**Projektplan Gruppe B: 3D Audio**

Gruppenmitglieder: Felix Pfreundtner, Matthias Lederle, Huaijiang Zhu, Molka Elleuch, Manuela Heiss

**Projektvision:**

„Wir wollen ein Konzert simulieren, das ein Hörer auf seinem Kopfhörer wahrnimmt, als würde er „inmitten der Band“ sitzen!“

**Projektbeschreibung:**

Was kann die Abgabeversion:

- Simulation von bis zu 6 Musikern beziehungsweise eines 5.1 Lautsprechersystems

- Hörer kann frei im Raum bewegt werden

- keine Berücksichtigung der Raumkaustik, Abstrahlung der Lautsprecher unter Freifeldbedingungen

- Anordnung der Lautsprecher direkt auf Hörer gerichtet

- Lautsprecher drehen sich bei Bewegung des Hörers automatisch auf den Hörer zu

- x,y Position des Hörers und Speaker kontinuierlich (Realtime) in GUI einstellbar, Höhe konstant

- Berücksichtigung der horizontalen Kopfbewegungen durch einen Headtracker

**Entwicklungsplan/Zeithorizont:**

**Woche 1: bis Dienstag, 19. Mai:**

* Icebreaking / Teambuilding
* Projektplanerstellung
* Einarbeitung in Audiosignalverarbeitung, HRTF (KEMAR), GUI Programmierung, HeadTracker
* Mindestliteratur für alle: Auralization (Vorländer, 2008), Kapitel 6.3, 7, 9, 15.1, 16.1

**Woche 2 + 3: bis Di, 2. Juni:**

|  |  |
| --- | --- |
| Signalverarbeitungsalgorithmus Team | GUI & Headtracker Team |
| 1. Schritt   * Programmieren einlesen KEMAR HRTF Dateien * Programmieren einlesen eines mono Wave Files * Signalverarbeitung des Eingangssignals zu einem binauralen Ausgabesignal abhängig von den Paramatern horizontaler Hörer-Quelle Winkel und Hörer-Quelle Abstand * Widergabe des generierten Ausgabesignals   2. Schritt:   * Einbeziehen von Realtimeparameterveränderungen * Nach jedem Zeitintervall Δt (Länge eines Fourierblocks) für jede Quelle Auswerten der von GUI gesendeten Paramater horizontaler Hörer-Quelle Winkel und Hörer-Quelle Abstand * Anpassen des binauralen Ausgabesignals zum Zeitpunkt t anhand der aktualisierten Parameter | * Programmierung einer 2D GUI * 6 Quellen und 1 Hörer frei durch Mausziehen bewegbar * Anzahl an Quellen über +/- Button einstellbar * Ausgangsanordnung der Quellen entspricht typischer Heimkinoanordnung, Reset Button stellt Ausgangszustand wieder her * Design ähnlich User-Interface Asus Xonar Soundkarte:      * Raumgröße: Wohnzimmer * Pfad des Eingabesignals jeder Quelle in GUI auswählbar * Quellen und Hörer sind Werte in kartesischen Koordinaten, Ursprung des Koordinatensystems ist der Hörer * Blickrichtung des Hörers und Ausrichtung der Quellen sichtbar machen * Playbutton: Berechnen der Parameter horizontaler Hörer-Quelle Winkel und Hörer-Quelle Abstand, Übergabe der Paramater in Polarkoordinaten an Signalverarbeitungsalgorithmus * Im Zeitintervall Δt: Bei Verschiebung der Quellen oder des Hörers Neuberechnung der Parameter: horizontaler Hörer-Quelle Winkel und Hörer-Quelle Abstand, Weitergabe der Daten an den   Signalverarbeitungsalgorithmus |

