktion zu gewährleisten, muss der gesamte Ordner* `02_Matlab` *inkl. aller Unterverzeichnissen dem Matlab-Suchpfad hinzugefügt werden!*

3.1 Erstellung synthetischer Signale

Zur Erstellung synthetischer Signale dienen alle Funktionen im Pfad `02_Matlab/Functions/GenerateSyntehticSignal`. Das Programm `BuildSyntheticSignal.m` erzeugt unter Angabe von einer gewünschten Anzahl an Raumkoordinaten ein synthetisches Signale mit einer Dauer von ca. 16sec.

3.1.1 Grundlegende Bedienung

Nach dem Programmstart folgt zunächst die Abfrage aller benötigten Parameter in der Form:

```
Enter Signal-to-Noise-Ratio (DB) [100]:
Please enter number of positions [2]:
                ...
```

Nach Eingabe des gewünschten Wertes kann das Programm durch Betätigen der Enter-Taste fortgesetzt werden. Sollen die in den eckigen Klammern dargestellten Standartwerte genutzt werden muss lediglich die Enter-Taste betätigt werden.

3.1.2 Raumpositionen

Das synthetische Signal simuliert einen Sprecher, der sich durch den Raum bewegt. Dies wird unter Angabe von N Positionsstützstellen

`Please enter number of positions [2]:`

sowie M linear interpolierten Raumpositionen erreicht.

`Please enter number of interpolation points [5] :`

Der Sprecher bewegt sich somit von Positionsstützstelle 1 bis N entlang der M interpolierten Positionen. Je mehr Interpolationsstützstellen gewäht werden, desto mehr Winkel werden eingenommen (feinere Winkelschritte).

3.1.3 Signaltyp

Nach Eingabe der Raumpositionen kann zwischen folgenden Signaltypen gewählt werden:

1. männlicher Sprecher `[(0):Male]`
2. weiblicher Sprecher `[(1):Female]`
3. selbst mit dem Array aufgenommener männlicher Sprecher `[(2):Recorded]`
4. bandbegrenztes weißes Rauschen `[(3):Noise]`

Ist das Programm mit der Signalerstellung fertig, werden die Raumwinkel in tabellarischer Form auf der Konsole ausgegeben und das synthetisch Signal kann abgespeichert werden. Dabei wird die Datei im Format `FileName_SignalType_N-Pos-M-Intpl_SNR-dB.mat` in Pfad `02_Matlab/Functions/GenerateSyntehticSignal/SyntheticFiles` abgelegt. Im Anschluss gibt es die Möglichkeit das Programm zu beenden oder ein weiteres synthetisches Signal mit dem gleichen Bewegungsverlauf aber einem anderen Signaltyp zu erstellen.

3.2 Simulation des gesamten Algorithmus mit synthetischen oder realen Signalen

Die Simulation des gesamten Algorithmus wird mit dem Programm `Simulation.m` durchgeführt. Nach dem Programmstart erfolgt erneut die Eingabe der benötigten Programmparameter. Die Bedeutung der einzelnen Parametereinstellungen ist in Tab. 2 aufgeführt:

Parameter	Funktion	Standardwert
`Show debug plots`	während der Simulation werden die Funktion R sowie das Histogramm angezeigt	(0) : Nein
`Please select file type`	Auswahl, ob Simulation mit synthetischem oder aufgenommenem Signal Das Programm greift dabei wahlweise auf den Inhalt der folgenden Ordner zu: synthetische Signale: `02_Matlab/Functions/GenerateSyntehticSignal/SyntheticFiles` reale Signale: `02_Matlab/TI_DSP/Recorded`	(0) :synthetisch
`Select simulation file:`	Auswahl der gewünschten Simulationsdatei. Nach der Auswahl erscheint ein Plot, in dem der Bewegungsverlauf dargestellt wird.	(1): Datei an erster Stelle
`Enter energie limit [dB]`	Angabe des Schwellenwertes für die Blockenergie	-10 dB
`Enter number of simulations`	Angabe, wie viele Blöcke simuliert werden sollen	maximale Anzahl
`Enter histogramm buffer length`	Angabe, wie viel Ergebnisse im Histogrammringspeicher abgelegt werden sollen.	50 Werte

