# Qualifying Exam (Oral)

Gabriel Zalles Ballivian

# G Zalles

- Phd Computer Music UCSD
- MA Music Tech NYU
- BA Music UCSD (ICAM program)

# Overview of this presentation

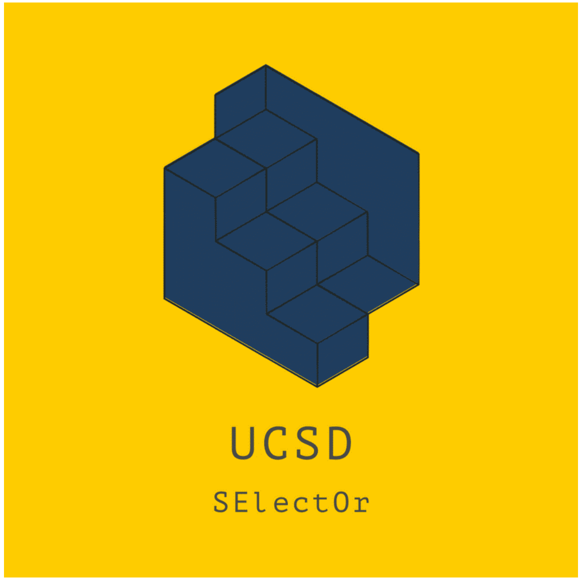
- **Ch 1: Free and Open Source Spatial Audio Tool-kits (Q3)**
  - Intro, ambiX, IEM Plug-in Suite, SPARTA, conclusion.
- **Ch 2: Low-cost Microphone Array Design and Calibration (Q2)**
  - Intro, Middlicott & Wiggins, MSMA, HOSMA, SpHEAR, z-array, conclusion.
- **Ch 3: Equitable Distribution of Spatial Music Using WebXR (Q1)**
  - Intro, Resonance, JSAmbisonics, HOAST, WHAM, “Pigments of Imagination”, conclusion.

# Background

- Ch 1: Free and Open Source Spatial Audio Tool-kits (Q3)
  - Create spatial music: **UCSD SElectOr**
- Ch 2: Low-cost Microphone Array Design and Calibration (Q2)
  - Record spatial music: **z-array**
- Ch 3: Equitable Distribution of Spatial Music Using WebXR (Q1)
  - Explore XR: **Synth-esthesia**

# Background

- Ch 1: Free and Open Source Spatial Audio Tool-kits (Q3)
  - Create: **UCSD SElectOr**
    - **Year one: improvisation ensemble.**
    - **Year two: collaborations and concerts.**
    - **Year three: collaborative project (POI).**

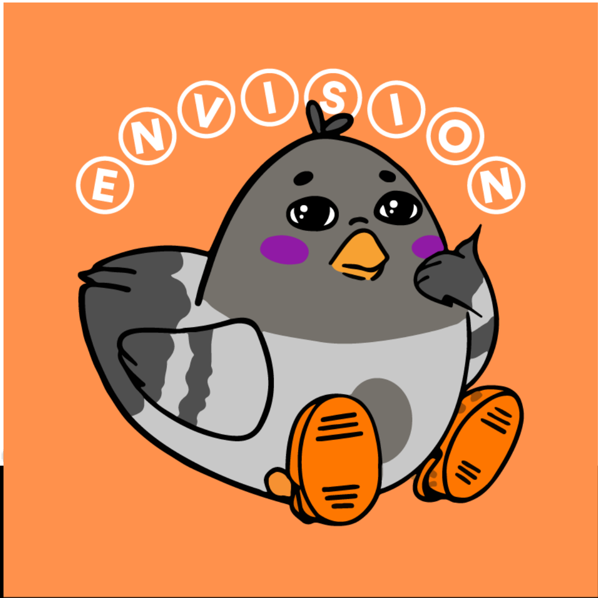


3OA in CPMC122

# Background

- Ch 2: Low-cost Microphone Array Design and Calibration (Q2)
  - Record: **z-array**
    - FOA w/ MEMS (AES147) objective measurements.
    - Recorded stimuli for FOA subjective experiment + got IRB approval.
    - Built 8-channel prototype with MEMS to test calibration.

PROJECT IN A BOX



# Background

- Ch 3: Equitable Distribution of Spatial Music Using WebXR (Q1)
  - Explore XR: **Synth-esthesia**
    - Unity music project w/ HTC Vive
    - Collaboration with undergraduates.
    - Demo.

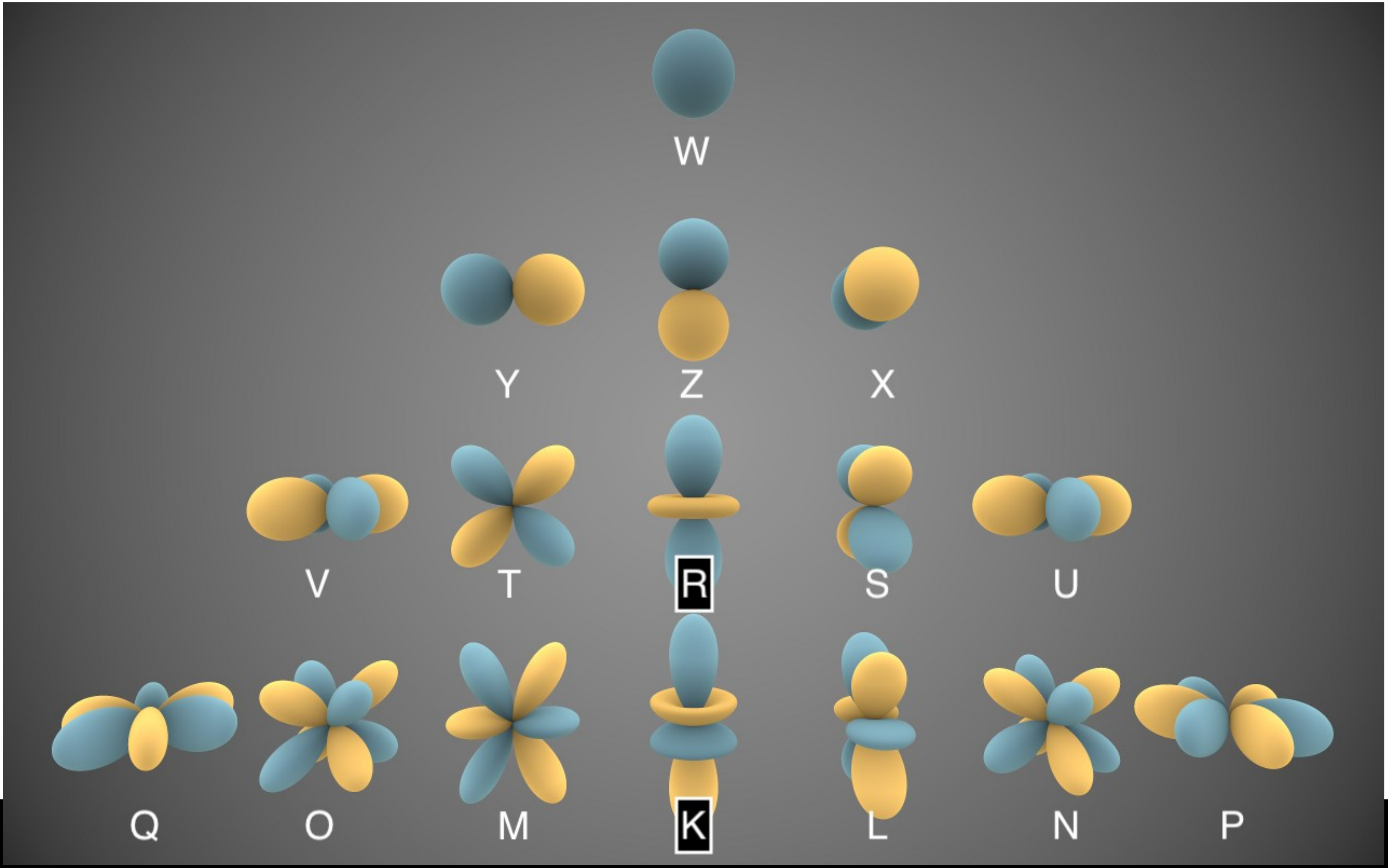
TRITON XR

# Background

## Synth-esthesia

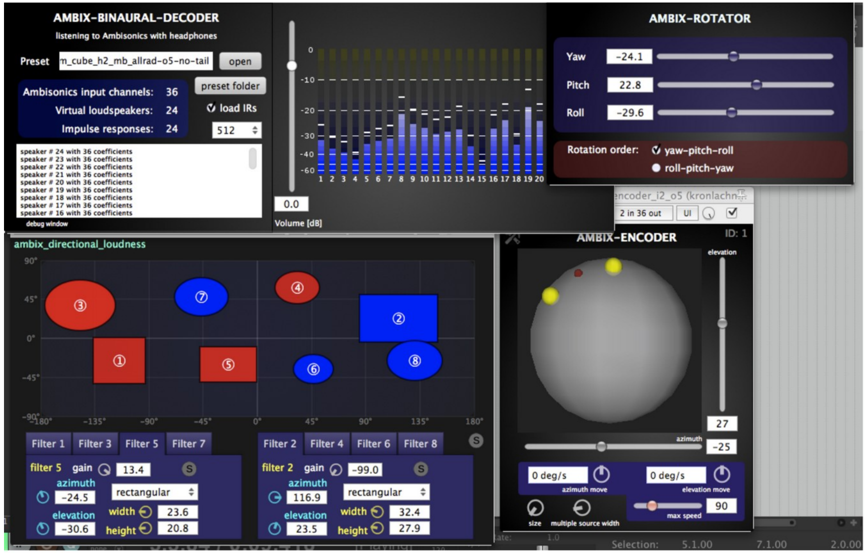
- Re-focus on low-cost, open-source, accessible and equitable software/hardware.
  - Inspirations: MSP & FLL (CCRMA).
- POI: A-frame VR project w/ undergrads (WebXR).
- Submitted to Audio Mostly conference as music submission.

# ambisonics



# Chapter 1 - Question

*Accessible spatial audio toolkit for composition, performance, sound design and instruction.*



Preset: m_cube_h2_mb_allrad-o5-no-tail

Ambisonics input channels: 36
Virtual loudspeakers: 24
Impulse responses: 24

☑ load IRs

512

speaker # 24 with 36 coefficients
speaker # 23 with 36 coefficients
speaker # 22 with 36 coefficients
speaker # 21 with 36 coefficients
speaker # 20 with 36 coefficients
speaker # 19 with 36 coefficients
speaker # 18 with 36 coefficients
speaker # 17 with 36 coefficients
speaker # 16 with 36 coefficients

Volume [dB]: 0.0

Yaw: -24.1
Pitch: 22.8
Roll: -29.6

Rotation order: ☑ yaw-pitch-roll ● roll-pitch-yaw

2 in 36 out

ID: 1

elevation: 27
azimuth: -25

0 deg/s azimuth move
0 deg/s elevation move
max speed: 90

filter 5 gain 13.4
azimuth -24.5
elevation -30.6
rectangular
width 23.6
height 20.8

filter 2 gain -99.0
azimuth 116.9
elevation 23.5
rectangular
width 32.4
height 27.9

iem

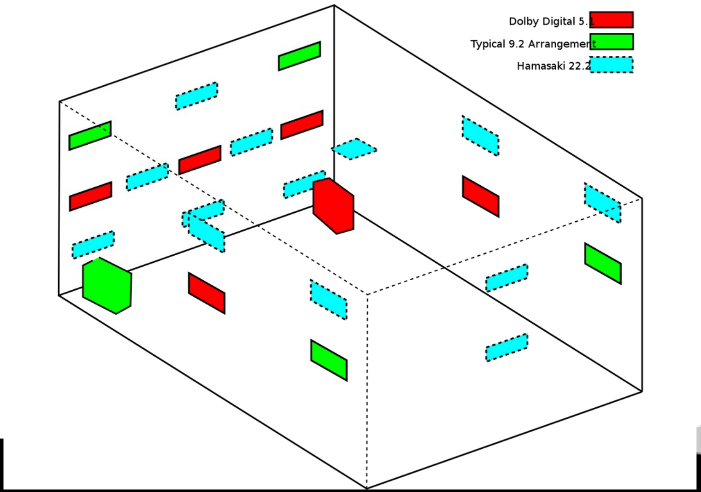
Aalto University
School of Electrical Engineering

# Chapter 1 - Intro

- *Discuss what is needed for such a toolkit, and what makes this toolkit.*
  - **Cross-platform**
  - **Free**
  - **Open source**
  - **Well documented**
  - **Well supported**
  - **Intuitive**

# Chapter 1 - Intro

- *What makes this toolkit?*
  - **Binaural decoding + head-tracking**
- What about surround sound, OBA, WFS, XTC, VBAP, etc.?



# Chapter 1 - Intro

- *Additional questions:*
  - Can these toolkits be used for live performance?
  - How do these toolkits work?
  - What has **not** been done yet?

# Chapter 1: AmbiX + MCFX

- AmbiX
  - FOSS, comprehensive, intuitive, a few unique effects.
    - **Directional loudness**
    - **Warp**
    - **Widening**
  - Utilities:
    - Gain + Delay (calibration) [IEM (Distance Compensator) easier]
    - Delay (sync w/ remote feed [telematics] or video).
  - Encode (OSC), decode, binaural, etc.
  - Publications :)

# Chapter 1: AmbiX (Dir. Loudness/Warping)

- T-design for sampling SH domain (*Hardin and Sloane - 1996*).
- Uniform sampling allows for direct integration.
- Steps:
  - **b** = normalize (4π/L), SH transposed “decode”, weight (**g**)**,** SH (w/ angle changes) “re-encode”.

$$\boldsymbol{T} = \mathrm{diag}\{\boldsymbol{b}\}\overset{\text{sampling}}{\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)}\,\mathrm{diag}\left\{\boldsymbol{g}\left(\boldsymbol{\Theta}_t\right)\right\}\boldsymbol{Y}\left(\mathcal{T}^{-1}\left\{\boldsymbol{\Theta}_t\right\}\right)$$

Kronlachner’s thesis uses 4π/L.

Paper uses different value.

L is number of nodes in t-design.

# Chapter 1: AmbiX (Dir. Loudness/Warping)

$$\boldsymbol{T} = \operatorname{diag}\{\boldsymbol{b}\}\boxed{\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)}\operatorname{diag}\{\boldsymbol{g}(\boldsymbol{\Theta}_t)\}\boxed{\boldsymbol{Y}\left(\mathcal{T}^{-1}\{\boldsymbol{\Theta}_t\}\right)}$$

- **T** is a (N+1)^2 by (N+1)^2 matrix.
  - **A . T = A’** (where **A** is encoded ambisonic audio)
- **Y^**T . **Y** = **I** (identity matrix)
- T^-1 is the angle transform unto coordinates Θ_t from t-design.
- **g** is the gain compensation (diagonal, <u>after</u> new angles are calculated)

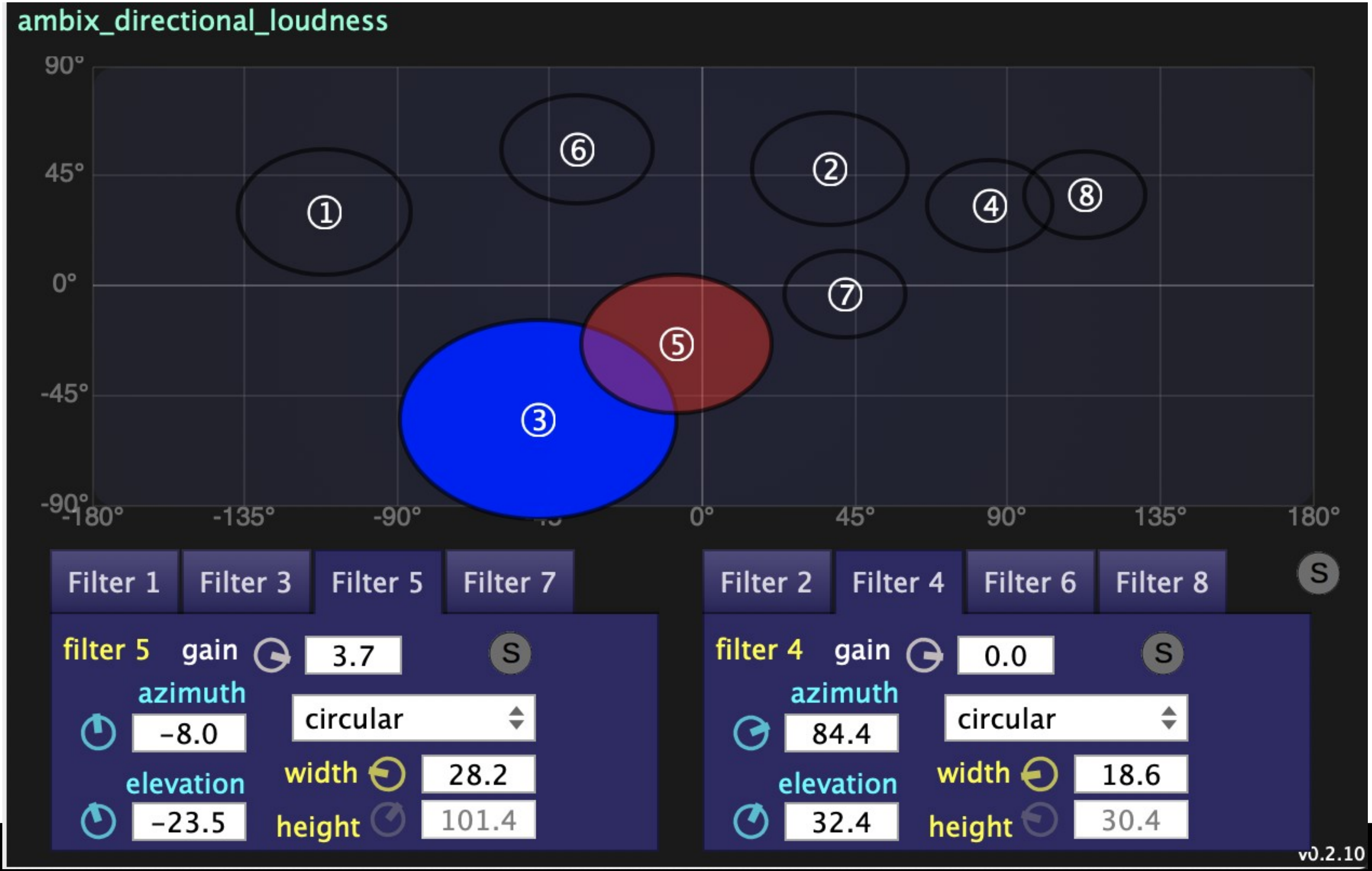
# Chapter 1: AmbiX (Dir. Loudness/Warping)

$$\boldsymbol{T} = \operatorname{diag}\{\boldsymbol{b}\}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)\operatorname{diag}\{\boldsymbol{g}(\boldsymbol{\Theta}_t)\}\boldsymbol{Y}\left(\mathcal{T}^{-1}\{\boldsymbol{\Theta}_t\}\right)$$

- **Y** is calculated using real-valued SH equation.
- Using ACN and SN3D convention (ambix).
- Associated Legendre functions evaluated for sin(*θ*).
- Ambisonic degree (m) determines right hand argument.
  - Azimuth (ϕ).

$$Y_n^m(\varphi, \vartheta) = N_n^{|m|} P_n^{|m|}(\sin(\vartheta)) \begin{cases} \sin|m|\varphi, & \text{for } m < 0 \\ \cos|m|\varphi, & \text{for } m \geq 0 \end{cases}$$

# Chapter 1: AmbiX (Dir. Loudness/Warping)



filter 5 gain 3.7
azimuth -8.0
elevation -23.5
circular
width 28.2
height 101.4

filter 4 gain 0.0
azimuth 84.4
elevation 32.4
circular
width 18.6
height 30.4

- Great for SF recordings.
  - From mic array.
- Remove noisy signals from SF.
- Mixing/ mastering
- Con: <u>no visual</u>

# Chapter 1: AmbiX (Dir. Loudness/Warping)

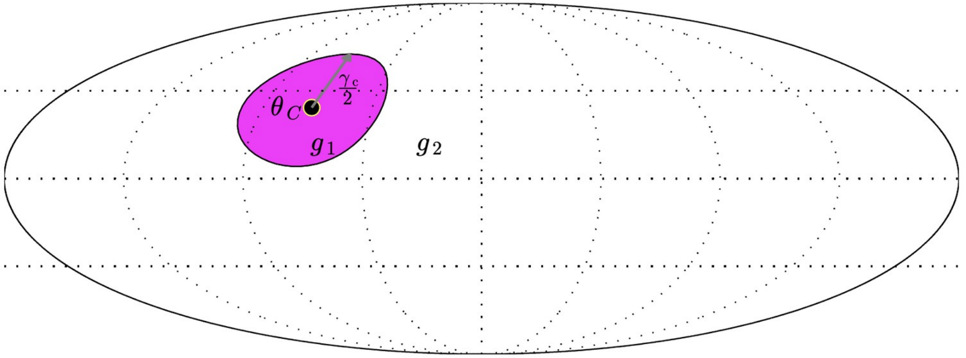
$$\boldsymbol{T} = \mathrm{diag}\{\boldsymbol{b}\}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)\,\mathrm{diag}\{\boldsymbol{g}(\boldsymbol{\Theta}_t)\}\,\boldsymbol{Y}\left(\cancel{\mathcal{T}^{-1}}\{\boldsymbol{\Theta}_t\}\right)$$

- Dir. loudness has no T^-1 transform.
- Only calculate the **g** coefficients.
- Use *unit step function u( )* to mute nodes.
  - Cropped region is either amplified or inverted then added to original.