**Woche 4+5: bis Di, 16. Juni:**

|  |  |
| --- | --- |
| Signalverarbeitungsalgorithmus Team | GUI & Headtracker Team |
| * Erweiterung des Algorithmus von einer Quelle auf 6 Quellen * Falls notwendig: Laufzeitoptimierung des Algorithmus * Normalisieren der Lautstärker aller Quellen * Mischen der einzelnen binauralen Ausgabesignale zu einem einzigen Ausgabesignal unter Berücksichtigung des Abstandes der Quellen zum Hörer | * Integration des Headtrackers in das GUI * Einlesen der relativen horizontalen Winkelveränderungen des Kopfes * Einbeziehen der Winkelveränderungen in den Polarwinkel zur Quelle * Anzeigen der Kopfausrichtung im GUI |

**Woche 6: bis Di, 23. Juni:**

* Tests + Präsentation erstellen

**Woche 7: bis Di, 30. Juni:**

* Zeitpuffer

**Woche 8: bis Di, 7. Juli: Präsentation**

**Interne / externe Schnittstellen:**

Vom Signalverarbeitungsalgorithmus zum GUI:

* Keine Parameterübergabe

Vom GUI zum Signalverarbeitungsalgorithmus:

* Bei Drücken des Playbuttons und Veränderungen des Hörers oder der Quelle: aktualisieren der Paramater horizontaler Hörer-Quelle Winkel und Hörer-Quelle Abstand in Polarkoordinaten
* Dateipfad zum wave Eingangssignal jeder Quelle

Vom Headtracker zum GUI:

* Horizontale Winkelveränderung des Hörers im Verhältnis zur Ausgangslage t=0 des Kopfes

Vom Benutzer zum GUI:

* Anzahl der Quellen über +/- Button
* Pfad zum wave Eingangssignal der Quellen
* x,y Position der Quellen (durch Verschieben im GUI)
* Reset Button stellt Ausgangsanordnung der Quellen wieder her

Main Funktion:

* Die Main Funktion startet das GUI
* Über das GUI werden vom Benutzer alle notwendigen Eingabeparameter eingegeben
* Danach ruft das GUI den Signalverarbeitungsalgorithmus auf
* Der Signalverarbeitungsalgorithmus generiert zum Zeitpunkt t=0 eine binaurale Soundausgabe über die vom GUI berechneten Parameter horizontaler Hörer-Quelle Winkel (Auswahl der richtigen HRTF für jede Quelle) und Hörer-Quelle Abstand (Anteil einer Quelle an Klangmischung)
* Nach jedem Fourierblock prüft der Signalverarbeitungsalgorithmus ob sich die vom GUI gesendeten Parameter horizontaler Hörer-Quelle Winkel und Hörer-Quelle Abstand verändert haben, falls ja wird eine andere HRTF mit dem Quellensignal gefaltet bzw. der Anteil der Quelle an der Klangmischung angepasst

**Gruppen Einteilung:**

- Signalverarbeitungsalgorithmus Team: Felix Pfreundtner, Matthias Lederle, Huaijiang Zhu

- GUI & Headtracker Team: Molka Elleuch, Manuela Heiss

**Fragen an unseren Betreuer:**

Wie kann man die horizontale Winkelausgabe des LDV Headtracker auslesen? Ist hierfür Literatur vorhanden?

Ist das Auswerten von horizontalen Veränderung der Quellen- und Hörerposition für das Projekt ausreichend oder sollten wir zusätzlich eine vertikale Dimension mit einbeziehen?

Kann der Signalverarbeitungsalgorithmus auch nicht objektorientiert programmiert werden?

Ist es nötig für den räumlichen Eindruck Hall zu den Audiosignalen hinzuzufügen?

Ist in Python das Einlesen einzelner Frames eines Audiosignals möglich oder muss stets das gesamte wave Signal eingelesen werden?

Wie können Soundausgabeprobleme (Ruckler) zwischen zwei, mit unterschiedlichen HRTF gefalteten, Fourierblöcken am besten verhindert werden?

Gibt es Möglichkeiten den FFT Algorithmus beziehungsweise die Faltung mit den HRTF zu beschleunigen? Ist nötig bei 6 Eingangssignalen und Realtime Hörer-Quelle Winkel Veränderungen auf diese Möglichkeiten zurückzugreifen?