Parameter	Funktion	Standardwert
`Enter histogramm threshold in percentage`	Schwellenwert, ab dem ein Wert als wahrscheinlich eingestuft wird. Dieser richtet sich nach der Histogrammringspeicherlänge und ist in % anzugeben	20 %

Tab. 2: einstellbare Parameter der MATLAB Simulation

4. Echtzeit (DSP)

Das DSP-Programm lässt sich durch die in Tab. 3 aufgeführten Parameter steuern. Diese befinden sich in der Datei `setup.h` wobei gilt ist das `#define` gesetzt, arbeitet das Programm in diesem Modus und alle dafür nötigen Programmteile werden vom Compiler übersetzt.

Hinweis: *Vor Programmstart ist sicherzustellen, dass die Abtastfrequenz auf 48kHz eingestellt ist (Dipschalter am linken unteren Rand des DSP-Aufbaus).*

Das Programm kann wie dargestellt in zwei Varianten ausgeführt werden:

Variante 1: Das Programm läuft im C-Mode. Alle hardwarespezifischen Elemente werden vom Compiler ignoriert. Als Quellsignal dient anstelle des ADC-Speichers eine H-Datei. Diese befindet sich im Ordner `H_Sim_Files` und enthält ein Array mit Test-Abtastwerten.

Variante 2: Das Programm läuft im DSP-Mode. Alle hardwarespezifischen Elemente werden vom Compiler übersetzt. Als Quellsignal dient der ADC-Speicher `adcbuffer`.

Des weiteren können vom Benutzer zwei Parameter (Variable) während der Programmlaufzeit veränder werden um das System den gegebene Raumbedingungen anzupassen. Diese sind:

- Variable `energielimit (beliebig [Standard:40])`
- Variable `int16_HistogramThreshold (0 bis 30)`

Zur Modifikation dieser Variablen müssen diese während einer aktiven Debug-Session dem CCS2-Watchwindow hinzugefügt werden.

[2] Code Composer Studio

#define	Funktion	Kommentar	Variante	
			1	2
C_MODE	Programm läuft im C -ONLY Modus	Alle hardwarespezifischen Programmteile werden entfernt	x	
PROFILE_MODE	Messung der Programmlaufzeit	Pinbelegung siehe Tabelle unten		x
UART_MODE	Senden der Ergebnisse	Die geschätzten Raumwinkel werden via UART-Schnittstelle versandt		x
ADC_LOOP_THROUGH_MODE	Verifikation der Echtzeit	Durchschleifen des ADC (Kanal 1) zum DAC (Kanal 1).		x
HISTOGRAM_MODE	Ergebnisglättung	Die Ergebnisse werden einer Histogrammschätzung unterzogen	x	x
SEARCH_OPT_MODE	geschwindigkeitsoptimierte Raumsuche	Absuchen des Raumes in variablen Schrittweiten. **Muss zum Erfüllen des Echtzeitkriteriums aktiviert sein!**	x	x

Tab. 3: einstellbare Parameter des DSP-Programms

Ist der PROFILE_MODE aktiviert, lässt sich das Zeitverhalten der implementierten Methoden unter Verwendung eins Oszilloskops an folgenden Kontakten der Leiste T messen. Zu beachten ist, dass zum Aktivieren der GPIO‘s der Jumper JPH1 auf Position a des D.Module.6713 gesetzt werden muss (siehe dazu Dokument ugd6713.pdf in Pfad 04_Dokumente/02_Datenblaetter/ auf der CD)