Image from: “Spatial transformations for the enhancement of Ambisonic recordings” Kronlachner and Zotter (2014)

$$g(\boldsymbol{\theta}) = u\left(\boldsymbol{\theta}_{\mathrm{c}}^{\mathrm{T}}\boldsymbol{\theta} - \cos\frac{\gamma_{\mathrm{c}}}{2}\right)$$

θ_c = cartesian vector (center of cropped region)
θ = t-design coordinate
gamma_c = width of cropped region

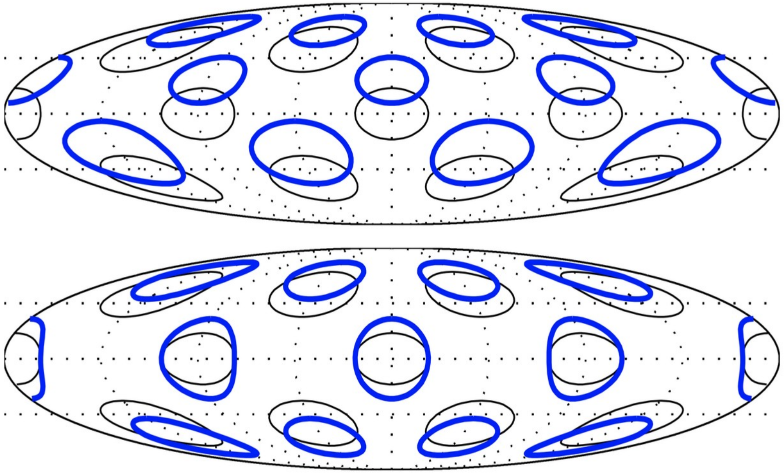


# Chapter 1: AmbiX (Dir. Loudness/Warping)

$$\boldsymbol{T} = \mathrm{diag}\{\boldsymbol{b}\}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)\,\mathrm{diag}\left\{\boldsymbol{g}(\boldsymbol{\Theta}_t)\right\}\boldsymbol{Y}\left(\mathcal{T}^{-1}\{\boldsymbol{\Theta}_t\}\right)$$

- Warping requires node angle and gain changes.
- Two types of warping:
  - Elevate/lower equator (top image)
  - Warp towards/away from equator (bottom image)

Image from: “Spatial transformations for the enhancement of Ambisonic recordings” Kronlachner and Zotter (2014)



# Chapter 1: AmbiX (Dir. Loudness/Warping)

$$\boldsymbol{T} = \mathrm{diag}\{\boldsymbol{b}\}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)\,\mathrm{diag}\{\boldsymbol{g}(\boldsymbol{\Theta}_t)\}\boxed{\boldsymbol{Y}\left(\mathcal{T}^{-1}\{\boldsymbol{\Theta}_t\}\right)}$$

- Elevate/lower equator:
  - **Bilinear transform (angle changes)**
  - Gain compensation to avoid clipping.
- $\mu = \sin(\vartheta)$
- Steps:
  - Isolate theta_x, theta_y, theta_z
  - Replace $\vartheta$ with warped version.
  - Re-calculate SH
    - using new theta_x, theta_y, theta_z.

$$\tilde{\mu} = \frac{\mu + \alpha}{1 + \alpha\mu}$$

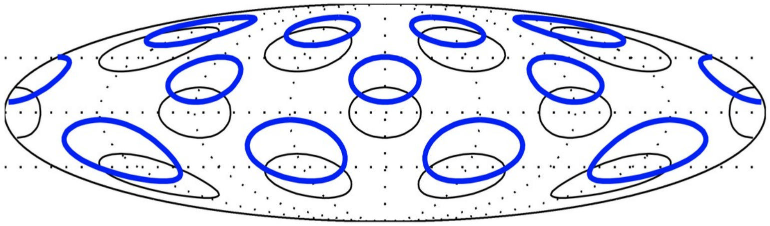
bilinear

$$\tilde{\vartheta} = \arcsin\left(\frac{\mu + \alpha}{1 + \alpha\mu}\right)$$

$$\vartheta = \arctan\frac{\theta_z}{\sqrt{\theta_x^2 + \theta_y^2}}$$

Cartesian to polar: elevation

$$\tilde{\mu} = \sin(\tilde{\vartheta})$$



# Chapter 1: AmbiX (Dir. Loudness/Warping)

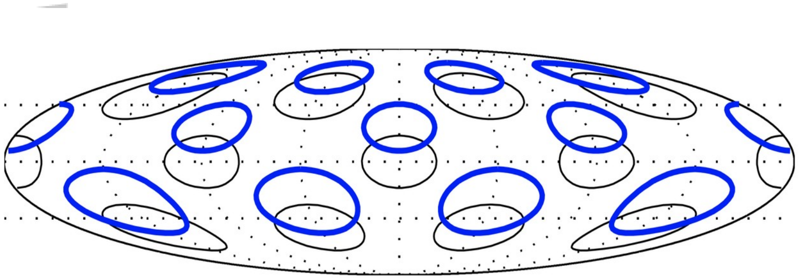
$$\boldsymbol{T} = \mathrm{diag}\{\boldsymbol{b}\}\boldsymbol{Y}^{\mathrm{T}}\left(\boldsymbol{\Theta}_t\right)\mathrm{diag}\,\boxed{\{\boldsymbol{g}\left(\boldsymbol{\Theta}_t\right)\}}\,\boldsymbol{Y}\left(\mathcal{T}^{-1}\left\{\boldsymbol{\Theta}_t\right\}\right)$$

- Elevate/lower equator:
  - Bilinear transform (angle changes)
  - **Gain compensation to avoid clipping.**
- Use **new** angle values.
- Compute weight for each node.
- α determines warp direction.
  - [-1, 1]

$$g(\mu) = \frac{\sqrt{1-\alpha^2}}{1+\alpha\mu}$$

$$g(\sin(\vartheta)) = g\left(\sin\underbrace{\left(\tan^{-1}\left(\frac{\theta_z}{\sqrt{\theta_x^2+\theta_y^2}}\right)\right)}_{\vartheta}\right)$$

$$\vartheta = \arctan\frac{\theta_z}{\sqrt{\theta_x^2+\theta_y^2}}$$



# Chapter 1: AmbiX (Dir. Loudness/Warping)

$$\boldsymbol{T} = \mathrm{diag}\{\boldsymbol{b}\}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{\Theta}_t)\,\mathrm{diag}\{\boldsymbol{g}(\boldsymbol{\Theta}_t)\}\,\boldsymbol{Y}\left(\mathcal{T}^{-1}\{\boldsymbol{\Theta}_t\}\right)$$

- Warp towards/away from equator
- Used in HOAST for acoustic zoom.
- Implementation follows same process as before.

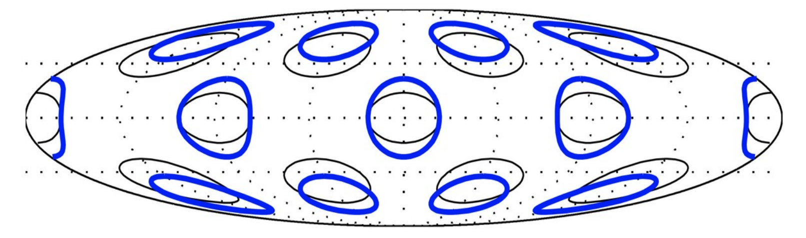
$$\tilde{\mu} = \begin{cases} \frac{(|\beta|-1)+\sqrt{(|\beta|-1)^2+4|\beta|\mu^2}}{2|\beta|\mu}, & \text{for } \beta > 0 \\ \frac{(1-|\beta|)\mu}{1-|\beta|\mu^2}, & \text{for } \beta < 0 \end{cases}$$

Angle transform

Gain adjustment

$$g(\mu) = \left(\frac{1-|\beta|\mu^2}{\sqrt{(1-|\beta|)(1+|\beta|\mu^2)}}\right)^{\mathrm{sgn}\{\beta\}}$$

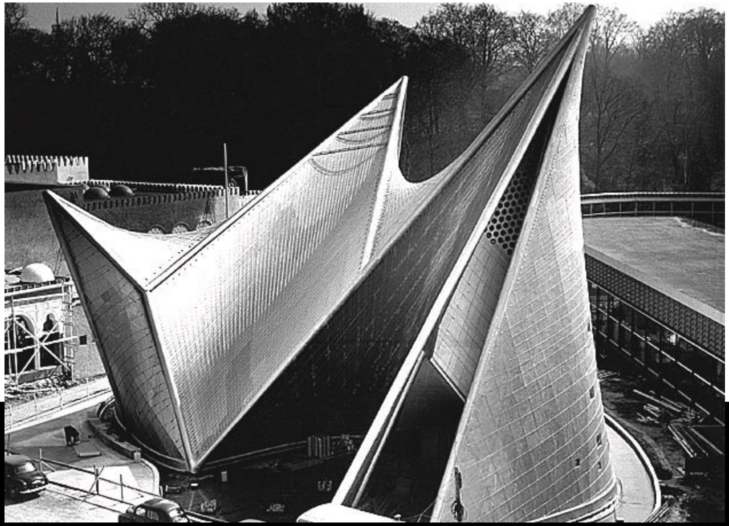
$$\tilde{\mu} = \sin(\tilde{\vartheta})$$



# Chapter 1: AmbiX (Dir. Loudness/Warping) [todo]

- Kronlachner’s thesis has some interesting hyperbolic paraboloid warping curves.
- One could also design a system like *expr* where users input arbitrary warping functions.

$$\begin{aligned}\mu &= \cos\vartheta\\ \alpha &= 0.8\sin 2\phi,\\ \tilde{\mu} &= \tfrac{\alpha+\mu}{1+\mu\alpha}\\ \tilde{\vartheta} &= \arccos\tilde{\mu}.\end{aligned}$$



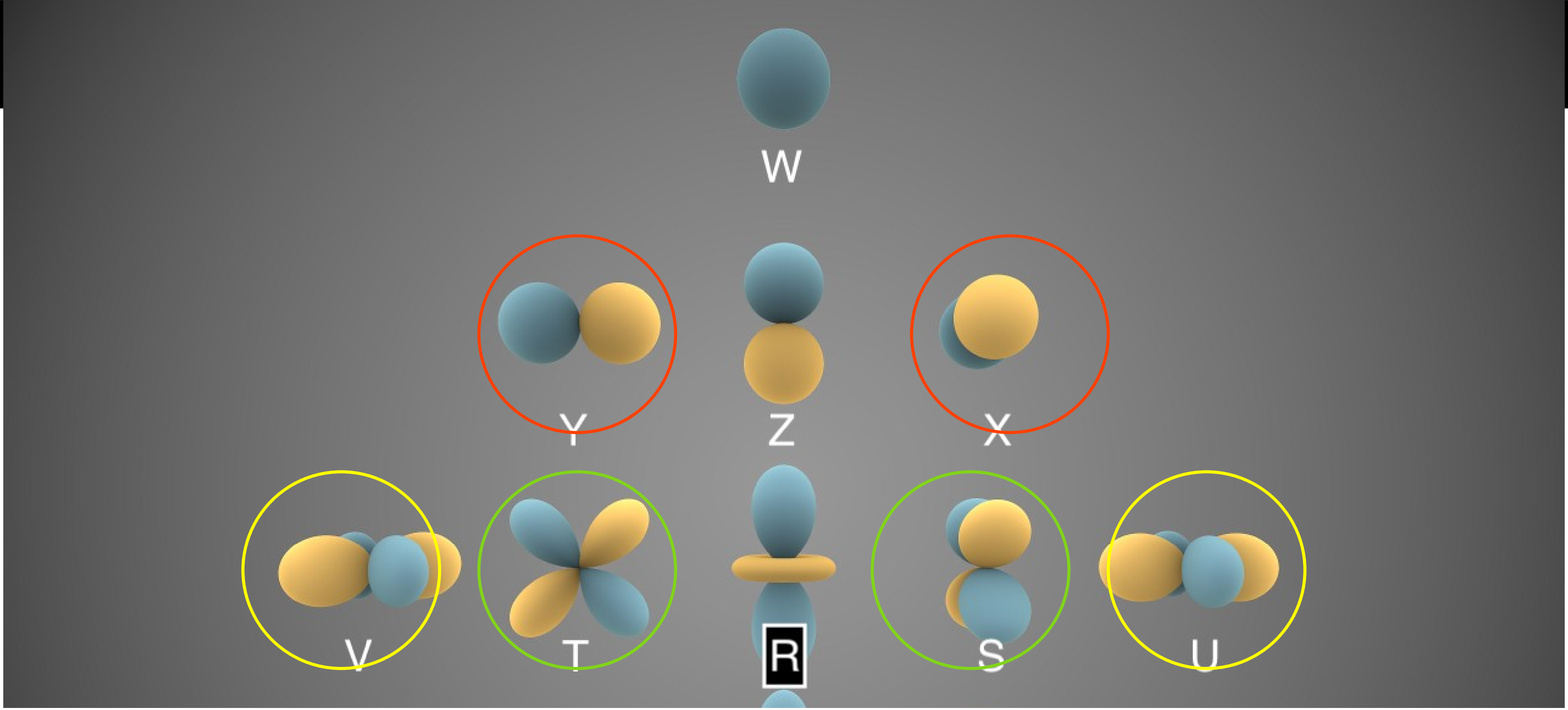
Inspired by Philips Pavilion
1958 expo
Varese

# Chapter 1: AmbiX (Widening)

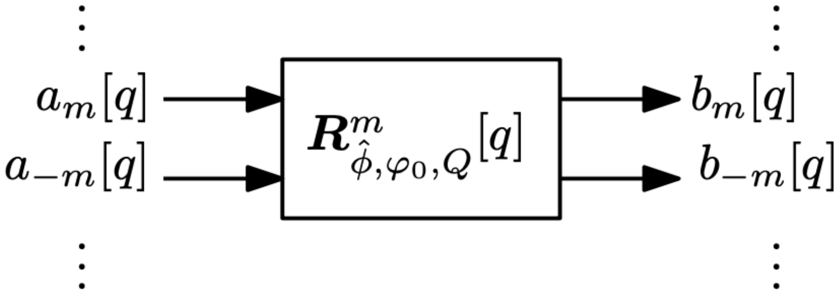
- Time-domain filter defined as product of *cos* with Bessel function (J_lambda).
  - Bessel functions: cylindrical harmonic equations (appear often in ambisonics research).
- Used to define frequency-dependent rotations about z-axis.
- Full widening will use pre/post-rotations.
  - Zotter method

$$H(\alpha,\beta,\Omega) = \cos[\alpha\cos(\Omega)+\beta]$$

$$\xrightarrow{\mathcal{IFT}} h(\alpha,\beta,t) = \sum_{\lambda=-\infty}^{\infty} \cos\left(\frac{\pi}{2}|\lambda|+\beta\right)\underline{J_{|\lambda|}(\alpha)}\delta(t+\lambda)$$


W
Y
Z
X
V
T
R
S
U

Etc.


$a_m[q]$
$a_{-m}[q]$
$\boldsymbol{R}^m_{\hat{\phi},\varphi_0,Q}[q]$
$b_m[q]$
$b_{-m}[q]$
$\vdots$
$\vdots$
$\vdots$
$\vdots$

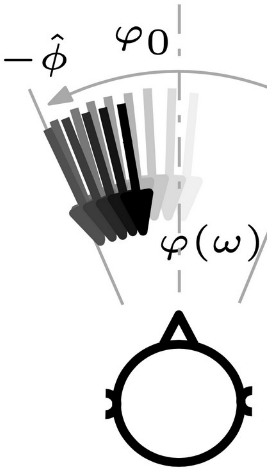
# Chapter 1: AmbiX (Widening)

$$H(\alpha,\beta,\Omega) = \cos[\alpha\cos(\Omega)+\beta]$$

$$\xrightarrow{\mathcal{IFT}} h(\alpha,\beta,t) = \sum_{\lambda=-\infty}^{\infty} \cos\left(\frac{\pi}{2}|\lambda|+\beta\right) J_{|\lambda|}(\alpha)\delta(t+\lambda)$$

sub

$$\boldsymbol{R}^{m}_{\hat{\phi},\varphi_0,\mathrm{Q}}[\lambda \mathrm{Q}] = \begin{bmatrix} h\Big(m\hat{\phi}, m\varphi_0, (\lambda-\Lambda)\mathrm{Q}\Big] & h\Big(m\hat{\phi}, m\varphi_0+\frac{\pi}{2}, (\lambda-\Lambda)\mathrm{Q}\Big] \\ h\Big(m\hat{\phi}, m\varphi_0-\frac{\pi}{2}, (\lambda-\Lambda)\mathrm{Q}\Big] & h\Big(m\hat{\phi}, m\varphi_0, (\lambda-\Lambda)\mathrm{Q}\Big] \end{bmatrix}$$



Lambda [-9, 9]

- $m$ is the ambisonic degree of the harmonic,
- $\hat{\phi}$ is the dispersion constant (e.g. how wide or narrow the image will be),
- $\varphi_0$ is *Direction of Arrival* (DoA) (e.g. the center of the widened area),
- $\lambda$ is an real integer value index, pertaining to the non-zero entries of the filters,
- Q is the "time grid" and defines the number of zero entries between non-zero filter entries in the time domain (i.e. 110), and
- $\Lambda$ is the truncation offset (i.e. 9) that defines the number of non-zero entries in each filter ( $0 <= \lambda <= 2\Lambda$).

# Cons

- Time grid cannot (glitches) be modulated/automated (different implementation might allow this).
- Some parameters not explained in the paper or documentation.
  - “The parameter Single Sided can be used to truncate the acausal part of the filter and therefore avoid pre-echoes.” (Kronlachner thesis – Figure caption)
- Not as much online documentation with these FX = confusion.
- However, lo-fi graphics = runs on cheap computers.

# Chapter 1: IEM Plug-in Suite

- Could not find publications ass. with these FX
  - Developers focus on online docs.
- Pro:
  - The easiest to use and has everything one needs to produce in ambisonics.
- Lack of publications means we have to inspect the code in order to understand the details.

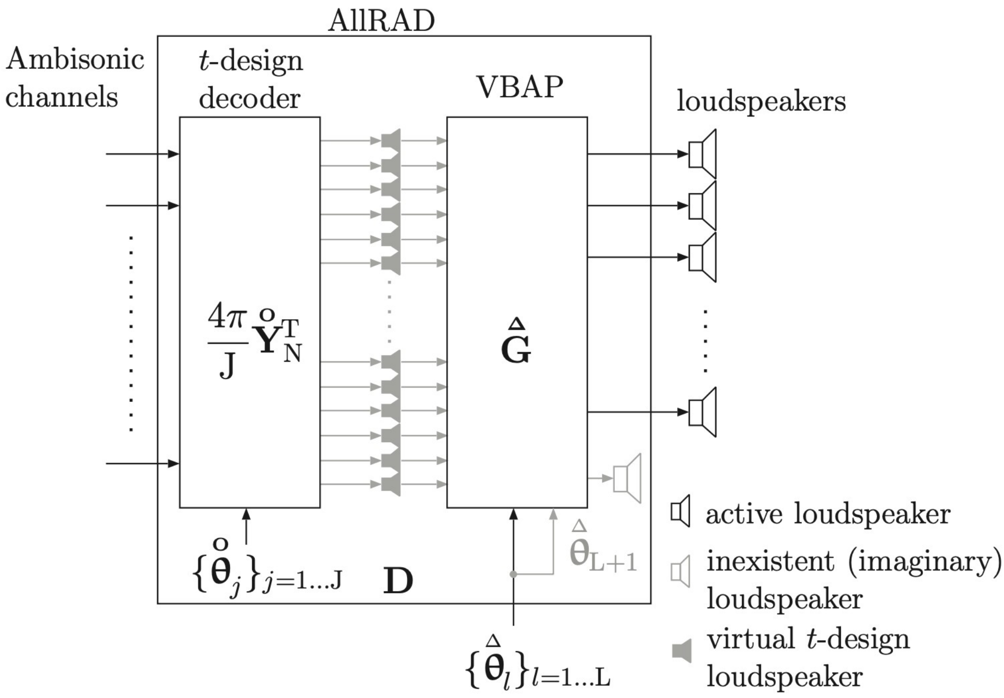
# Chapter 1: IEM Plug-in Suite

- **MultiBandCompressor (MBC):**
  - like traditional MBC but applies same compression to all SH
    - (based on W).
- **OmniCompressor:**
  - simpler the MBC, only one band. Also applies same dynamics to all SH
    - (based on W).
- **RoomEncoder:**
  - image source method (ISM) reverb (expensive CPU, FDN cheaper)!
    - ISM not discussed in detail

# Chapter 1: IEM Plug-in Suite

- **AllRADecoder:** all round ambisonic decoder.
  - Ambi decode *virtually* to t-design
  - Initialize VBAP using real speaker coordinates.
  - Render each virtual speaker signal using VBAP
    - Constant energy and spread.

ALL-ROUND AMBISONIC PANNING AND DECODING (Zotter 2012)



# Chapter 1: IEM Plug-in Suite

- **BinauralDecoder:** via MagLS - encode HRTFs into SH (LS).
  - MagLS optimizes freq response
    - Solving only for mag at high freq.
    - Normal LS suffers from high-freq roll-off (SPR, TA, etc.).

**Naive LS**

$$\hat{\mathbf{H}} = \mathbf{H}\mathbf{D}^{\mathbf{T}} \qquad \mathbf{D} = \mathbf{L}^{\dagger} = \mathbf{L}^{\mathbf{T}}\left(\mathbf{L}\mathbf{L}^{\mathbf{T}}\right)^{-1}$$

HRTF Expansion

Moore-Penrose

L - HRTF directions

$$\mathbf{L} = \begin{bmatrix} Y_0^0(\Phi_1,\Theta_1) & Y_0^0(\Phi_i,\Theta_i) & \dots & Y_0^0(\Phi_N,\Theta_N) \\ Y_1^{-1}(\Phi_1,\Theta_1) & Y_1^{-1}(\Phi_i,\Theta_i) & \dots & Y_1^{-1}(\Phi_N,\Theta_N) \\ \vdots & \vdots & \vdots & \vdots \\ Y_n^m(\Phi_1,\Theta_1) & Y_n^m(\Phi_i,\Theta_i) & \dots & Y_n^m(\Phi_N,\Theta_N) \end{bmatrix}$$

According to Gorzel et al.

# Chapter 1: IEM Plug-in Suite

- **DirectionalCompressor:** dynamic compressor for specific area.
  - Likely based on Dir. Loudness FX.
  - Side-chain masked region using un-masked region.
  - Listen to each individually.

**Directional**Compressor

SN3D
Auto (3rd)

Spatial mask properties
0.00 ° Azimuth
0.00 ° Elevation
40.00 ° Width

General settings
0.0 dB Pre Gain
Full

Compressor 1 ON
Masked
Masked
-10.0 dB Threshold
0.0 dB Knee
0.0 dB MakeUp
4.0 : 1 Ratio
30.0 ms Attack
150.0 ms Release

Compressor 2 ON
Masked
Masked
-10.0 dB Threshold
0.0 dB Knee
0.0 dB MakeUp
4.0 : 1 Ratio
30.0 ms Attack
150.0 ms Release

# Chapter 1: IEM Plug-in Suite

- **DirectivityShaper:** encode 4 bands w/ different angles + control weighting (vmic)
  - Demo (if time allows)
- **DistanceCompensator:** advanced calibration software, input distances
  - FX calculates DLAs and gain compensations
  - assumes hardware/ software levels are balanced (e.g. no measurement).
- **DualDelay:** two delay lines with LFOs panned in ambisonics.

# Featured Piece

- POI: XR + low-cost ambisonics.
- Song by Tim Gmeiner (ICAM) and support by Eito Murakami (ICAM).
- **Mixed w/ IEM plug-in suite (and Reaper) in 3OA.**
- Using as much FOSS and X-compatible software as possible.
- https://gabrielzalles.neocities.org/

# Chapter 1: Aalto Suite

- **SPARTA suite**:
  - Main part of the suite.
- **COMPASS suite**:
  - Coding and Multi-directional Parameterisation of Ambisonic Sound Scenes (COMPASS)
- **HO-DirAC suite**:
  - Higher Order Directional Audio Coding suite is based on the DirAC method
- **Other**:
  - CroPaC decoder (Cross-Pattern Coherence Decoder)
  - HOSSIR (Higher-order Spatial Impulse Response Rendering)

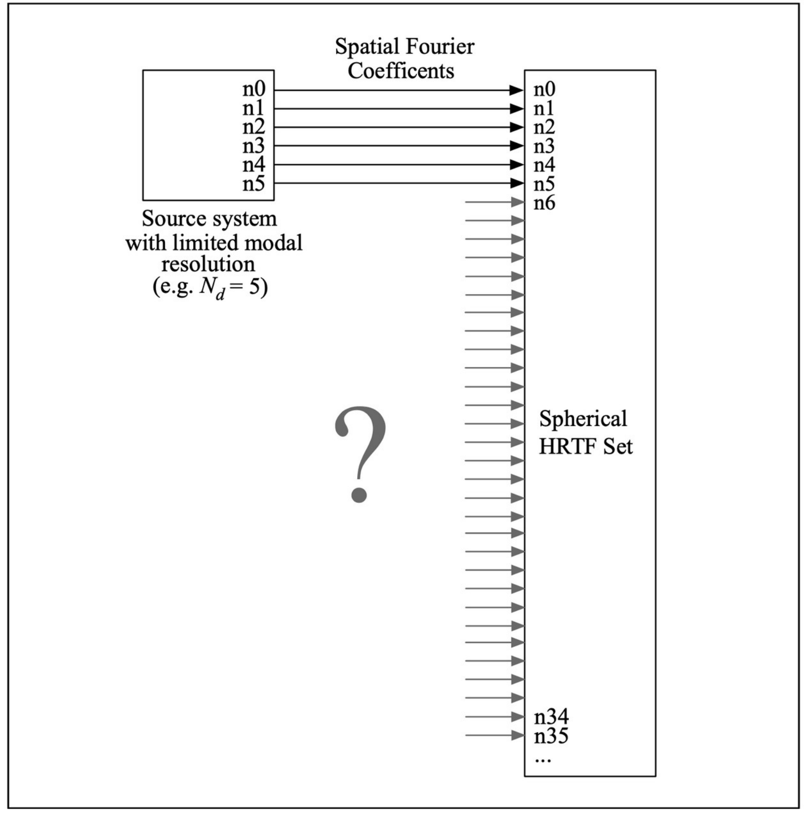
# Chapter 1: Aalto Suite

- **SPARTA suite**:

  **sparta_ambiBIN**

  - **LS:** already discussed.
  - **SPR:** re-sample HOA HRTF matrix to lower order. Higher orders contains high-frequency info.
  - **TA:** next slide
  - **MagLS:** more later.

**Binaural Reproduction of Plane Waves With Reduced Modal Order (SPR) [Bernschutz et al. 2014]**

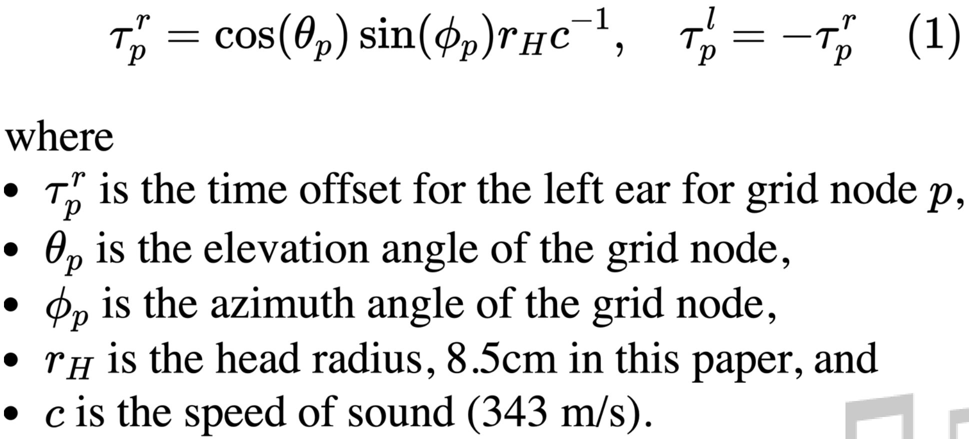


# Chapter 1: Aalto Suite

- **SPARTA suite**:

**sparta_ambiBIN**

- **TA: geometric model + all-pass filter.**



$$\tau_p^r = \cos(\theta_p)\sin(\phi_p) r_H c^{-1}, \quad \tau_p^l = -\tau_p^r \qquad (1)$$

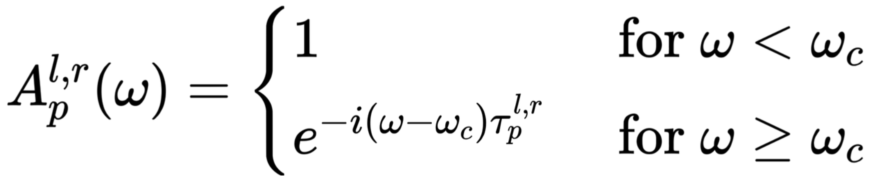
where

- $\tau_p^r$ is the time offset for the left ear for grid node $p$,
- $\theta_p$ is the elevation angle of the grid node,
- $\phi_p$ is the azimuth angle of the grid node,
- $r_H$ is the head radius, 8.5cm in this paper, and
- $c$ is the speed of sound (343 m/s).

**Calculate ITD**

subtract linear phase part of HRTFs at high freq (above 2kHz).

ITDs are more salient at low frequencies (Duplex Theory – Lord Raleigh).



$$A_p^{l,r}(\omega) = \begin{cases} 1 & \text{for } \omega < \omega_c \\ e^{-i(\omega-\omega_c)\tau_p^{l,r}} & \text{for } \omega \geq \omega_c \end{cases}$$

w_c = cut-off freq.
p = position of node.

**Design filter.**

**Final step is applying the filters. Equation not shown. Just convolution.**

# Chapter 1: Aalto Suite

- **SPARTA suite**:

**sparta_ambiBIN**

- **MagLS: normal LS below critical freq + mag LS above**
  - **phase correction for high frequencies.**

$$\hat{\boldsymbol{h}}_{\mathrm{SH},k} = \left(\boldsymbol{Y}_{\mathrm{N}}^{\mathrm{T}} \boldsymbol{Y}_{\mathrm{N}}\right)^{-1} \boldsymbol{Y}_{\mathrm{N}}^{\mathrm{T}} \left[h_{l,k}\right]_{l}$$

Bin = k
Index of direction = l (el)
Number of directions = L
N = ambisonic order
T = transpose
Y = SH of HRTF positions
HRTFs = h

$$\hat{\phi}_{l,k-1} = \angle\left\{\boldsymbol{y}_{\mathrm{N}}(\boldsymbol{\theta}_l)^{\mathrm{T}} \hat{\boldsymbol{h}}_{\mathrm{SH},k-1}\right\},$$

$$\hat{\boldsymbol{h}}_{\mathrm{SH},k} = \left(\boldsymbol{Y}_{\mathrm{N}}^{\mathrm{T}} \boldsymbol{Y}_{\mathrm{N}}\right)^{-1} \boldsymbol{Y}_{\mathrm{N}}^{\mathrm{T}} \left[\left|h_{l,k}\right| e^{\mathrm{i}\hat{\phi}_{l,k-1}}\right]_{l}$$

Starting with 1st bin above f_c, calculate phase (angle()). Use magnitude response with modified phase to find MagLS solution.

# Chapter 1: Aalto Suite

- **SPARTA suite**:

    **sparta_ambiDEC -**

    - **AllRAD:** already discussed.
    - **EPAD:** use SVD to obtain eigenvalues of decoder matrix.
        - Removing them from solution results in energy preserving decoder.
        - The final EPAD design is then optimized for hemispherical layouts using a second SVD.
    - **SAD:** discussed later.
    - **MMD:** discussed later.

AllRAD, SAD, MMD discussed more later.

# Chapter 1: Aalto Suite

- **SPARTA suite**:

**sparta_ambiDEC -**

  - **EPAD: SVD on Y + remove singular values + hemisphere optimize.**

Expansion and re-factorization.
For a hemisphere.

$$\boldsymbol{D}_{\mathrm{EPAD}} = \hat{\boldsymbol{V}}\hat{\boldsymbol{U}}^{\mathrm{T}}$$

$$\check{\boldsymbol{U}}\check{\boldsymbol{S}}\check{\boldsymbol{V}}^{\mathrm{T}} = \mathrm{svd}\{\boldsymbol{U}^{\mathrm{T}}_{:,1:\mathrm{N}+1}\boldsymbol{Y}_{\mathrm{N}}\}$$

For a regular sphere
Pseudo-inverse sans
eigenvalues

**Solution**: V hat and U hat from re-factorization.

$$\boldsymbol{D}_{\mathrm{EPAD}} = \check{\boldsymbol{V}}_{:,1:\mathrm{N}+1}\check{\boldsymbol{U}}^{\mathrm{T}}\boldsymbol{U}^{\mathrm{T}}_{:,1:\mathrm{N}+1}$$

AllRAD, SAD, MMD discussed more later.

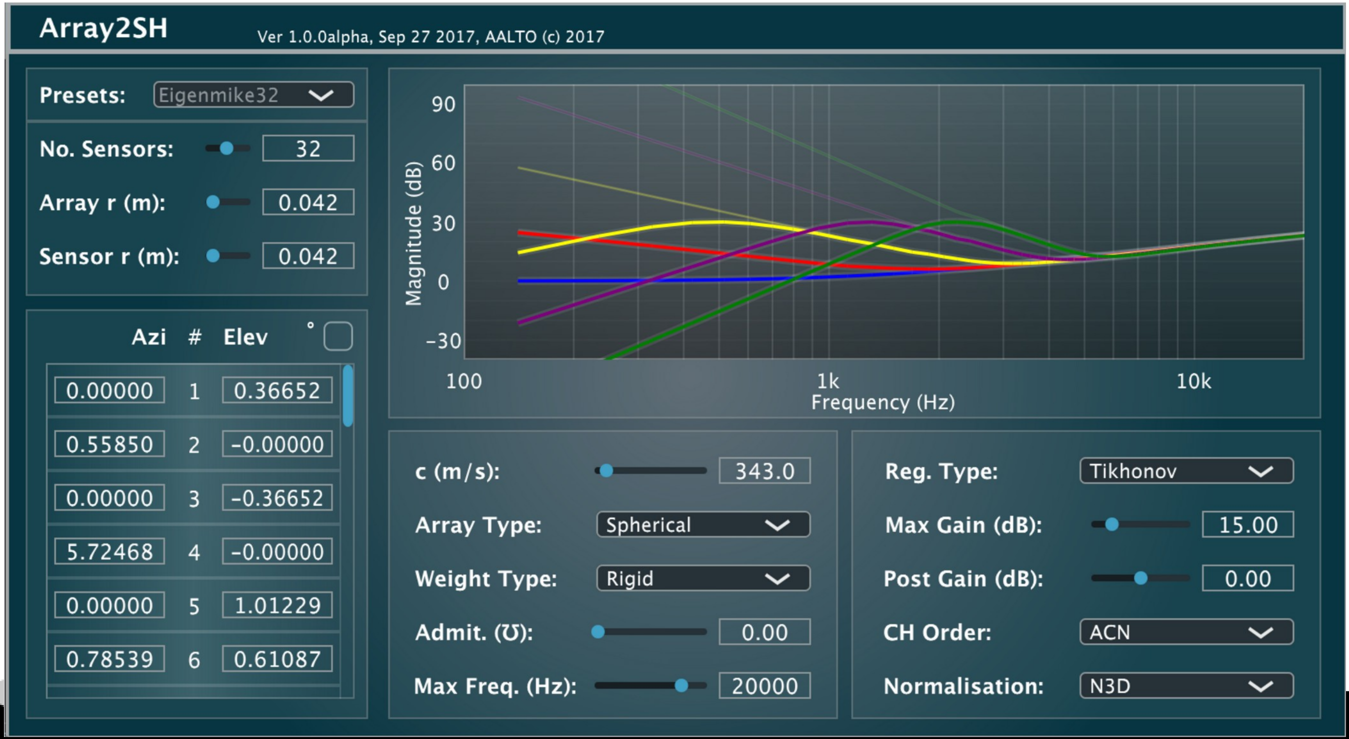
# Chapter 1: Aalto Suite

- **SPARTA suite**:

**sparta_array2sh: encoding matrix for microphone array (regular/irregular) + radial filters.**

- Array radius
- Sensor radius
- Coordinates
- Array type
- Rigid/hollow
- Etc.

Model based encoding.
More on SMAs in Chapter 2.



**Array2SH** Ver 1.0.0alpha, Sep 27 2017, AALTO (c) 2017

**Presets:** Eigenmike32

**No. Sensors:** 32

**Array r (m):** 0.042

**Sensor r (m):** 0.042

| Azi | # | Elev |
|---|---|---|
| 0.00000 | 1 | 0.36652 |
| 0.55850 | 2 | -0.00000 |
| 0.00000 | 3 | -0.36652 |
| 5.72468 | 4 | -0.00000 |
| 0.00000 | 5 | 1.01229 |
| 0.78539 | 6 | 0.61087 |

**c (m/s):** 343.0

**Array Type:** Spherical

**Weight Type:** Rigid

**Admit. (℧):** 0.00

**Max Freq. (Hz):** 20000

**Reg. Type:** Tikhonov

**Max Gain (dB):** 15.00

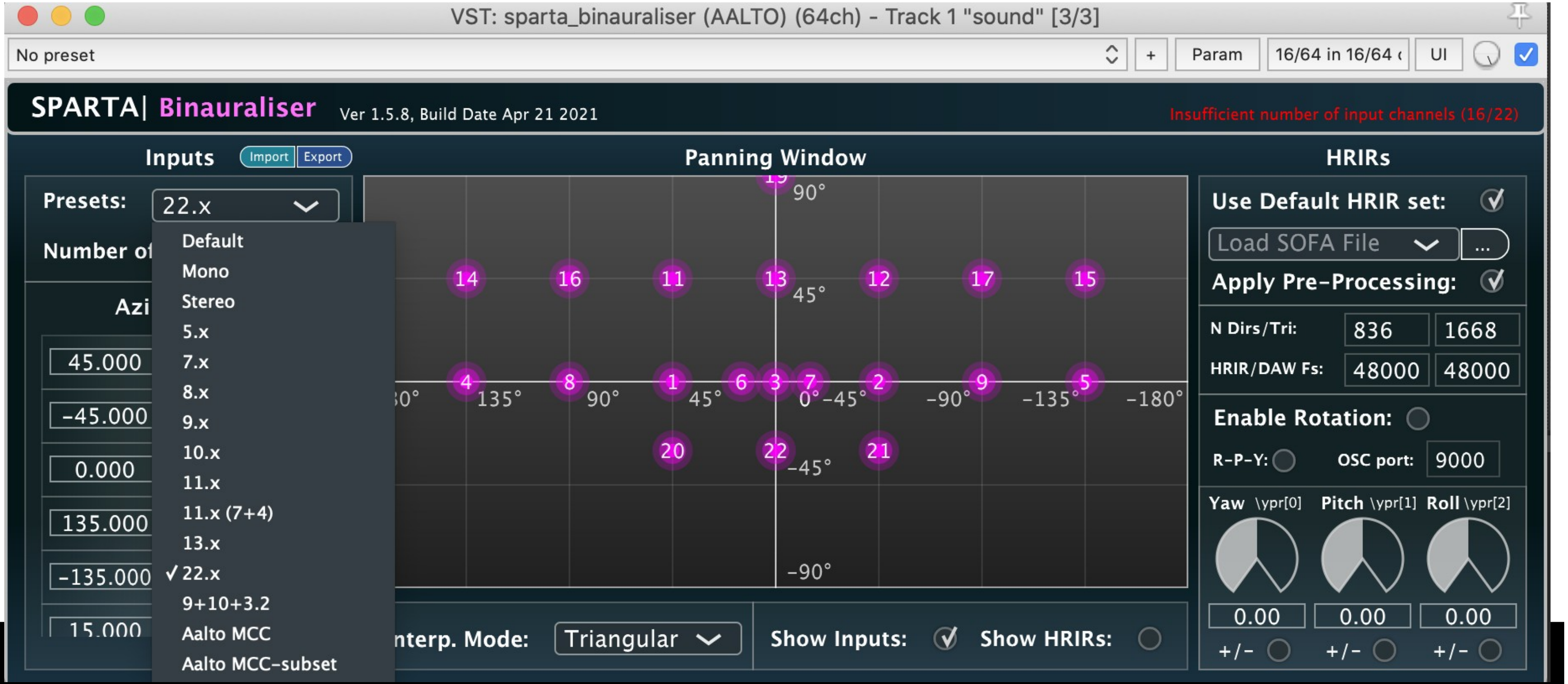
**Post Gain (dB):** 0.00

**CH Order:** ACN

**Normalisation:** N3D

# Chapter 1: Aalto Suite

- **SPARTA suite**:
  - **sparta_binauraliser**
    - Not ambisonic, used to binauralize speaker channels (surround sound).



VST: sparta_binauraliser (AALTO) (64ch) - Track 1 "sound" [3/3]

No preset

16/64 in 16/64 (

SPARTA| Binauraliser Ver 1.5.8, Build Date Apr 21 2021

Insufficient number of input channels (16/22)

Presets: 22.x

Interp. Mode: Triangular

Show Inputs: ☑

Show HRIRs: ○

Use Default HRIR set: ☑

Apply Pre-Processing: ☑

N Dirs/Tri: 836 1668

HRIR/DAW Fs: 48000 48000

Enable Rotation: ○

R-P-Y: ○

OSC port: 9000

Yaw \ypr[0] 0.00

Pitch \ypr[1] 0.00

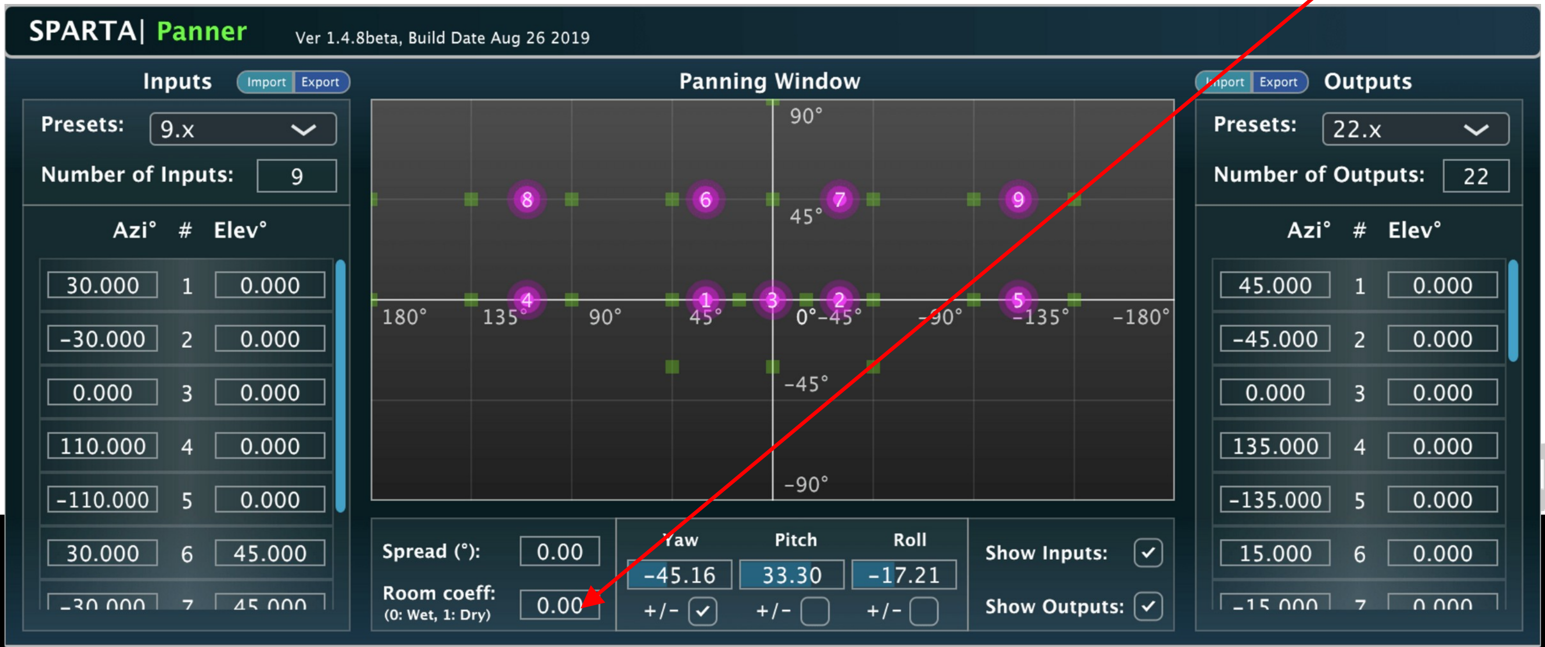
Roll \ypr[2] 0.00

**Aalto University**
School of Electrical Engineering

# Chapter 1: Aalto Suite

- **SPARTA suite:**
  - **sparta_panner**
    - Advanced VBAP: takes into account frequency dependent effects of speaker placement and room acoustics, and uses these additional criteria to normalize signal gains in the frequency domain.