Pin (Leiste T)	C-Funktion
23	EDMA
24	Copy2CmplxStruct
25	CalcVariance
26	FastCrossCorrelation
27	SearchAndFind
28	CreateHistogram
29	sendString
30	gesamten Algotithmus (ohne sendString)

6. Anzeige (GUI)

Abb. 2 zeigt den Aufbau der graphischen Benuteroberfläche. Diese kann wahlweise innerhalb der Eclips Entwicklungsumgebung gestartet werden (dazu muss der Ordner `04_GUI/GUI_WORKSPACE` auf der mitgelieferten CD als Eclips-Workspace ausgewählt werden) oder durch Ausführen der kompilierten `JAR-Datei` im Ordner `04_GUI/GUI_JAR`. Die Ordnerstruktur innerhalb von `GUI_JAR` darf nicht verändert werden, da das Programm in diesem Fall nicht auf benöigte Ressourcen zugreifen kann.

Hinweis: *Zur Programmfunktion innerhalb Eclips ist es zwingend notwendig die Bibliotheken `RXTXcomm.jar` (http://rxtx.qbang.org/wiki/index.php/Using_RXTX) sowie `jmatio.jar` (http://www.mathworks.com/matlabcentral/fileexchange/10759-jmatio-matlabs-mat-file-io-in-java) entsprechend deren Installations-hinweisen einzubinden.*

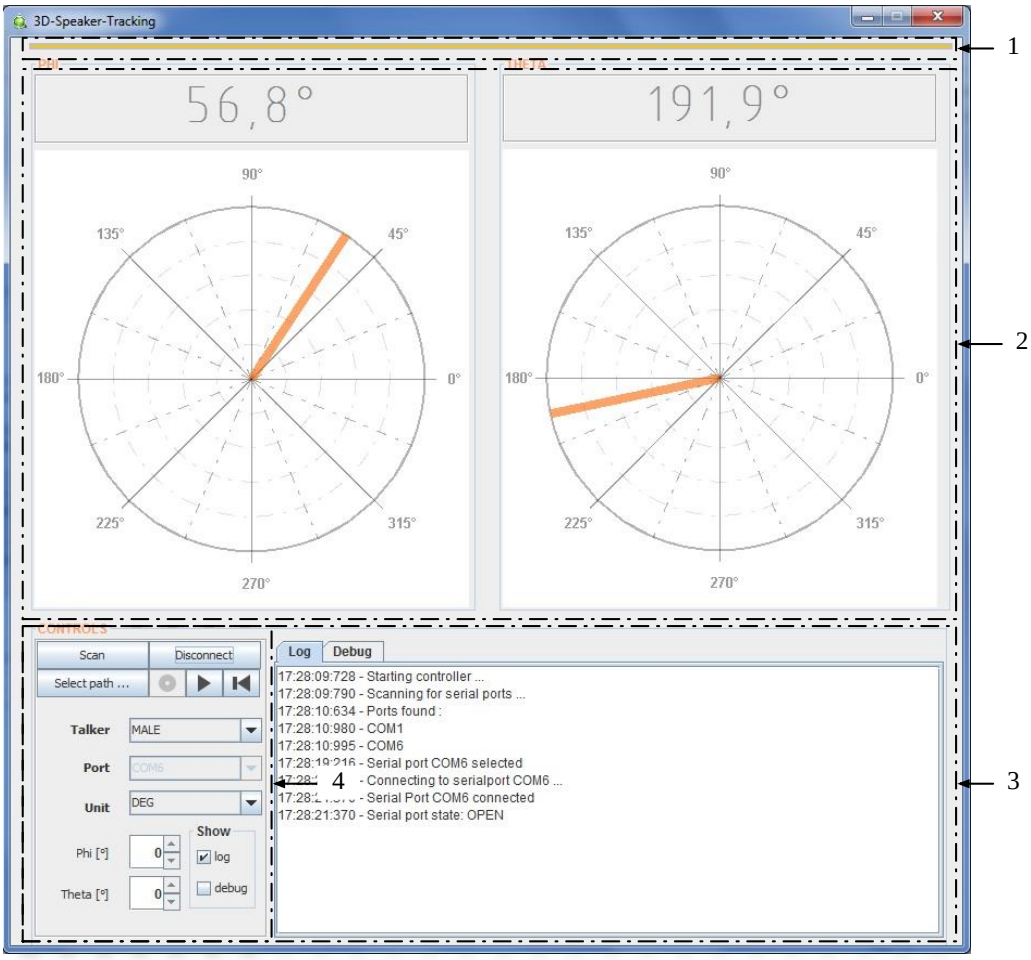
Die Oberfläche gliedert sich, wie dargestellt in vier Hauptbereiche:

1. Daten-LED – Diese beginnt grün/orange zu blinken bei Empfang serieller Daten.

2. Winkelanzeige – Zeigt die aktuellen Raumwinkel an.

3. Log- und Debug-Nachrichten – Hier werden dem Benutzer Informationen über den aktuellen Programmstatus mitgeteilt. Auf dem Reiter „Debug“ kann darüber hinaus kontrolliert werden, welche Daten empfangen werden und ob dabei Fehler entstehen. Das Hinzufügen von weiteren Nachrichten kann durch das Deaktivieren der `log` und/oder `debug` Häkchens unterbunden werden.

4. Einstellungen – Hier können vom Benutzer verschieden Einstellungen getroffen werden. Nach Programmstart wird der Computer zunächst nach serielle Schnittstellen abgesucht (Scanning ...). Die verfügbaren Ports werden anschließend im Aufklappmenü „Ports“ aufgelistet. Nach entsprechender Portwahl kann sich durch Betätigen des Connect/Disconnect zu diesem verbunden werden. Das Programm wechselt in den Serial-Port-Status: `OPEN` und wartet auf eintreffende Daten. Des Weiteren verfügt die Software über die Möglichkeit Audiodaten unter Verwendung der Soundkarte abzuspielen und die geschätzten Raumwinkel in eine MAT-Datei abzulegen. Diese kann anschließend mit dem Skript `ArrayMeasuring.m` im Pfad `02_MATLAB/MeasuringFiles` ausgelesen und graphisch dargestellt werden. Der Standart-Speicherort zur Ablage der MAT-Dateien befindet sich im Pfad `04_GUI/GUI_JAR/scr/MAT_Files`. Mit der Option „Select path ...“ kann mittels Dialogfenster ein andere Pfad ausgewählt werden. Anschließend besteht die Möglichkeit im Aufklappmenü „Talker “zwischen einer männlichen und einer weiblichen Stimme zu wählen. Durch Drücken der Play-Taste (Dreick) kann die Audiodatei vorgehört und ggf. das Array korrekt ausgerichtet werden. Bevor die Aufnahme durch betätigen der Record-Taste (Kreis) beginnt müssen die theoretischen Winkel Phi und Theta in die dafür vorgesehenen Felder eingetragen werden. Diese Werte werden anschließend zur Namensgebung der MAT-Datei verwendet. Die Aufnahme stoppt automatisch und die MAT-Datei wird erstellt wenn das Ende der Audiodatei erreicht wird. Die MAT-Datei wird im Format `dd-mm-yy_hh-mm-ss_Talker_Phi_PhiVal_Theta_ThetaVal.mat` im ausgewähten Verzeichnis abgespeichert (Bsp.: `12-08-13_22-08-22_MALE_Phi_45_Theta_0`). Zum Auslesen der MAT-Datei muss diese in das Verzeichnis `02_MATLAB/MeasuringFiles` kopiert werden. Nach dem Start des o.g. Skipts werden die zur Verfügung stehenden MAT-Dateien zur Auswahl aufgelistet.



Erforderliche Parameter fehlen oder sind falsch.