Room coeff



Ver 1.4.8beta, Build Date Aug 26 2019

Presets: 9.x

Number of Inputs: 9

| Azi° | # | Elev° |
|---|---|---|
| 30.000 | 1 | 0.000 |
| -30.000 | 2 | 0.000 |
| 0.000 | 3 | 0.000 |
| 110.000 | 4 | 0.000 |
| -110.000 | 5 | 0.000 |
| 30.000 | 6 | 45.000 |
| -30.000 | 7 | 45.000 |

Spread (°): 0.00

Room coeff: (0: Wet, 1: Dry) 0.00

| Yaw | Pitch | Roll |
|---|---|---|
| -45.16 | 33.30 | -17.21 |
| +/- ☑ | +/- ☐ | +/- ☐ |

Show Inputs: ☑

Show Outputs: ☑

Presets: 22.x

Number of Outputs: 22

| Azi° | # | Elev° |
|---|---|---|
| 45.000 | 1 | 0.000 |
| -45.000 | 2 | 0.000 |
| 0.000 | 3 | 0.000 |
| 135.000 | 4 | 0.000 |
| -135.000 | 5 | 0.000 |
| 15.000 | 6 | 0.000 |
| -15.000 | 7 | 0.000 |

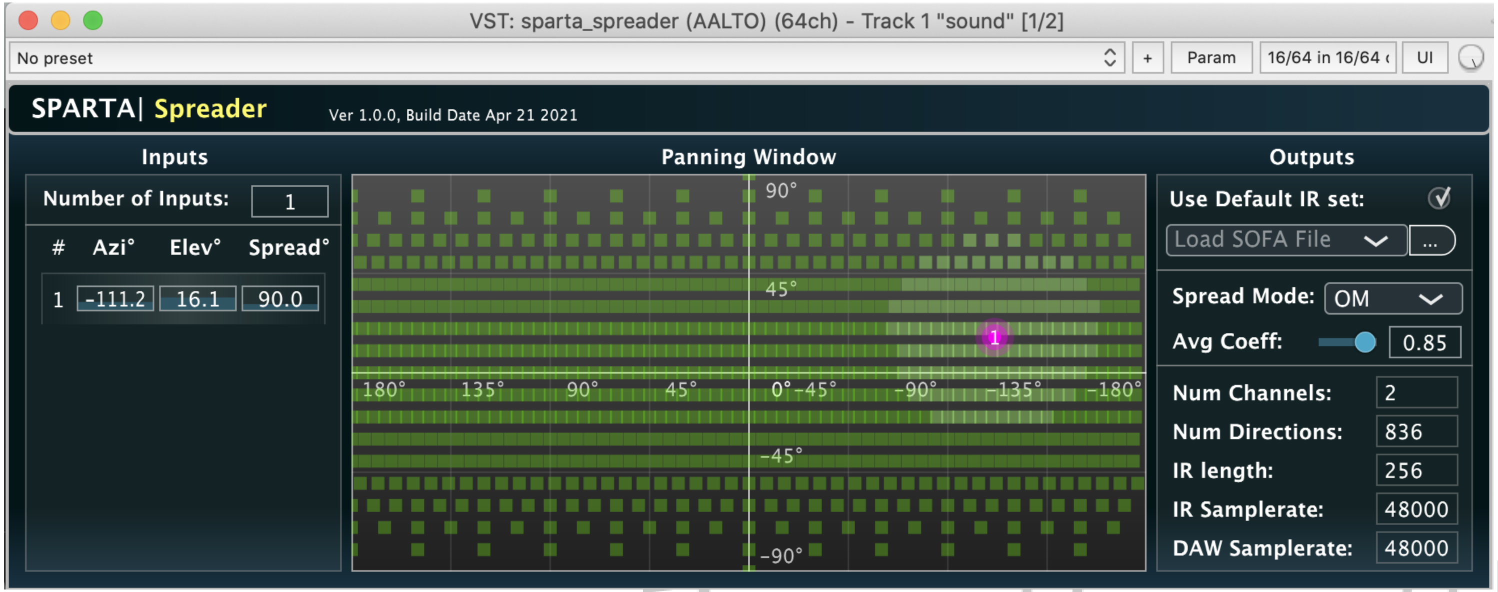
Image shows panning 9.x to 22.x

Aalto University
School of Electrical Engineering

# Chapter 1: Aalto Suite

- **SPARTA suite**:

  **sparta_spreader -** An arbitrary array (e.g. HRIRs or microphone array IRs) panner with coherent and incoherent spreading options.



VST: sparta_spreader (AALTO) (64ch) - Track 1 "sound" [1/2]

No preset

SPARTA| Spreader

Ver 1.0.0, Build Date Apr 21 2021

**Inputs**

Number of Inputs: 1

| # | Azi° | Elev° | Spread° |
|---|---|---|---|
| 1 | -111.2 | 16.1 | 90.0 |

**Outputs**

Use Default IR set: ✓

Load SOFA File

Spread Mode: OM

Avg Coeff: 0.85

Num Channels: 2

Num Directions: 836

IR length: 256

IR Samplerate: 48000

DAW Samplerate: 48000

Default is binaural output. Can load SOFA SMA IRs.

Aalto University
School of Electrical Engineering

# Chapter 1: Aalto Suite

- **SPARTA suite**:

  **sparta_spreader -**

  - **Not much info online. Leo McCormack got back to us via email:**

*The 'Basic' spreading mode is just making identical/coherent copies of the input signal and convolving them with all the HRIRs (or, e.g. microphone array IRs) that are within the spreading area, followed by taking the mean.*

Not very interesting for HRIRs, perhaps once there are more SMA IR Data-sets.

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**
  - COMPASS:
    - Statistics of direct sound and diffuse sound.
    - EVD of PSD, followed by SORTE (find num sources) and MUSIC (find DoA).

$$\mathbf{a}(t, f) = \mathbf{a}_{\mathrm{s}}(t, f) + \mathbf{a}_{\mathrm{d}}(t, f) = \mathbf{Y}_{\mathrm{s}}(t, f)\mathbf{s}(t, f) + \mathbf{a}_{\mathrm{d}}(t, f)$$

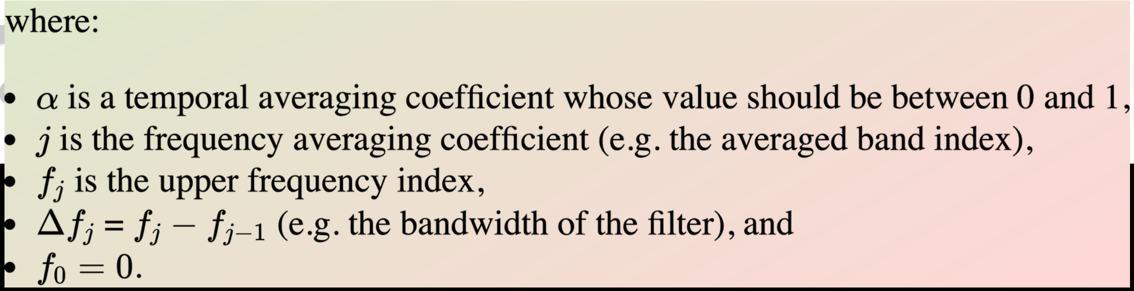
SF defined as source + diffuse signals.

Source signals defined as raw source signals plus encoding matrix.

Model.

PSD

$$\mathbf{C}_{\mathbf{a}}(t, j) = \alpha \mathbf{C}_{\mathbf{a}}(t-1, j) + \frac{(1-\alpha)}{\Delta f_j} \sum_{f_{j-1}+1}^{f_j} \mathbf{a}(t, f)\mathbf{a}^{\mathrm{H}}(t, f)$$



where:

- $\alpha$ is a temporal averaging coefficient whose value should be between 0 and 1,
- $j$ is the frequency averaging coefficient (e.g. the averaged band index),
- $f_j$ is the upper frequency index,
- $\Delta f_j = f_j - f_{j-1}$ (e.g. the bandwidth of the filter), and
- $f_0 = 0$.

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**
  - COMPASS:

PSD

$$\mathbf{C}_{\mathbf{a}}(t,j) = \alpha\mathbf{C}_{\mathbf{a}}(t-1,j) + \frac{(1-\alpha)}{\Delta f_j}\sum_{f_{j-1}+1}^{f_j} \mathbf{a}(t,f)\mathbf{a}^{\mathrm{H}}(t,f)$$

Multiple PSD matrices per window - based on frequency bands (ERB).

Value t corresponds to STFT window.

Sub-space has 3 dimensions.
i.e. 4-by-4 matrix of PSD for FOA
w/ J matrices (for each band)

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**
  - COMPASS:

EVD via toolbox.

$$\mathbf{C_a} = \mathbf{V}\mathbf{U}\mathbf{V}^{\mathrm{H}} = \sum_{m=1}^{K} \lambda_m \mathbf{v}_m \mathbf{v}_m^{\mathrm{H}} + \sum_{m=K+1}^{M} \lambda_m \mathbf{v}_m \mathbf{v}_m^{\mathrm{H}}$$

*Eigenvalue decomposition.*
*U is diagonal and contains eigenvalues.*
*Not the same as SVD! V are bases.*

*K = num sources assumed to be smaller than M.*
*M = num of SH (N+1)^2*

# Chapter 1: Aalto Suite

- EVD vs SVD.

*EVD uses only one basis, i.e. the eigenvectors, while he SVD uses two different bases, the left and right singular vectors.*

*The basis of the EVD is not necessarily orthogonal, the eigenbasis of the SVD is orthonormal!*

*Every matrix has a SVD, no questions asked. Not even every square matrix has an EVD.*

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**
  - COMPASS:

Using singular values from EVD we find the number of sources according to variances between eigenvalues (SORTE) *second order statistic of eigenvalues (skipping details)*

MUltiple SIgnal Classification (MUSIC) used to find DoAs.

*Noise subspace used to find smallest correlation w/ sampling grid.*

*Local minima = DoA*

$$\mathbf{p}_{\mathrm{MUSIC}} = \mathrm{diag}\left[\mathbf{Y}_{\mathrm{g}}^{\mathrm{T}}\mathbf{V}_{\mathrm{n}}\mathbf{V}_{\mathrm{n}}^{\mathrm{H}}\mathbf{Y}_{\mathrm{g}}\right]$$

Vn = noise subspace

Region where noise IS NOT, **MUST** be where sources ARE.

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**
  - COMPASS:

***Using DoAs, sources are extracted via beam-forming.***
***Power/amplitudes are calculated for sources and diffuse sound.***

***Three parameters are revealed by this process:***

- *Temporal smoothing*
  - *how many windows do we use for analysis?*
- *Diffuse v. directional source control (de-reverberation!).*
  - *Plus, ability to control de-correlation amount of diffuse signal.*
- *Parametric v. linear decoding.*

# Chapter 1: Aalto Suite

- **compass_binaural**
- **compass_binauralVR**
- **compass_decoder**

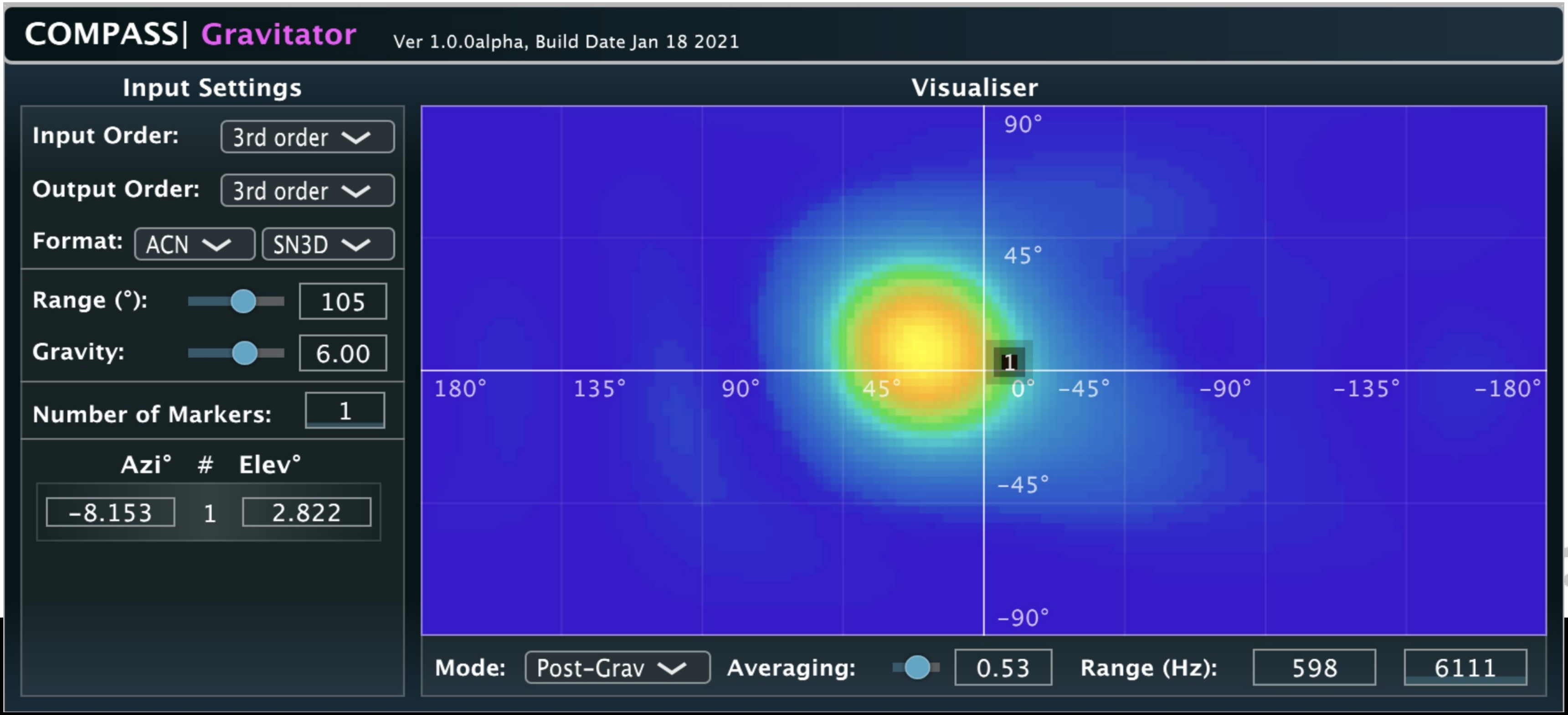
Nothing really that interesting here.
Other than aforementioned parameters.

One benefit is decoding lower order materials to HOA.

If K > M/2 the model might “fail”. Some of the sources will be treated as diffuse sound.

# Chapter 1: Aalto Suite

**compass_gravitator** – move all sources towards a desired location!



COMPASS| Gravitator Ver 1.0.0alpha, Build Date Jan 18 2021

Input Settings

Input Order: 3rd order

Output Order: 3rd order

Format: ACN SN3D

Range (°): 105

Gravity: 6.00

Number of Markers: 1

| Azi° | # | Elev° |
|---|---|---|
| -8.153 | 1 | 2.822 |

Mode: Post-Grav

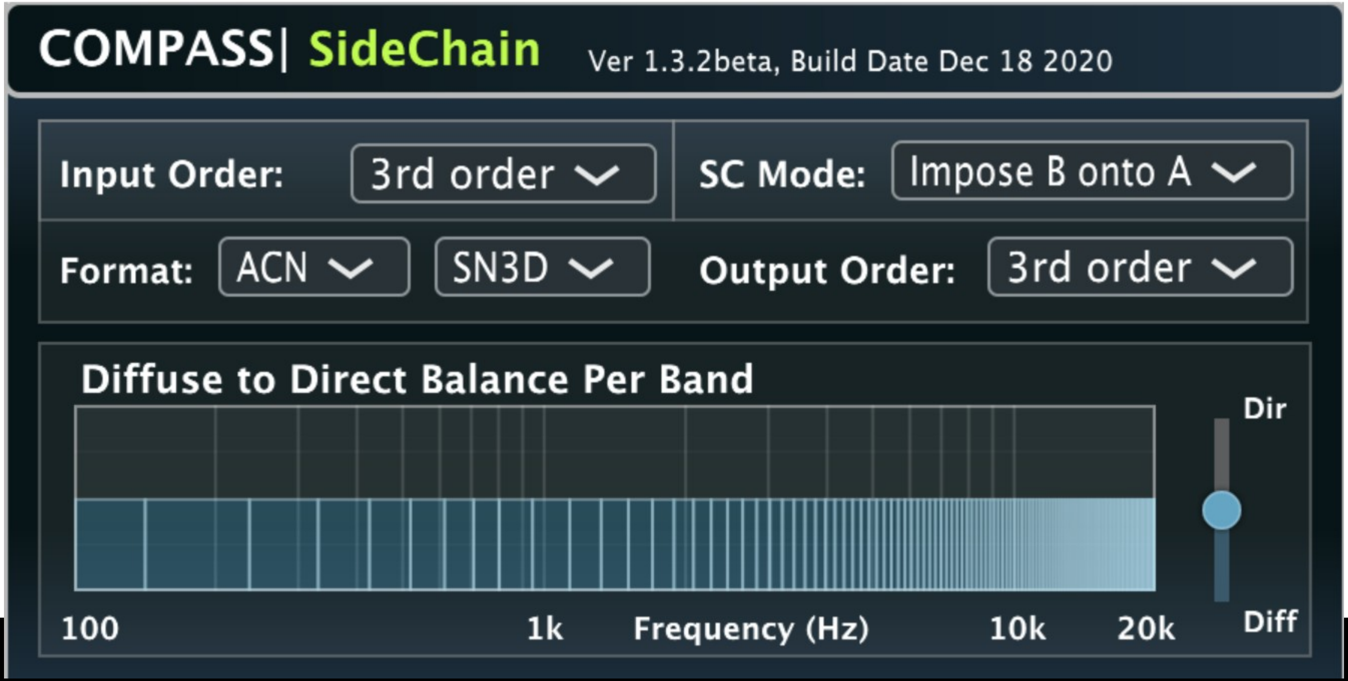
Averaging: 0.53

Range (Hz): 598 6111

# Chapter 1: Aalto Suite

**compass_sidechain** – apply parameters of one SF unto another.

- Not side chain in the traditional sense (dynamic compressor).



COMPASS| SideChain Ver 1.3.2beta, Build Date Dec 18 2020

Input Order: 3rd order

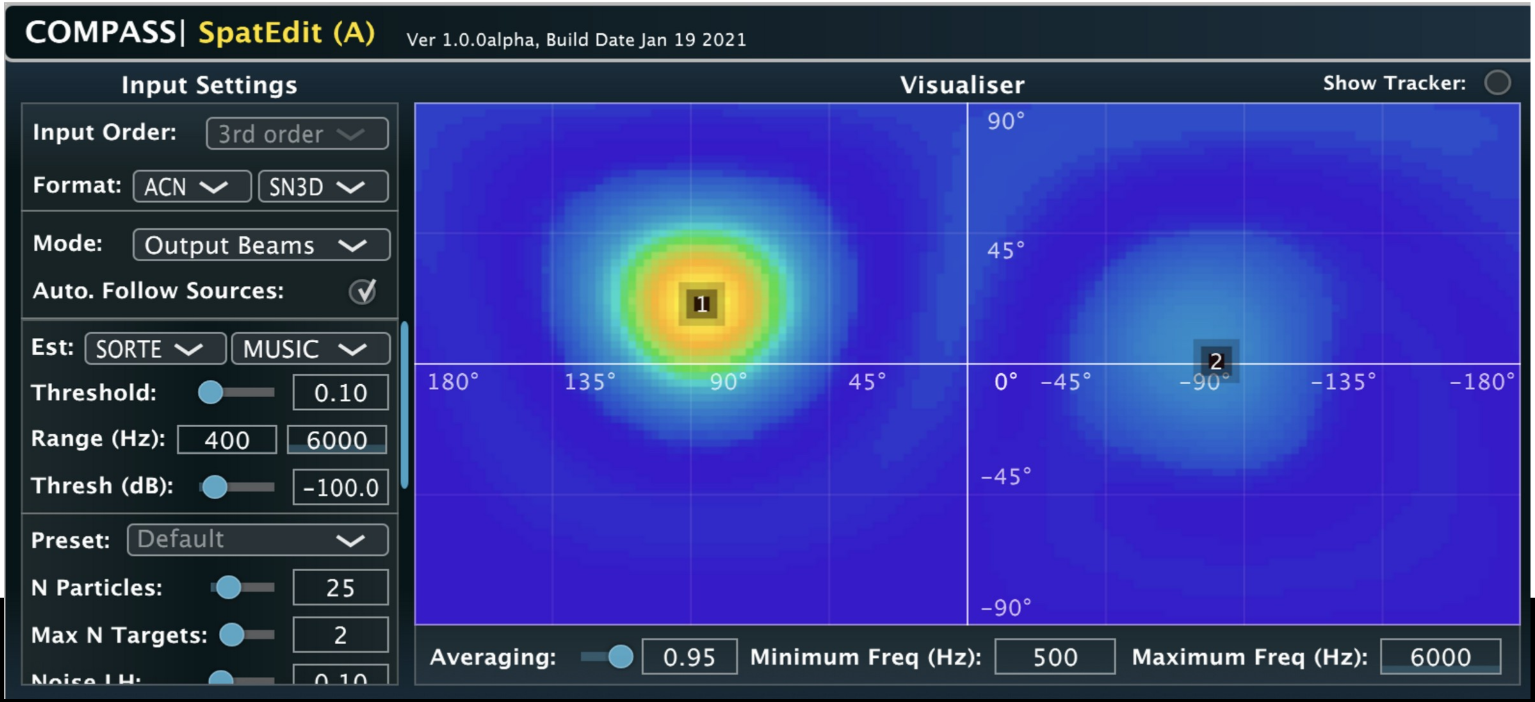
SC Mode: Impose B onto A

Format: ACN SN3D

Output Order: 3rd order

# Chapter 1: Aalto Suite

**compass_spatedit** – extract sources so FX can be applied to them. Alternatively, in second mode, extracts diffuse response, to apply FX.



COMPASS| SpatEdit (A) Ver 1.0.0alpha, Build Date Jan 19 2021

Input Settings

Input Order: 3rd order

Format: ACN SN3D

Mode: Output Beams

Auto. Follow Sources: ☑

Est: SORTE MUSIC

Threshold: 0.10

Range (Hz): 400 6000

Thresh (dB): -100.0

Preset: Default

N Particles: 25

Max N Targets: 2

Noise LH: 0.10

Averaging: 0.95

Minimum Freq (Hz): 500

Maximum Freq (Hz): 6000

# Chapter 1: Aalto Suite

**compass_tracker** – multi-source tracker.

- can be used to create “stems” from a complex SF.
  - Documentation does not describe if diffuse field is discarded, or applied to each stem.
  - Optimized for moving sources.

COMPASS| Tracker Ver 1.0.0beta, Build Date Dec 19 2020

**Analysis Settings**

Analysis Order: 4th order
Format: ACN, SN3D
Est: DPD-T, ESPRIT
Threshold: 2.00
Range (Hz): 2000 : 6000
Thresh (dB): -100.0

**Tracker Settings**

Preset: LOCATA-T4
N Particles: 60
Max N Targets: 2
Noise LH: 0.10
Meas Noise: 35.00
Noise SpecD: 0.30
Birth LH: 0.005
Death Alpha: 8.00
Death Beta: 1.00
Azi Vel: 5.00
Elev Vel: 0.00
Kill Dist: 0.00

**Output Settings**

Beamformer: Hyper-card
Enable CroPaC Post-Filter: ○
CroPaC lambda: 0.00
Binauralise Beamformers: ○
Use Default HRIR set: ☑
Load SOFA File

# Chapter 1: Aalto Suite

- **compass_upmixer** - (1-3rd order input, 2-7th order output)
    - Game-changer for accessible ambisonic production.
    - Many computers can’t handle HOA.
    - Instead up-mix for HDLA concerts!!!

    - Could not find source code for COMPASS :(

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

- Simpler than COMPASS, analysis performed inside regular regions of sphere.
- In FOA-DirAC we use W to find instantaneous pressure and XYZ to find velocity.
- In HOA we use *weighted* SF for HOA analysis (based on segments of sphere).

Fraunhofer IIS license!
No source :(

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

$$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$$

Magnitude squared of W channel = instantaneous power spectrum

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

<u>HO-DirAC:</u>

$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$ Magnitude squared of W channel = instantaneous power spectrum

$\hat{S}_{\mathrm{vv}}(k) = \mathbf{v}^H(k)\mathbf{v}(k)$ Combined power spectrum. [y z x] vector product returns scalar.

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

$$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$$

Magnitude squared of W channel = instantaneous power spectrum

$$\hat{S}_{\mathrm{vv}}(k) = \mathbf{v}^H(k)\mathbf{v}(k)$$

Combined power spectrum. [y z x] vector product returns scalar.

$$\hat{\mathbf{s}}_{\mathrm{pv}}(k) = p^*(k)\mathbf{v}(k)$$

X-spectrum. Returns 3 element vector. Complex values.

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

  - active intensity vector ($i_a(k)$), energy density ($E(k)$), and diffuseness ($\psi(k)$)

$$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$$

Instantaneous power spectrum

$$\mathbf{i_a}(k) = -\Re\{\hat{\mathbf{s}}_{\mathbf{pv}}(k)\}$$

$$\hat{S}_{\mathrm{vv}}(k) = \mathbf{v}^H(k)\mathbf{v}(k)$$

Combined power spectrum

$$\hat{\mathbf{s}}_{\mathrm{pv}}(k) = p^*(k)\mathbf{v}(k)$$

X-spectrum

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

- active intensity vector ($i_a(k)$), energy density ($E(k)$), and diffuseness ($\psi(k)$)

$$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$$

Instantaneous power spectrum

$$\hat{S}_{\mathrm{vv}}(k) = \mathbf{v}^H(k)\mathbf{v}(k)$$

Combined power spectrum

$$\hat{\mathbf{s}}_{\mathrm{pv}}(k) = p^*(k)\mathbf{v}(k)$$

X-spectrum

$$\mathbf{i_a}(k) = -\Re\{\hat{\mathbf{s}}_{\mathbf{pv}}(k)\}$$

$$E(k) = \frac{1}{2}\left[\hat{S}_{\mathrm{vv}}(k) + \hat{S}_{\mathrm{pp}}(k)\right]$$

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

  - active intensity vector ($\mathbf{i}_a(k)$), energy density ($E(k)$), and diffuseness ($\psi(k)$)

$$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$$

Instantaneous power spectrum

$$\hat{S}_{\mathrm{vv}}(k) = \mathbf{v}^H(k)\mathbf{v}(k)$$

Combined power spectrum

$$\hat{\mathbf{s}}_{\mathrm{pv}}(k) = p^*(k)\mathbf{v}(k)$$

X-spectrum

$$\mathbf{i}_{\mathbf{a}}(k) = -\Re\{\hat{\mathbf{s}}_{\mathbf{pv}}(k)\}$$

$$E(k) = \frac{1}{2}\left[\hat{S}_{\mathrm{vv}}(k) + \hat{S}_{\mathrm{pp}}(k)\right]$$

$$\psi(k) = 1 - \frac{2\|\Re\{\hat{\mathbf{s}}_{\mathrm{pv}}(k)\}\|}{\hat{S}_{\mathrm{vv}}(k) + \hat{S}_{\mathrm{pp}}(k)}$$

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

  - active intensity vector ($i_a(k)$), energy density ($E(k)$), and diffuseness ($\psi(k)$)

$$\hat{S}_{\mathrm{pp}}(k) = |p(k)|^2$$

Instantaneous power spectrum

$$\hat{S}_{\mathrm{vv}}(k) = \mathbf{v}^H(k)\mathbf{v}(k)$$

Combined power spectrum

$$\hat{\mathbf{s}}_{\mathrm{pv}}(k) = p^*(k)\mathbf{v}(k)$$

X-spectrum

$$\mathbf{n}(\Phi(k)) = \frac{\mathfrak{R}\{\hat{\mathbf{s}}_{\mathbf{pv}}(k)\}}{\|\mathfrak{R}\{\hat{\mathbf{s}}_{\mathbf{pv}}(k)\}\|}$$

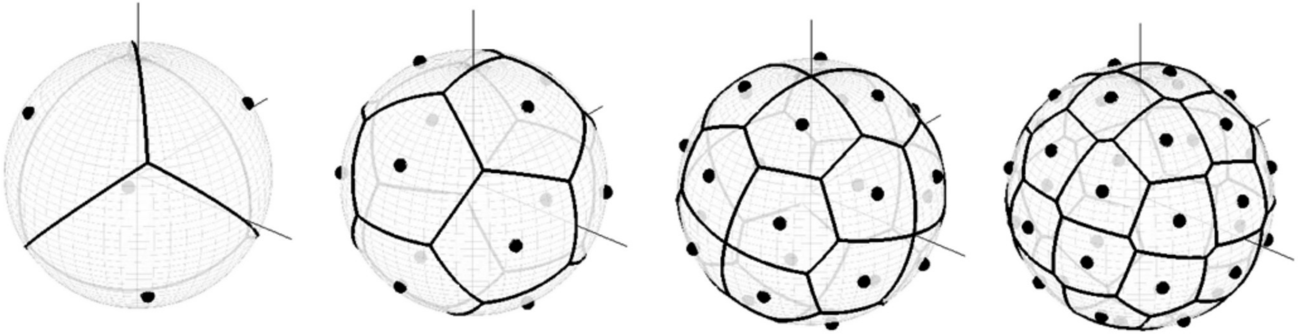
DoA (1 source) FOA

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**

HO-DirAC:

- Higher order DirAC!
  - Same analysis but on segments (“beam-forming”).
  - Features decoder, binaural and up-mixer (just different flavor).



HO-DirAC
(Politis et.al., 2015)

HOA
(9+ch)

~M sector source +
~M sector diffuse components

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR (apart).**
  - CroPaC:
    - Real-time DoA, using x-correlation of W and X (*+ post-filter*).
      - No EVD
    - Binaural decoder, FOA only.
    - Single source.
    - Too much of a good thing?
      - 4 different binaural decoders…
      - In addition to binauralizer VST.
      - Confusing.

# But before...

- Can these toolkits be used for live performance?
  - Yes
    - IEM vstplugin~ external.
  - Or, send OSC to Reaper or VSTs for sound installations.
    - Route audio via SoundFlower, etc.

# Chapter 1: Aalto Suite

- **COMPASS, HO-DirAC, CroPaC, HOSSIR.**
  - HOSSIR:
    - Use IR from HOA SMAs to create virtual loudspeaker IR.
    - Still in beta.
    - FOA mics are still quite expensive (Ch2).

# Chapter 1

What has not been done yet?

- **AmbiX**:
  - Warping with arbitrary math functions.
- **IEM**:
  - Publications explaining how these were created.
- **Aalto**:
  - Simpler musical language to help musicians. Documentation is too complex.

# Questions?

- End of first presentation.

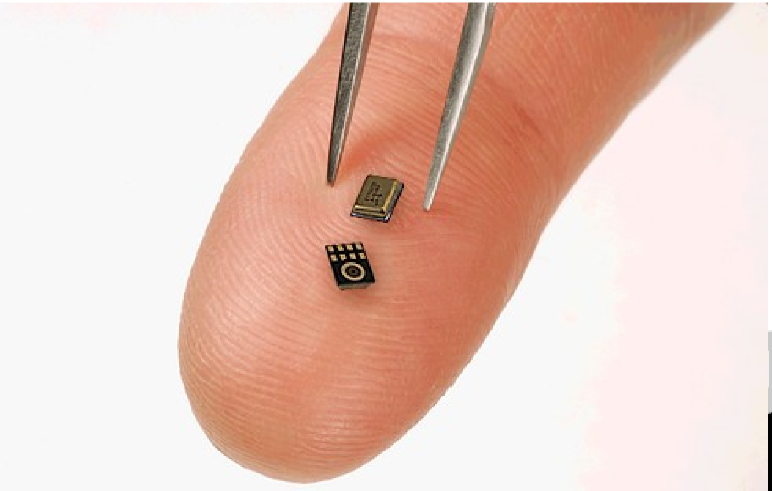
# Chapter 2 – Low-cost HOA SMA

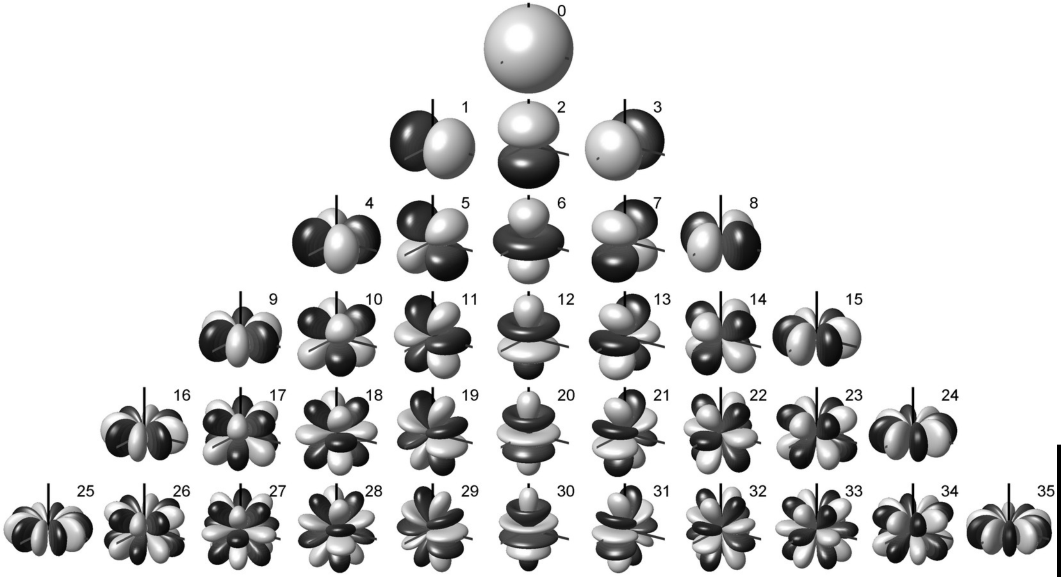
Ambisonic arrays are quite flexible and useful for ambisonic production.

However, they are very expensive.

*Can we make a cheap SMAs using MEMS capsules?*

How do we calibrate these arrays?





# Chapter 2 – Low-cost HOA SMA

Why SMAs?

No need to write down musician/microphone positions.

If wireless mic, don't need to automate position changes.

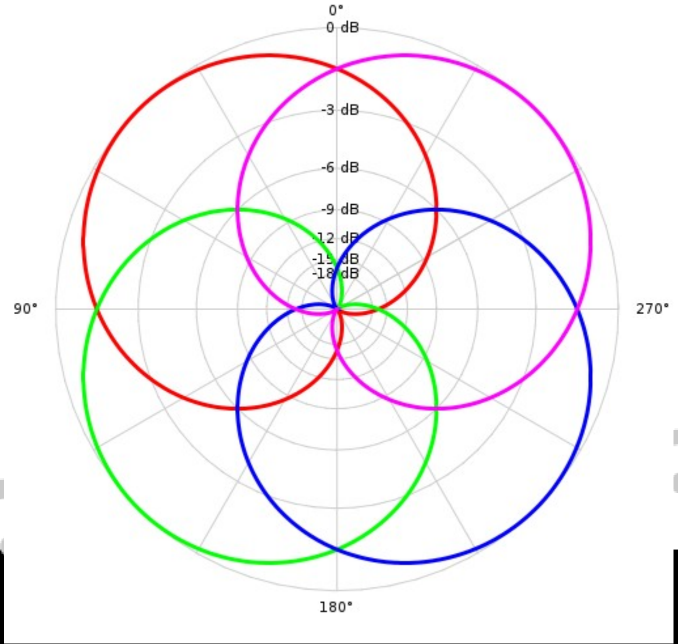
Can be used outdoors to capture nature.

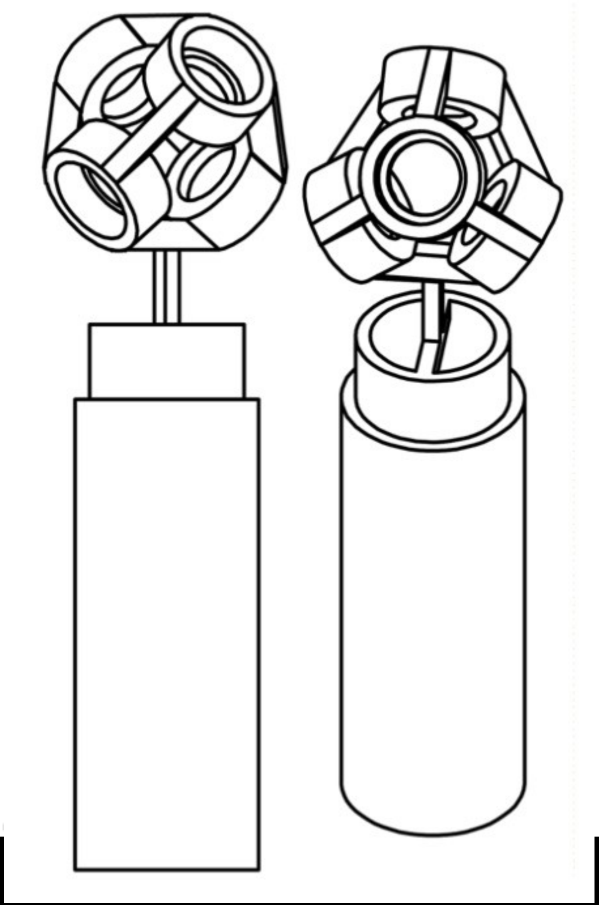
Direct integration with cameras.

**VR, AR, MR = XR**

# Chapter 2 – Low-cost HOA SMA

- In FOA we derive three orthogonal velocity microphones from 4 cardioid microphones.
- An additional pressure signal is the sum of elements (W).





# Chapter 2 – Low-cost HOA SMA

SpHEAR

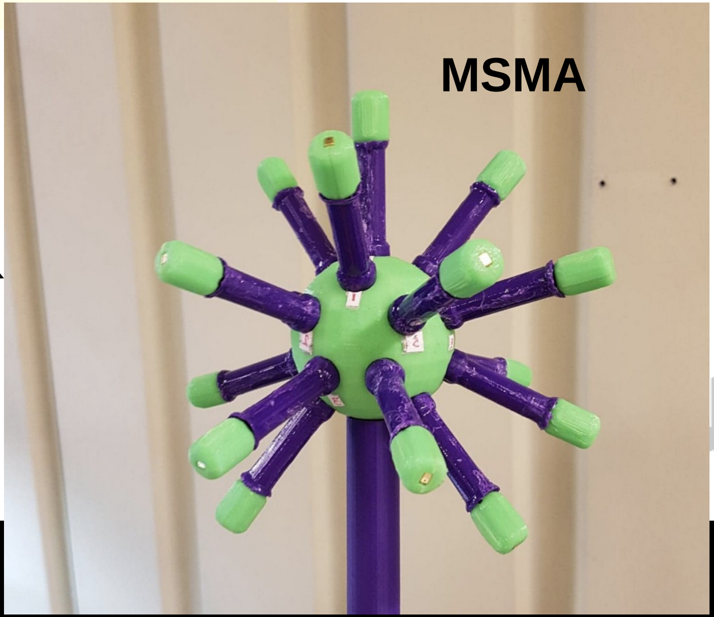
HOA is more complex

**But better spatial resolution**

**HOSMA**

Blender

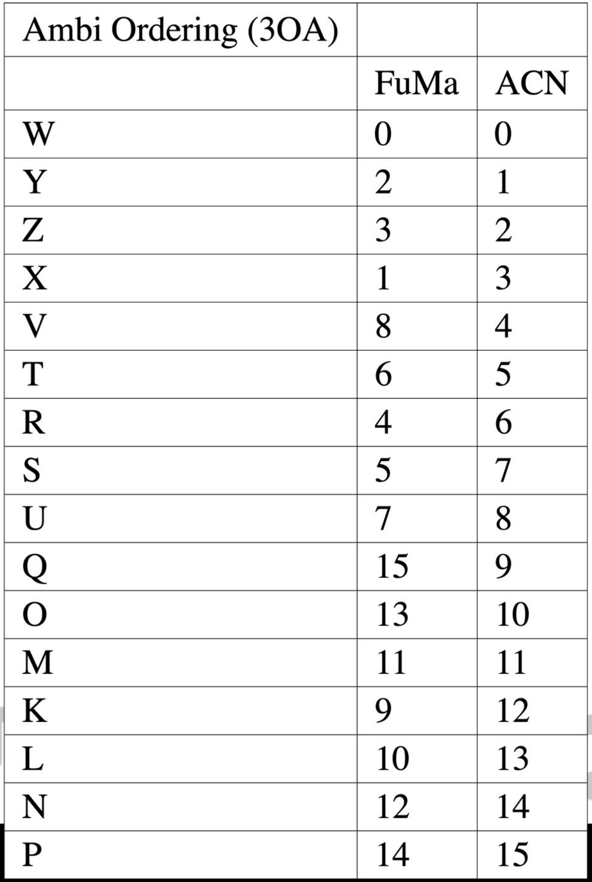
OpenSCAD



**MSMA**

# Chapter 2 – Low-cost HOA SMA

- Potential problems:
  - Ordering/labeling/naming of harmonics.
  - Normalization of harmonics.
    - We want to have consistency among researchers.
  - Ambix (Nachbar 2011) standard now.
    - SN3D + ACN ordering.



| Ambi Ordering (3OA) | | |
|---|---|---|
| | FuMa | ACN |
| W | 0 | 0 |
| Y | 2 | 1 |
| Z | 3 | 2 |
| X | 1 | 3 |
| V | 8 | 4 |
| T | 6 | 5 |
| R | 4 | 6 |
| S | 5 | 7 |
| U | 7 | 8 |
| Q | 15 | 9 |
| O | 13 | 10 |
| M | 11 | 11 |
| K | 9 | 12 |
| L | 10 | 13 |
| N | 12 | 14 |
| P | 14 | 15 |

# Chapter 2 – Low-cost HOA SMA

Some authors ignore 4pi.
In which case first four coefficients are all 1.

- SN3D
  - Forces any SH above W to **not** exceed the level of W.
    - *‘High degree of generality - the encoding coefficients are recursively computable, and the first-order components are unity vectors in their respective directions of incidence.’ (Daniel)*

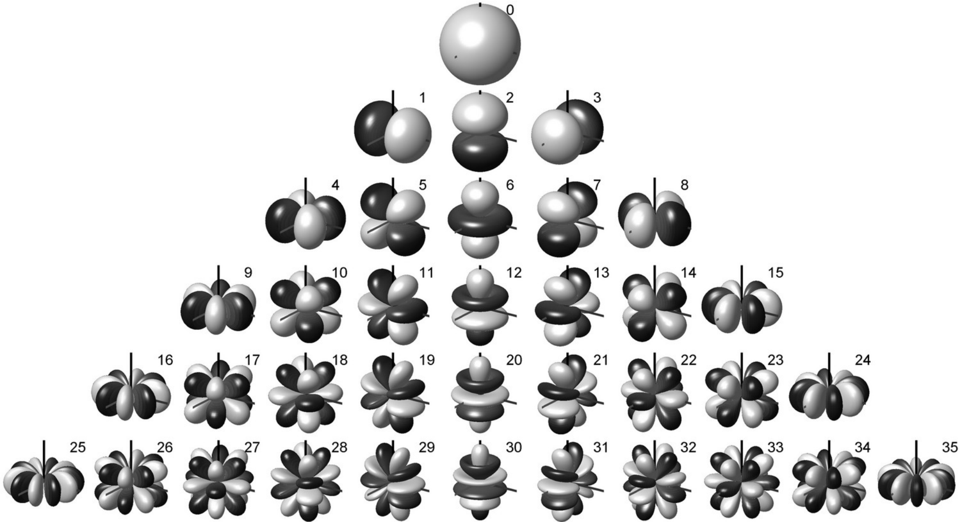
4pi term is independent of m or n.

$$N_n^{|m|} = \sqrt{\frac{2-\delta_m}{4\pi}\,\frac{(n-|m|)!}{(n+|m|)!}}$$

Ambisonic order = n, ambisonic degree = m.
**delta_m = 1 for m = 0 and 0 otherwise.**

# Chapter 2 – Low-cost HOA SMA

- Ambisonic order (n) = mathematical degree (l)
  - Here N = 5 = L. From top to bottom. 36 harmonics
    - n = [0, 1, 2, 3, 4, 5].
- Ambisonic degree (m) = mathematical order (m)
  - Ambisonic degree from -5 to +5. Left to right.

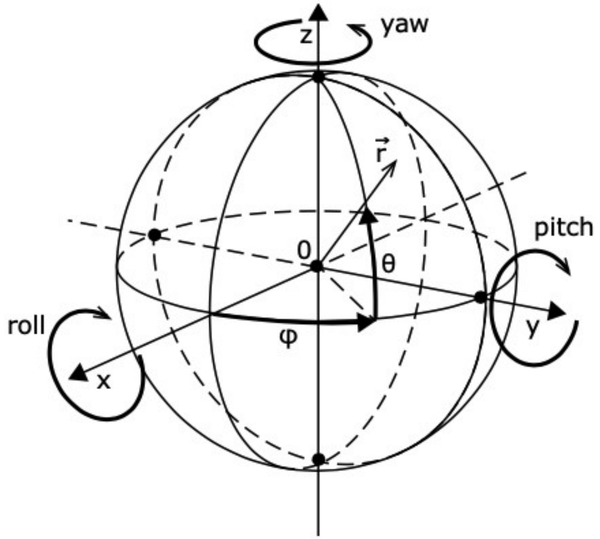


Not sure how this happened
Very confusing...

# Chapter 2 – Low-cost HOA SMA

X = Front/back, Y = Left/right, Z = Up/down.

- Coordinate system
  - ϕ = **azi**, increases counterclockwise
  - θ = **elev**, increases upwards to 90 degrees.
- Coordinate system is “standard” for ambisonics.



$$\boldsymbol{\theta} = \begin{pmatrix} \theta_x \\ \theta_y \\ \theta_z \end{pmatrix} = \begin{pmatrix} \cos\varphi\cos\vartheta \\ \sin\varphi\cos\vartheta \\ \sin\vartheta \end{pmatrix}$$

Cartesian vector

Conversion to spherical coordinate system.

$$\varphi = \arctan\frac{\theta_y}{\theta_x}, \quad \vartheta = \arctan\frac{\theta_z}{\sqrt{\theta_x^2+\theta_y^2}}$$

# Chapter 2 – Low-cost HOA SMA

$$Y_n^m(\phi,\theta) = N_n^{|m|} P_n^{|m|}(\sin(\theta)) \begin{cases} \cos(|m|\phi) & \text{if } m \geq 0 \\ \sin(|m|\phi) & \text{if } m < 0 \end{cases}$$

- Real-valued SH
  - Used to encode/decode
- P = ass. Legendre functions.
  - Solutions to wave equation in spherical domain.
  - In ambisonics we use all positive harmonics
    - No *Condon-Shortley*.
  - Can use recurrence to calculate P.

# Chapter 2 – Low-cost HOA SMA

Green is only calibration.
Others are about FOSH.

- 5 projects we want to talk about:
  - Middlicott et al. (@Derby) - A-format calibration methods
  - Gonzalez et al. (@Aalto) - MSMA
  - Moschner et al. (@Köln)- HOSMA
  - Lopez-Lezcano (@CCRMA) – SpHEAR
  - Zalles (@UCSD) - z-array

tux

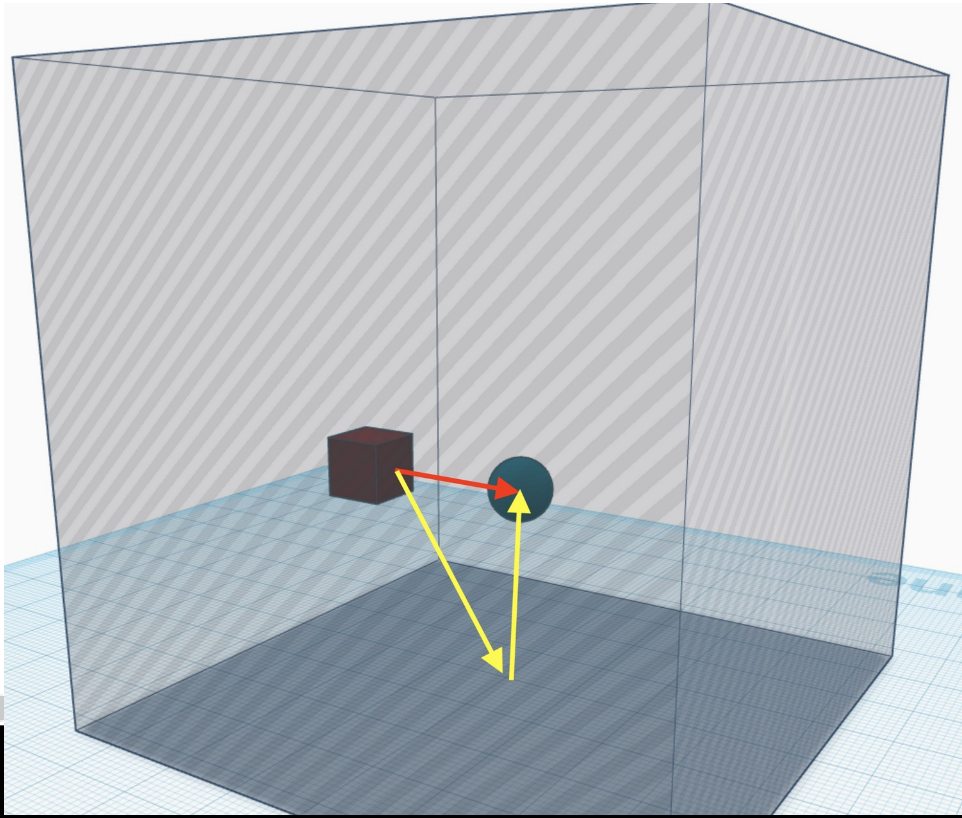
# Chapter 2 – Low-cost HOA SMA

- Middlicott et al. (@Derby) - A-format calibration methods
  - Planar 2OA array (5 capsules) evaluated in horizontal axis.
  - 4 different approaches to capsule calibration.
    - 1. Calibration by 1/3rd Octave Average Gain Matching,
    - 2. Calibration to a Specific Capsule’s On-Axis Frequency Response,
    - 3. Calibration to a Flat Frequency Response, and
    - 4. Calibration by Diffuse Field Equalization.

# Chapter 2 – Low-cost HOA SMA

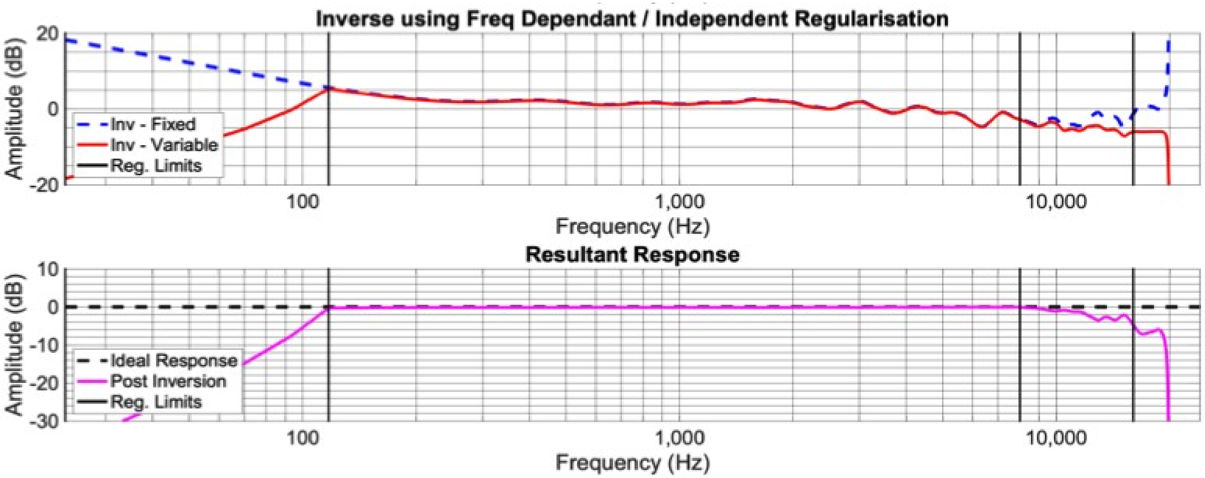
- Middlicott et al. (@Derby)
  - Speaker compensation:
    - Determine shortest reflection path
    - Trim IR
    - Window IR
    - Invert IR
      - w/ regularization.

https://mehlau.net/audio/floorbounce/



# Chapter 2 – Low-cost HOA SMA

- Speaker compensation:
  - ESS technique
  - Flat mic to find speaker response
  - Inverse filtering (Nelson/Kirkeby)
    - Target = FFT of dirac delta.
  - Epsilon = regularization vec.



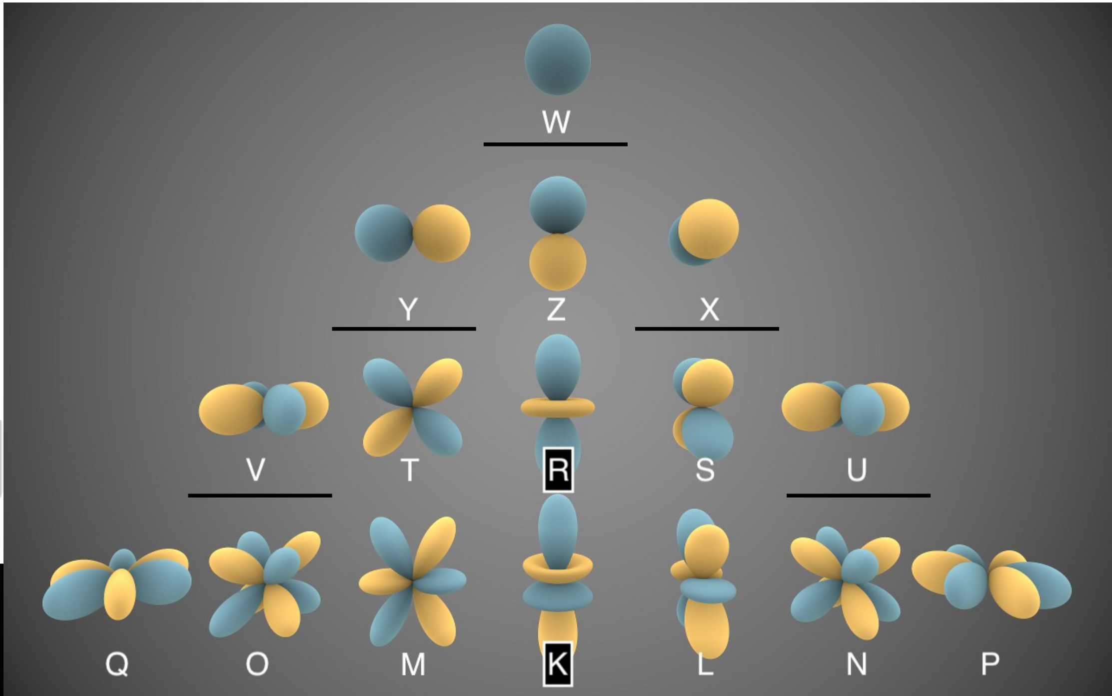
$$H_{inv}(\mathrm{k}) = \frac{H_{\mathrm{Target}}(k) \cdot \mathrm{Conj}(H(k))}{\mathrm{Conj}(H(k)) \cdot H(k) + \epsilon(k)}$$

# Chapter 2 – Low-cost HOA SMA

- Dense grid of IR.
  - In this case 90 measurements (2 degree increments)
    - Up to 180 degrees, then assume symmetry.
  - Turntable: Outline ET250-3D ($$$)
  - 10-15 second ESS, 2 meters away, total 450 IR.

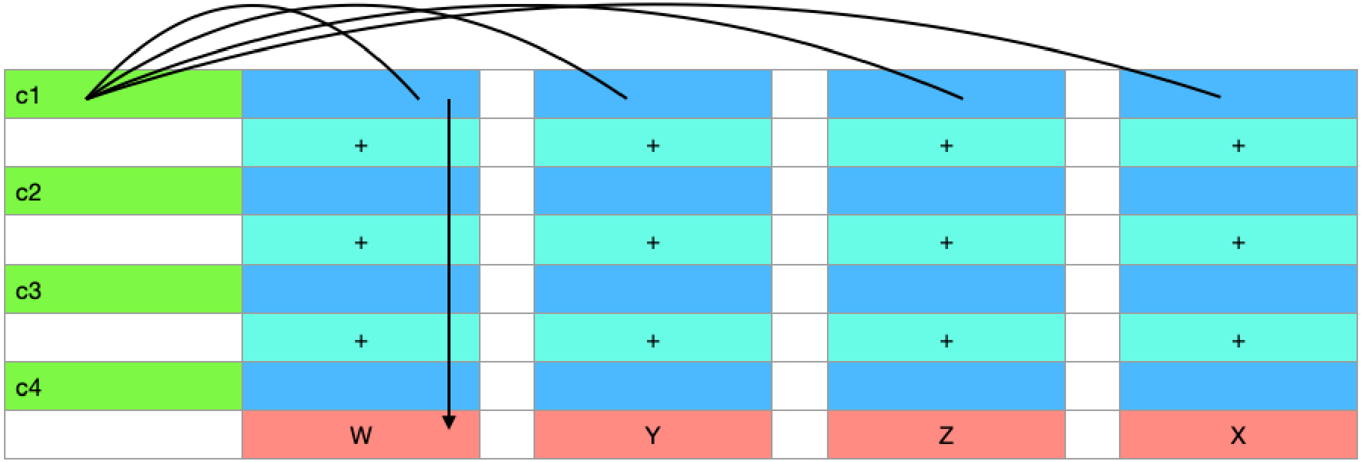
# Chapter 2 – Low-cost HOA SMA

- With all IR captured, 4 calibration methods are tested.
- The SH response is plotted to determine effectiveness.
- 2N+1 harmonics.
  - WYX + VU
  - Horizontal only



# Chapter 2 – Low-cost HOA SMA

- Encode SMA sigs:



| | | | | |
|---|---|---|---|---|
| c1 | | | | |
| | + | + | + | + |
| c2 | | | | |
| | + | + | + | + |
| c3 | | | | |
| | + | + | + | + |
| c4 | | | | |
| | W | Y | Z | X |

Regular = use SH as is.
Irregular = invert.

# Chapter 2 – Low-cost HOA SMA

## 1. Calibration by 1/3rd Octave Average Gain Matching

- Split spectrum of 5 on-axis responses into 1/3rd octaves.
- Avg mag over freq range.
- Use smallest magnitude to attenuate all other channels.

- Simple method and only requires 5 on-axis responses.

*This is simpler for horizontal only arrays!*

# Chapter 2 – Low-cost HOA SMA

- 4. Calibration by Diffuse Field Equalization.

- Uses all 450 IR measurements.

- The magnitude of all 90 measurements (per capsule) are averaged.

- These are used as targets to generate 5 inverse filters.

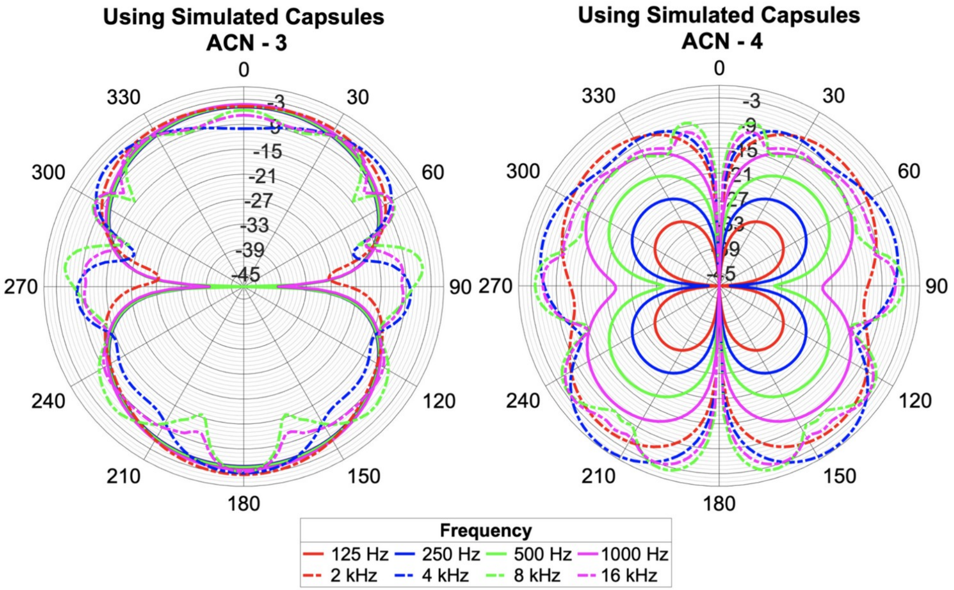
$$\mathrm{DFR(c)} = \sqrt{\frac{1}{D}\sum_{d=1}^{D}\left|FFT_{(c,d,k)}\right|^{2}}$$

Capsule = c
Direction = d
Bin = k

# Chapter 2 – Low-cost HOA SMA

- The SH signals are plotted up to 4kHz where spatial aliasing warps the response.
- No B-format calibration in this paper.
  - Simulated and ideal responses are also plotted.
  - Raw responses w/o filtering are also plotted.

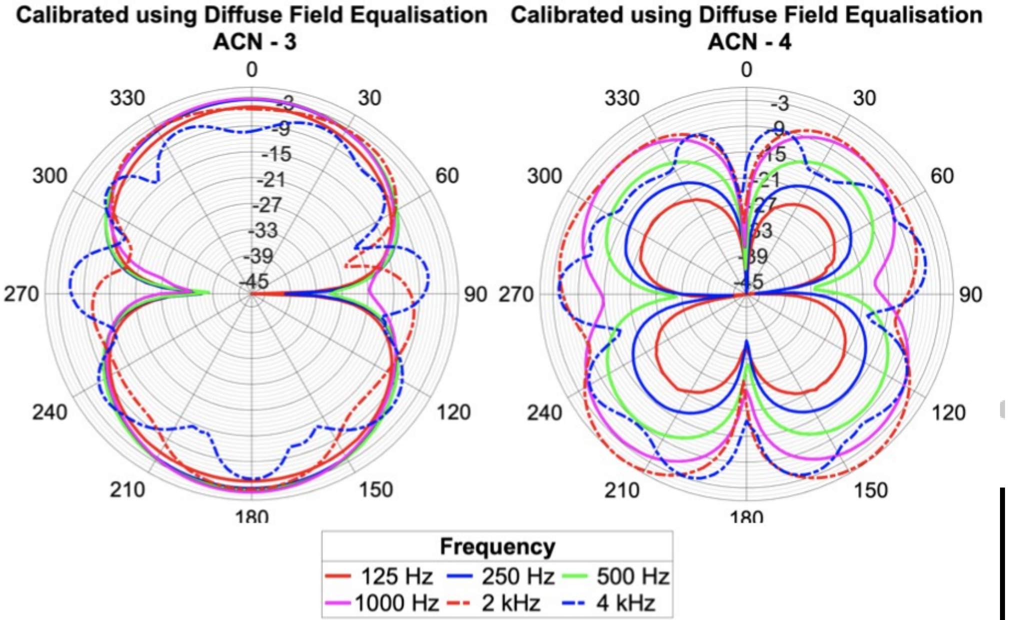
If capsules were perfectly flat and cardioid.
This would be the result.



Middlicott, Charlie, and Bruce Wiggins. "Calibration approaches for higher order ambisonic microphones." Audio Engineering Society, 2019.

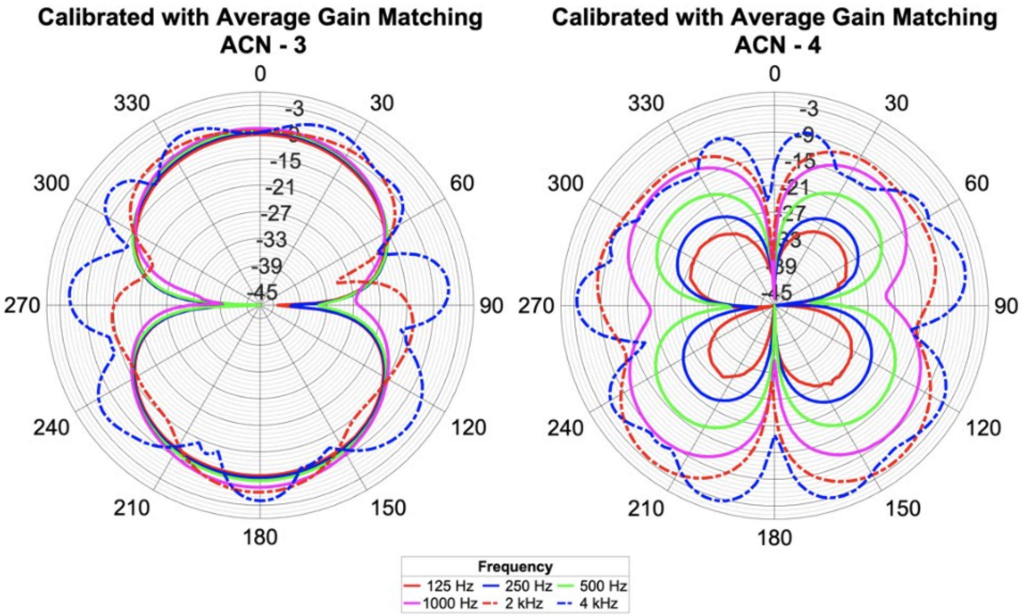
# Chapter 2 – Low-cost HOA SMA

- Paper concludes that average gain matching (right image) and DFR equalization (left image) yield the best results.
- Average gain matching requires far fewer measurements!



Calibrated using Diffuse Field Equalisation ACN - 3

Calibrated using Diffuse Field Equalisation ACN - 4



Calibrated with Average Gain Matching ACN - 3

Calibrated with Average Gain Matching ACN - 4

# Chapter 2 – Low-cost HOA SMA

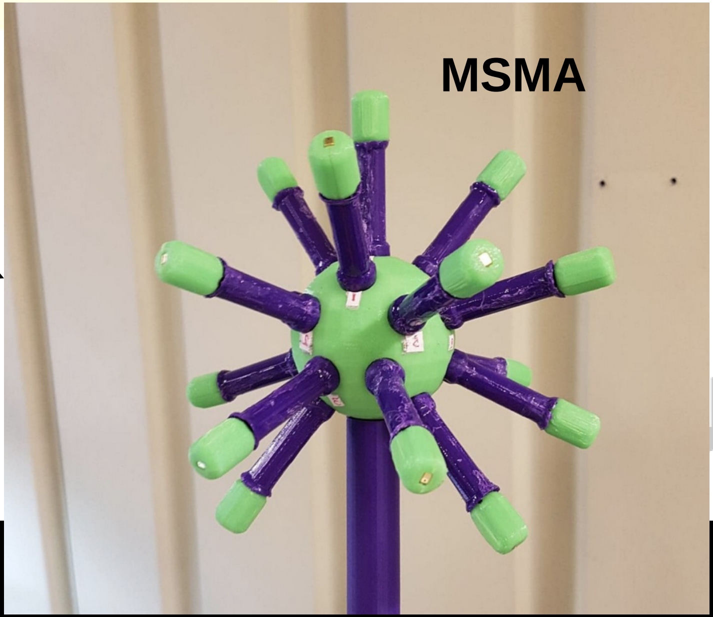
- Paper concedes HOA equalization needed to correct SH above aliasing freq.
- No radial filters.
- Methods only for horizontal arrays
  - Considerations for SMAs are more complicated.
  - Access to elevation angles?

On-axis meas.

# Chapter 2 – Low-cost HOA SMA

SpHEAR

OpenSCAD



**MSMA**

**HOSMA**

Blender

# Chapter 2 – Low-cost HOA SMA

- MSMA – Modular Spherical Microphone Array (Aalto)
  - MEMS capsules, analog, not best SNR
  - Why modular? We can change attachments for varying and multiple radii.
  - 19 total sensors, on the faces of an icosahedron (20 face regular platonic solid) **3OA**.
  - Low-cost 3D printing and OpenSCAD.

# Chapter 2 – Low-cost HOA SMA

- MSMA – Modular Spherical Microphone Array (Aalto)
  - **No measurements taken**, only simulations.
    - Results showed spatial correlation was dependent on frequency and array radius as expected.
      - Using “Spherical Array Processing” library (by Politis).

Spatial correlation measures how closely the simulations match the ideal responses.

# Chapter 2 – Low-cost HOA SMA

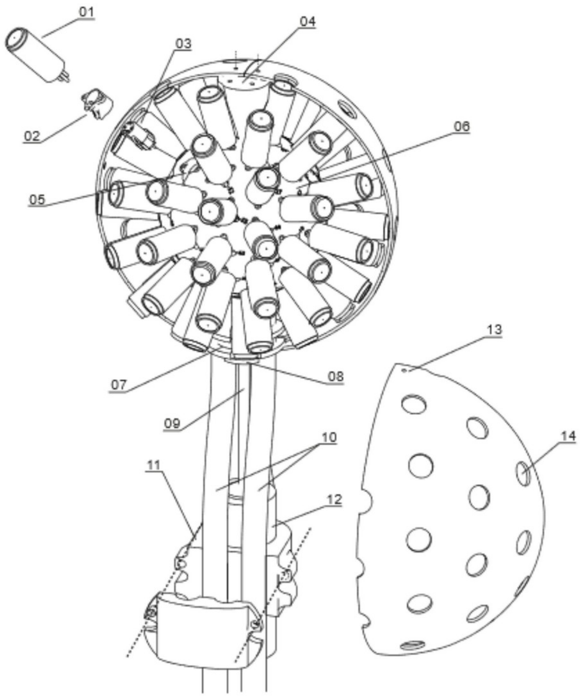
- MSMA – Modular Spherical Microphone Array (Aalto)
  - No equalization or radial filters.
  - Other findings:
    - The OpenSCAD code could not be located.
    - The PCBs and the rendered STL models can be downloaded.
    - Switching these capsules for digital MEMS might reduce cost considerably.
      - 16 channel interface (3OA) gets quite pricey ~$500 (at least).

# Chapter 2 – Low-cost HOA SMA

- HOSMA – Higher Order Spherical Microphone Array (Köln)
  - 64 chan. HOA SMA
    - Sennheiser ECMs **(7OA).**
  - Blender+ Python scripting.
    - Partially reconfigurable.

# Chapter 2 – Low-cost HOA SMA

- HOSMA – Higher Order Spherical Microphone Array (Köln)
  - Based on a Fliege grid.
    - Follows (N + 1)^2 rule!
  - Analog capsules = costly interfaces.
  - Sennheiser capsules <u>not available via retailer?</u>



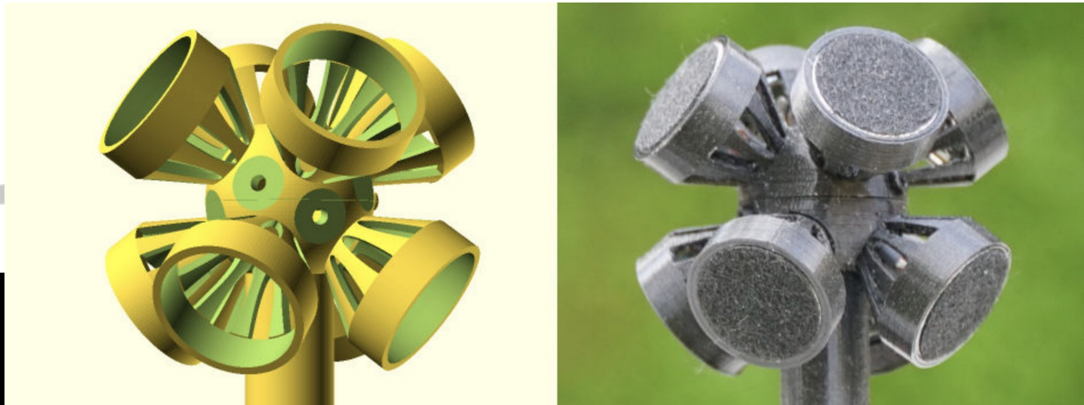
01 - Microphone KE14
02 - XLR Adapter
03 - Female XLR Socket
04 - Top Mount for outer sphere
05 - Hole for XLR Adapter
06 - Innercore
07 - Bottom Mount
08 - Fixing mount
09 - Lead Screw 8mm
10 - 32 * 2 * 0.14 mm$^2$ Multicore Cable
11 - Mount for Multicore Cables
12 - Steel Mount for Tripod
13 - Hole for M3 screw
14 - Outer Sphere

# Chapter 2 – Low-cost HOA SMA

- Findings:
  - Provide a good overview of EMI-shielding methods.
    - Copper tape and graphite spray applied to PCB tubes.
    - Conductive PLA material used for rigid baffle.
  - Models can be modified to incorporate cheaper components (Blender = FOSS).
  - Fliege grid extremely well-suited for ambisonics.
  - SOFiA toolbox useful to generate radial filters and simulate responses.

# Chapter 2 – Low-cost HOA SMA

- SpHEAR – Spherical Harmonic EAR.
  - Best documented project.
  - Cheap 3D printing solution + OpenSCAD.
  - Public PCBs and commercial capsules.
    - Primo capsules are expensive
    - Cardioid **but** analog.
  - Calibration far more sophisticated.
  - All FOSS!
    - GPL licensed.



# Chapter 2 – Low-cost HOA SMA

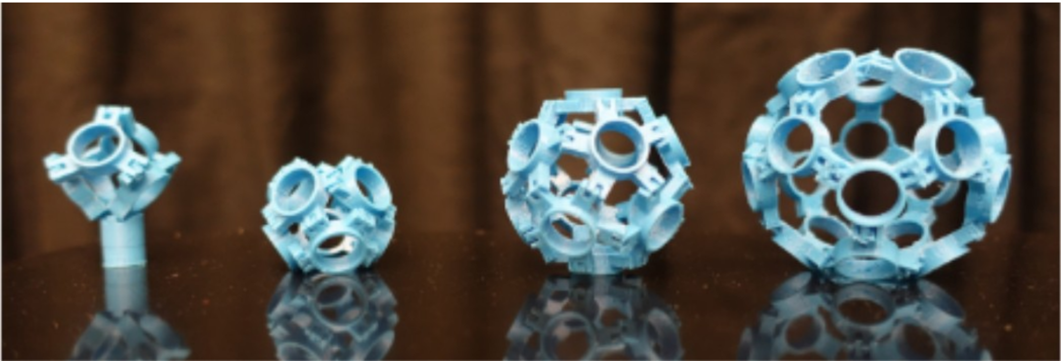
- SpHEAR – Spherical Harmonic EAR.
  - Robotic arm used for calibration
    - How can we calibrate without one?
  - Can we use cheaper components?
  - B-format equalization!

# Chapter 2 – Low-cost HOA SMA

- Considerations:
  - 3D printing (SLS, SLA or FFF?)
    - FFF is the cheapest, but models need to be simple.
  - PCBs + electronics (cost?)
    - Diodes, resistors, capacitors, capsules, other necessary tools?
  - Does it require individual calibration?
  - Recording interface?

Anechoic chamber

Tetra, octa, dodeca, icosa.



# Chapter 2 – Low-cost HOA SMA

- FFF
  - Cheapest, but models need to be simple.
  - Printer cost ~$300.
- SLS
  - Capable of creating high-detail parts in materials like metal
  - Parts suitable for mechanical purposes.
  - Printer cost ~$5000.
- SLA
  - Produce parts that are very high in detail, but brittle and not suitable for any mechanical purposes.
  - Printer cost ~$500.

ANET A8 DIY FDM 3D PRINTER

# Chapter 2 – Low-cost HOA SMA

- Why does 3D printer matter?
  - Prototyping >>> iterate quickly.
    - Turn around time for online services is slow and $.
    - Universities w/ fewer resources can still make these designs.
- Cheaper printers can’t produce small details.
  - Consider tolerance in design!

ANET A8 DIY FDM 3D PRINTER

# Chapter 2 – Low-cost HOA SMA

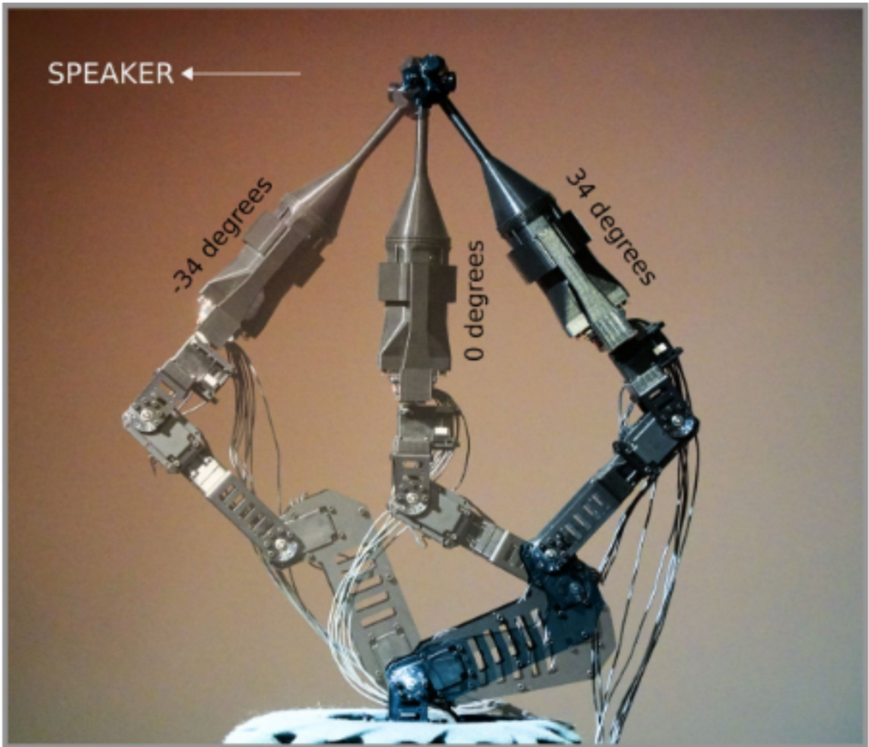
- PCBs + electronics (cost of components?)
  - Other tools:
    - Most fab-labs have soldering irons and wire cutters.
    - How about reflow ovens? (z-array)
    - How about laser cutters? (z-array)

Primo 200
$446.48 for 8 (before shipping and handling)

# Chapter 2 – Low-cost HOA SMA

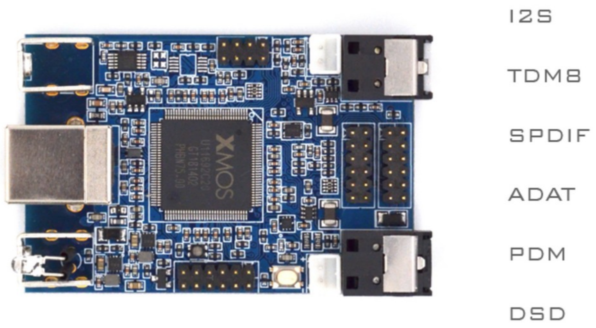
- Does it require individual calibration?
- Components:
  - Speaker (full frequency)
  - Flat microphone
  - Rotating table
- How effective is the system without individual calibration?
- Manufacturing tolerance
  - +/- 3 dB Primo
  - JND 1dB

Trossen robotics $2500



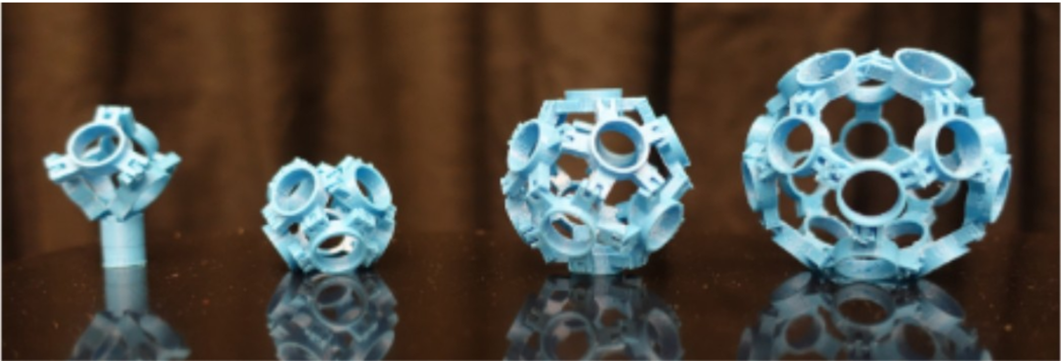
# Chapter 2 – Low-cost HOA SMA

- Recording interface?
  - HOA = expensive ADC.
  - Can we find alternatives?
- Digital MEMS have internal ADC!
  - MCUs/FPGAs designed to interface with these sensors.
    - Provide right voltage 3V
      - Instead of phantom power
    - Portable and small.



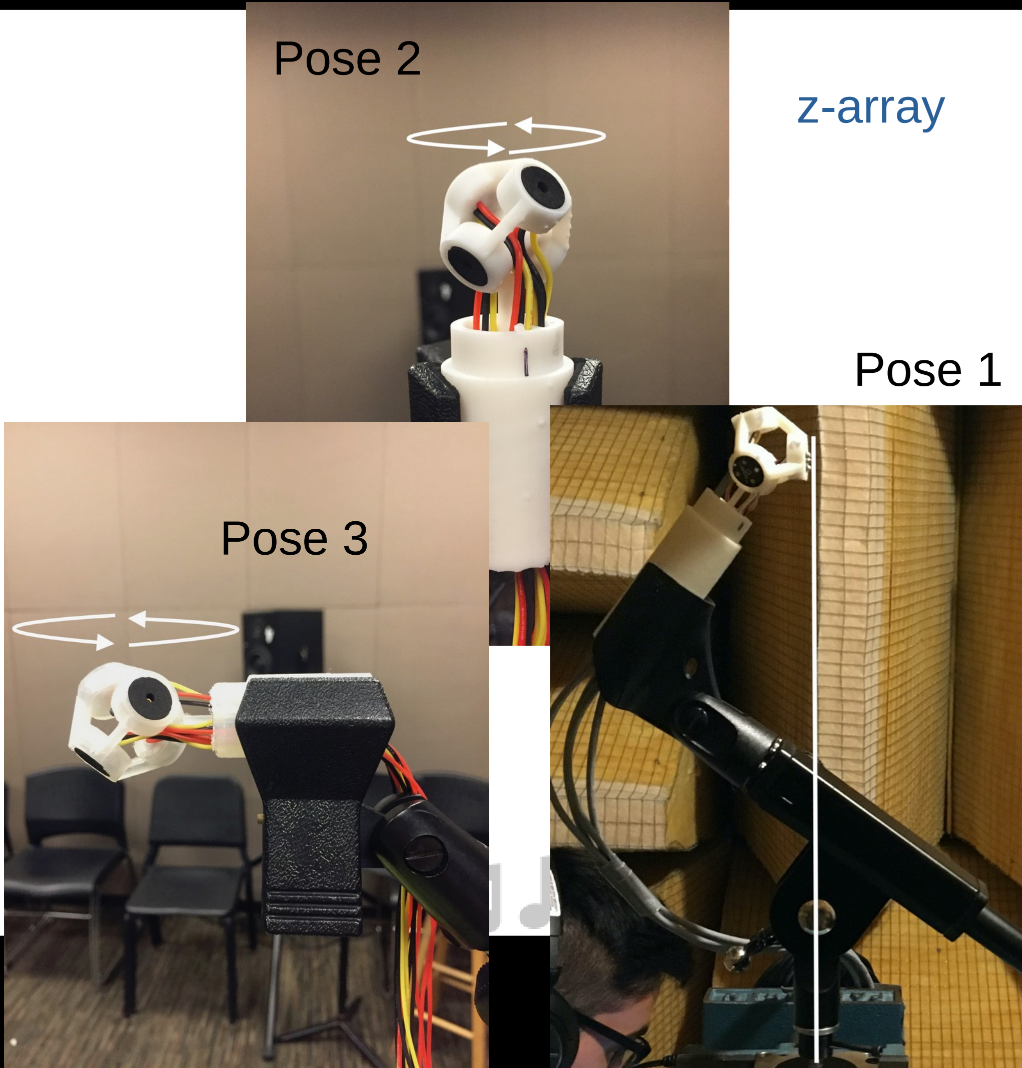
MCHStreamer
8 I2S Mics
16 PDM Mics

Tetra, octa, dodeca, icosa.



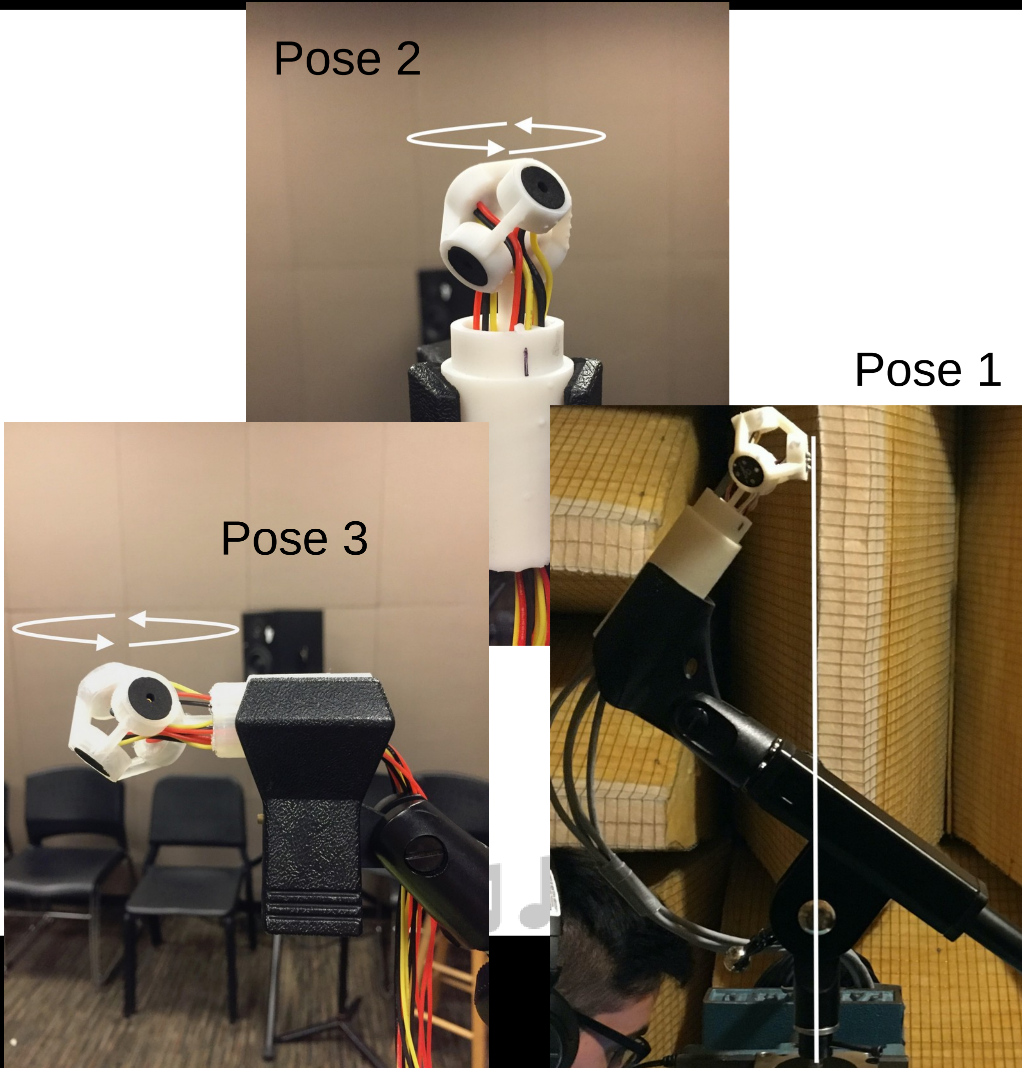
# Chapter 2 – Low-cost HOA SMA

- SpHEAR – Spherical Harmonic EAR.
  - Calibration
    - Reduce the calibration time as much as possible.
      - A-format
      - B-format
    - How many poses do we need?
    - Can we minimize the cost of calibration?



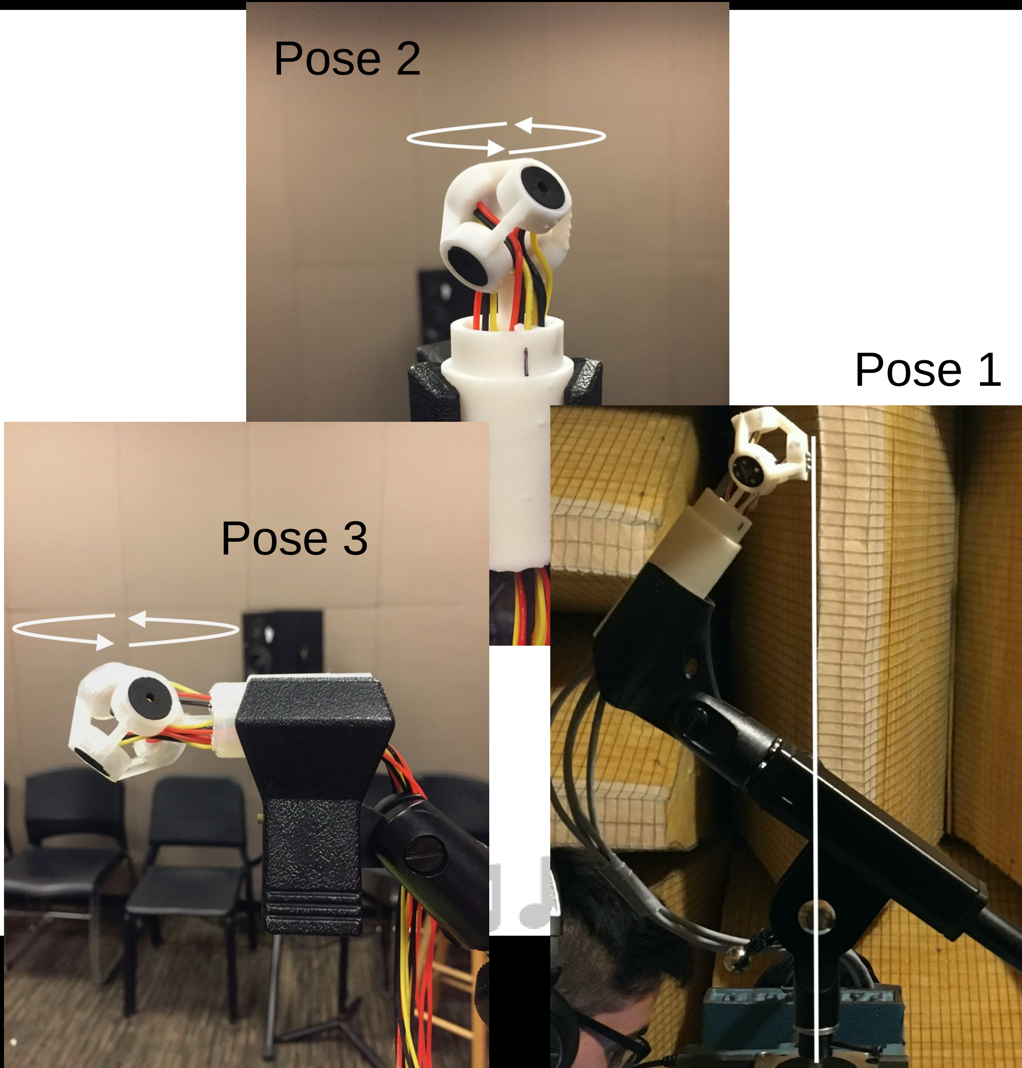
# Chapter 2 – Low-cost HOA SMA

- SpHEAR – Spherical Harmonic EAR.
  - A-format Calibration
    - In SpHEAR, A-format calibration done with pose 2.
    - 3 measurements from incident direction used.
    - Cardioid weighted



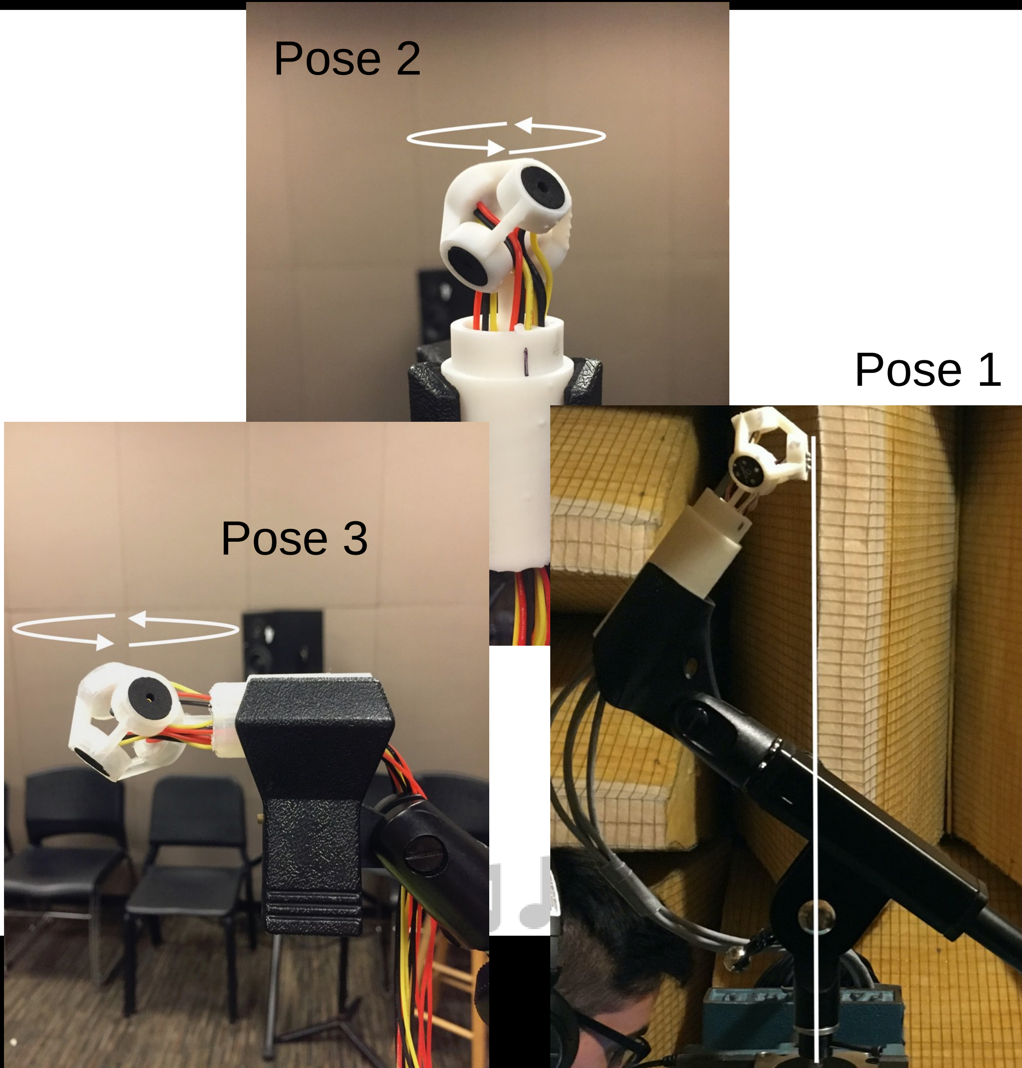
# Chapter 2 – Low-cost HOA SMA

- Encoding
  - Before we can B-format equalize, we need to encode.
  - Use SH matrix, evaluated at capsule positions.
  - Invert the matrix
    - (if irregular/non-uniform layout).



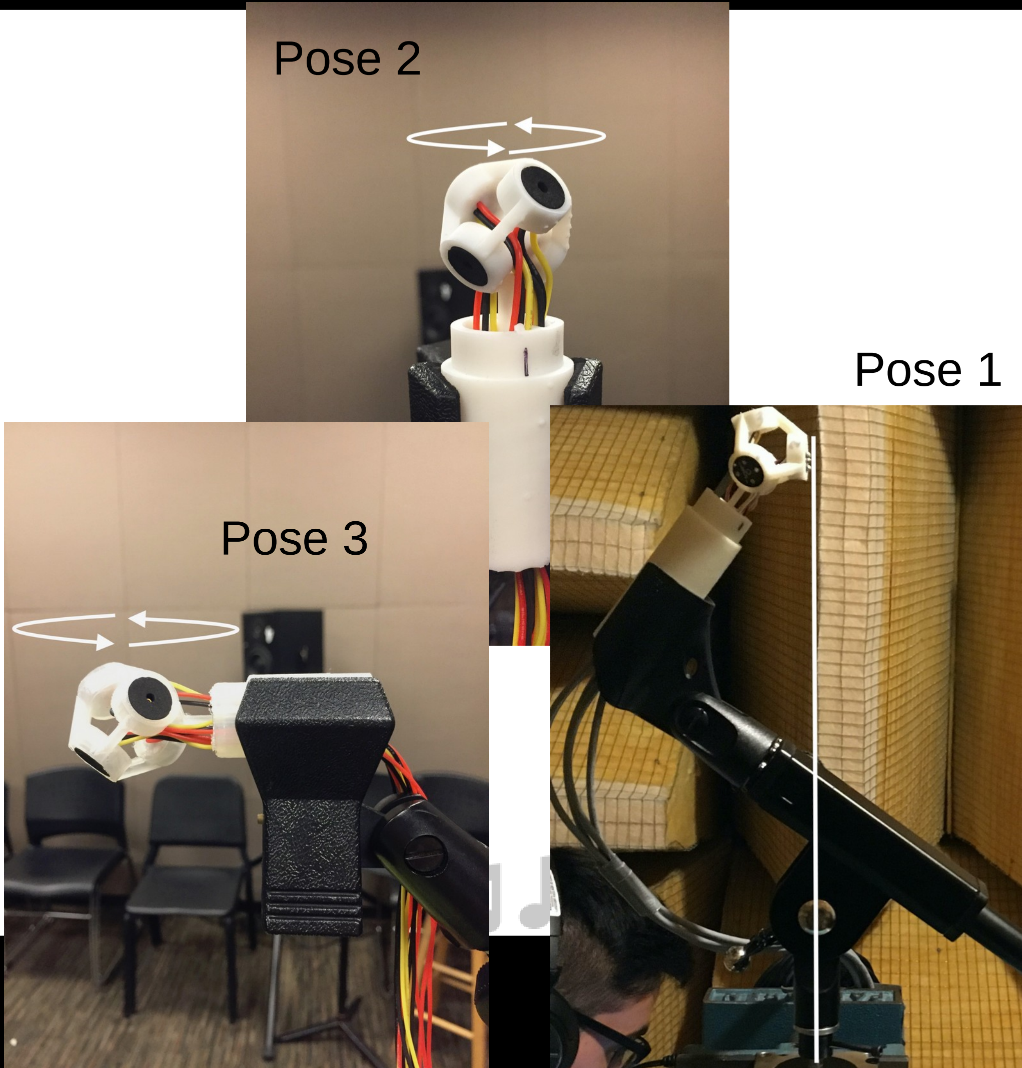
# Chapter 2 – Low-cost HOA SMA

- Encoding
  - In SpHEAR, the matrix is a LS solution based on measurement powers.
    - Averaged over “coincident” bandwidth (200-11.25kHz)



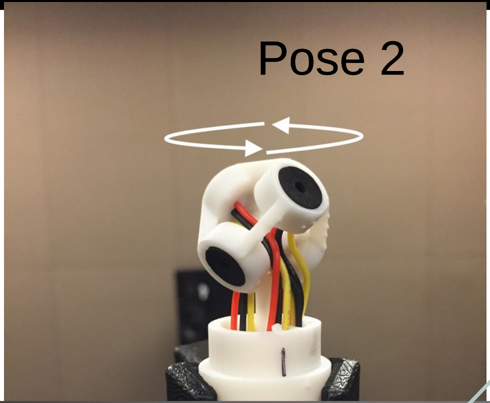
# Chapter 2 – Low-cost HOA SMA

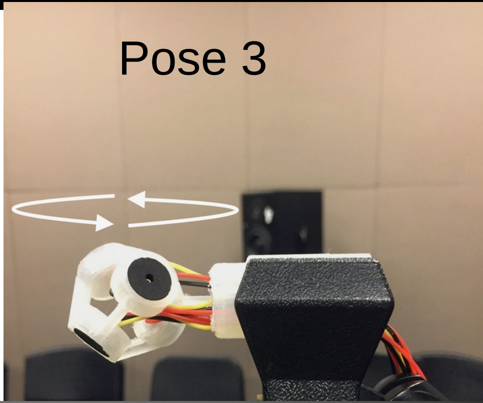
- B-format Calibration
  - Above coincident bandwidth we need to equalize to reduce aliasing effects.
  - In SpHEAR measurements around peak of SH are used.
  - However, we need elevation changes to sample certain harmonics
    - Pose 3: Z harmonic

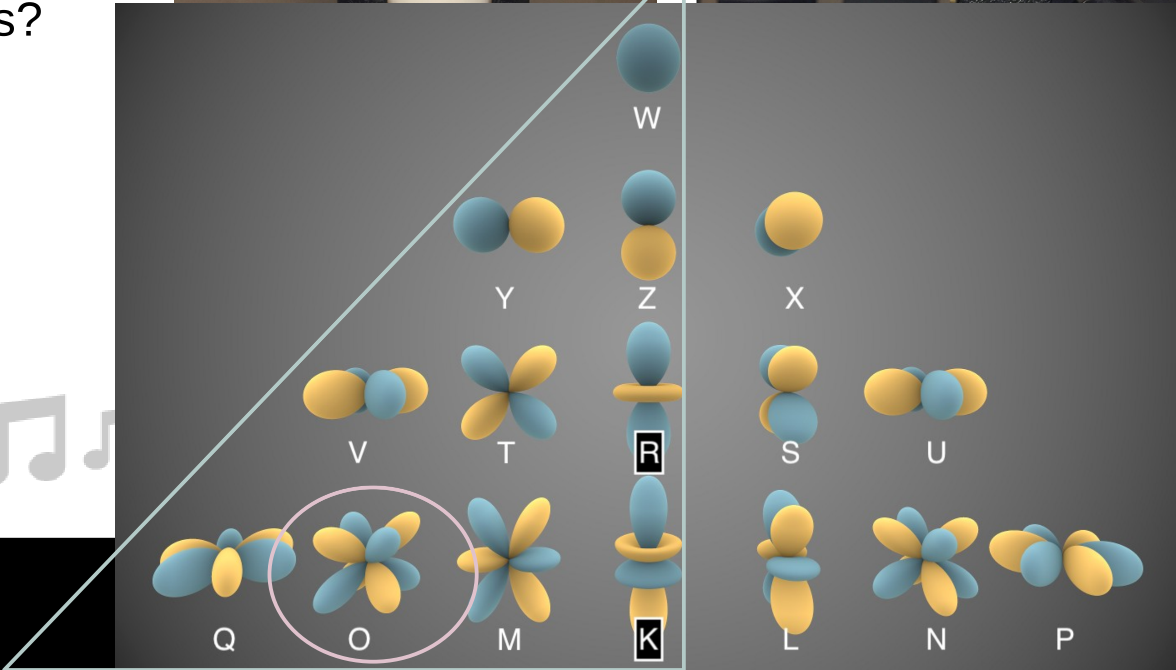


# Chapter 2 – Low-cost HOA SMA

- B-format Calibration
  - Pose 2+3:
    - Y and X (peak at 90 and 0 degrees).
    - Z at 0 degrees [pose 3]
    - Do we need *all* measurements?
  - Pose 2+3
    - Can’t sample S
    - Assume symmetry?
  - Pose 2+3
    - Can’t sample O
    - Need more poses!



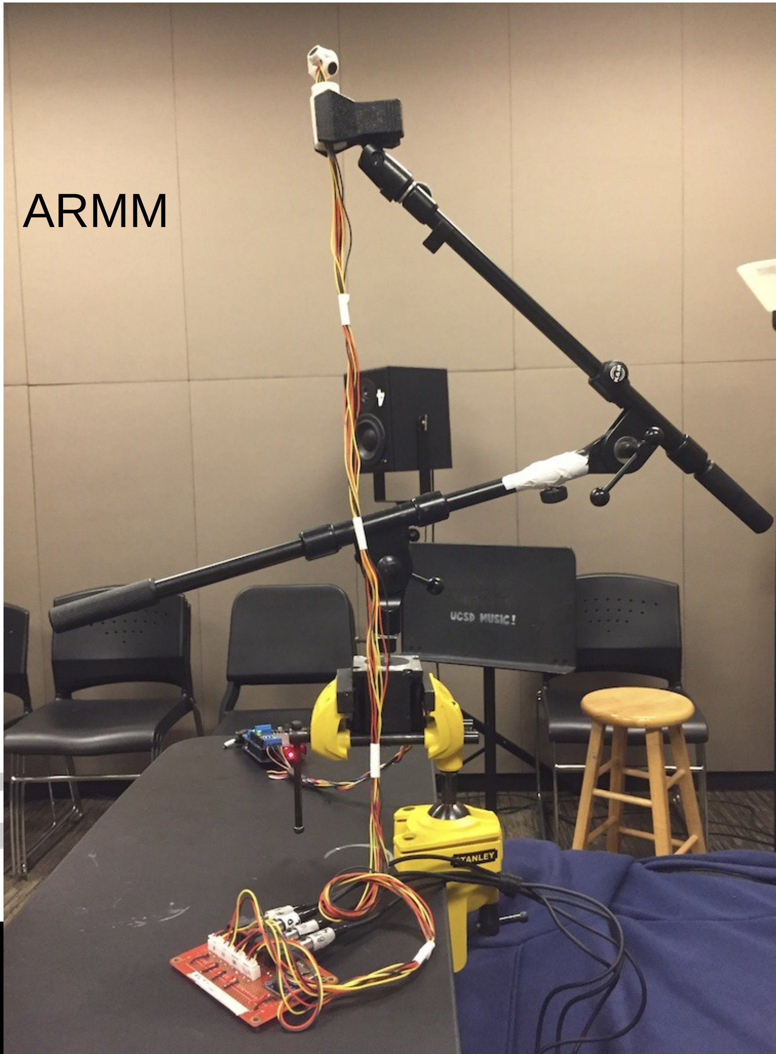




# Chapter 2 – Low-cost HOA SMA

- SpHEAR – Spherical Harmonic EAR.
  - B-format Calibration
    - Cheap calibration system
      - Clamp
      - Stepper motor
      - Arduino + Motor shield
      - Custom attachment (mic stand connector)
      - 2 boom poles
      - 1 alligator clip.

**Auto rot mic mount = armm**


ARMM

# Chapter 2 – Low-cost HOA SMA

- Cheap calibration system
  - Relies on ScanIR (NYU Matlab project)
  - Switch this out for a Pd patch.
    - ESS + recording + Arduino
- Create custom CAD model for stepper!

# Chapter 2 – Low-cost HOA SMA

- Only horizontal rotation.
- Need to manually change pose
- Need to manually ***center*** the array (3D)
  - These problems make it imperfect, but works reasonably well.
- Cost is minimal and uses common parts ~$100.

# Chapter 2 – Low-cost HOA SMA

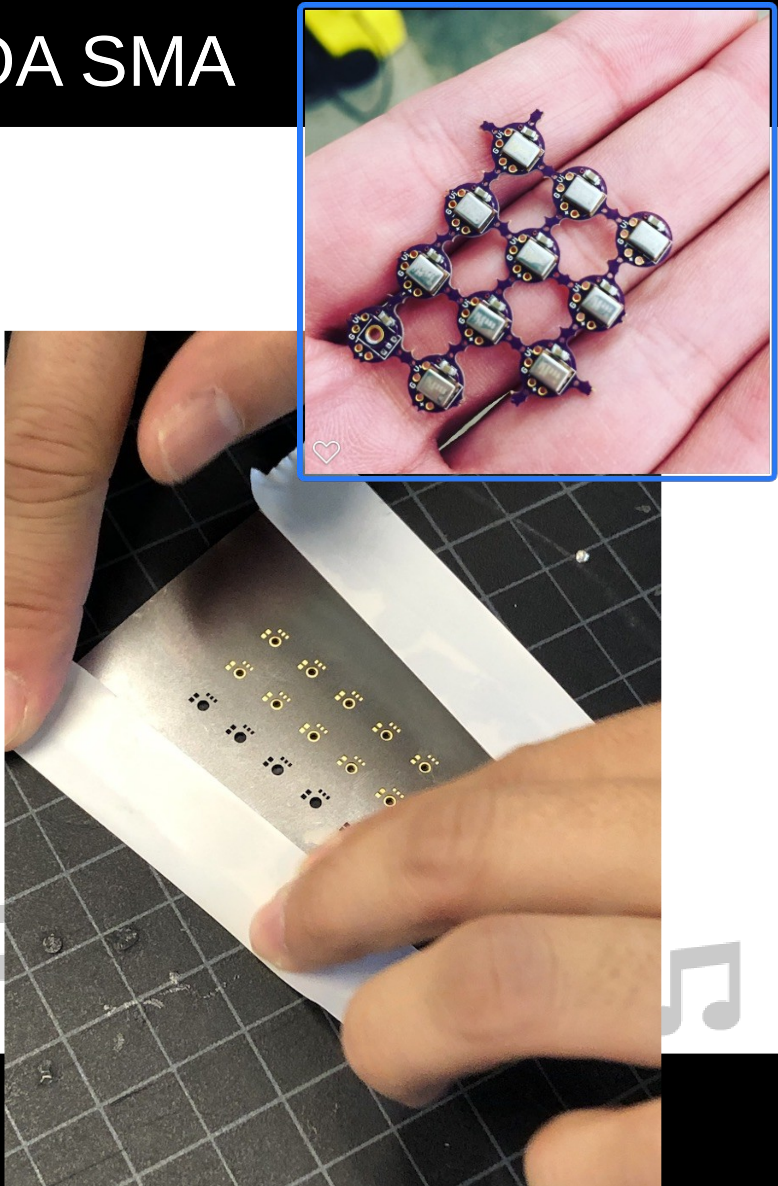
- Z-array (first prototype)
  - FOA Mics created with SolidWorks **(Yigal Kamel)** + TinkerCAD **(Me)**
  - PCBs created in Eagle CAD **(Charlie Mydlarz)** for ICS-40720
    - and ICS-43434 (I2S)
  - Reflow oven required

# Chapter 2 – Low-cost HOA SMA

- Z-array (first prototype)
  - Stencils created to facilitate PCB mounting
  - Laser cutter for foam rings
  - External battery
    - Coin cell + protoboard + Adafruit board

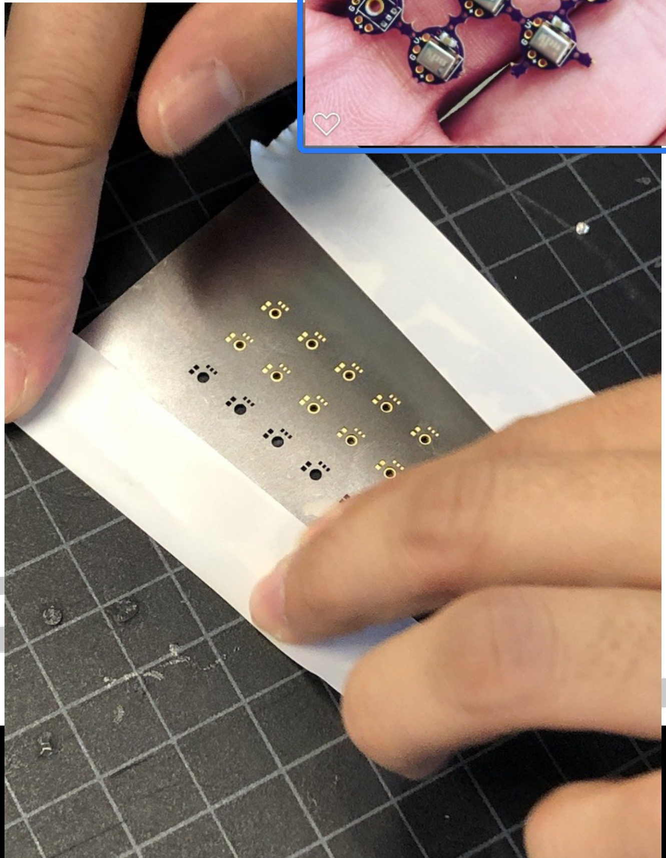
# Chapter 2 – Low-cost HOA SMA

- Z-array (first prototype)
  - Hard to print with cheap printers (bad).
  - Eagle files can be imported to open source software for editing (good).
  - Cheap reflow can be accomplished with hot-plate or skillet (some solder might be toxic – lead!!!)



# Chapter 2 – Low-cost HOA SMA

- Z-array (first prototype)
  - PCB stencils make placing components much easier, fairly cheap. (still requires a steady hand)
  - Need foam rings to protect capsules.
    - Actually very hard problem.
    - Need to find right material!!!
  - Next device will be USB powered.



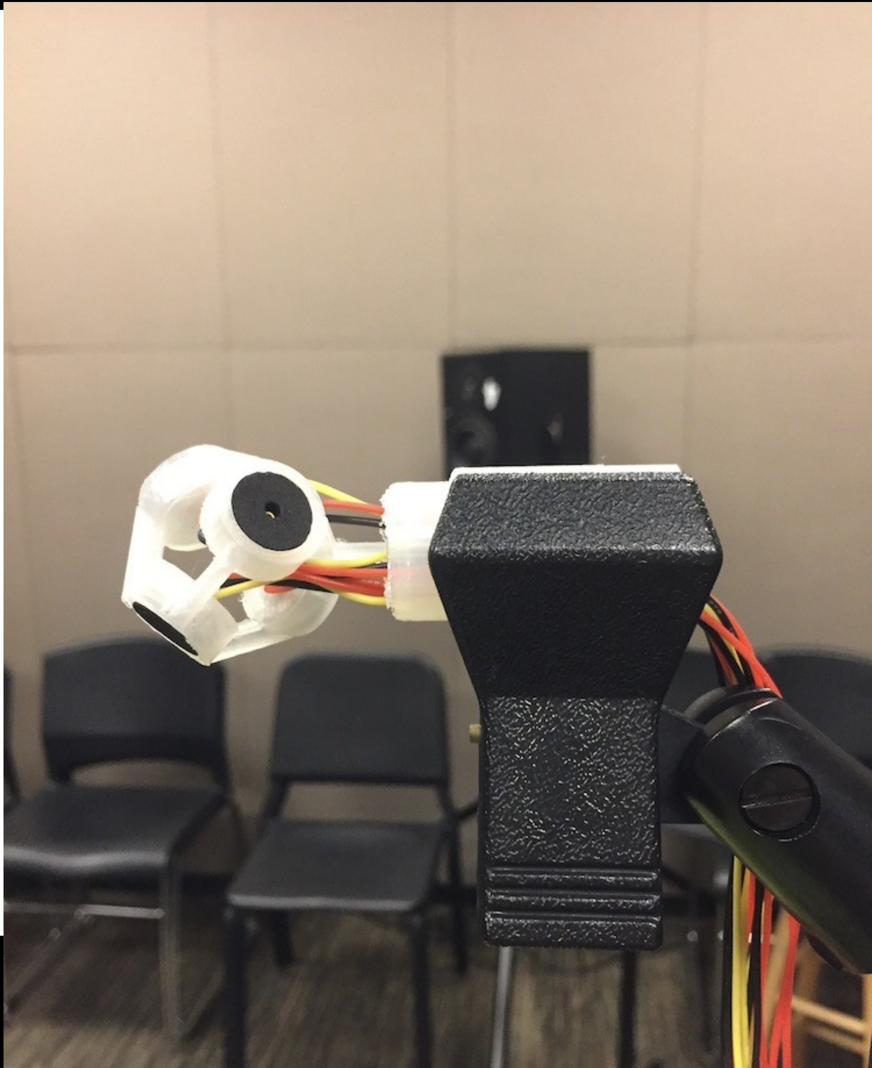
# Chapter 2 – Low-cost HOA SMA

MEMS are very fragile, the sound hole needs to be covered.

We must find a material that filters sound favorably.

Cutting these by hand is very time consuming but also a possible solution for cheap DIY build.

Exacto knife.



# Chapter 2 – Low-cost HOA SMA

Recorded several stimuli with FOA mics to perform a listening experiment (IRB approved).

Built an ICS-40720 Octathingy to test calibration routines.

**Alex Tung** designed custom CAD parts.
Additional custom parts in TinkerCAD.

*TinkerCAD is not FOSS but free and very easy to use.*

# Chapter 2 – Low-cost HOA SMA

Next steps:

- Conduct the subjective test of the FOA microphones.
- Use calibration routines to create our own encoder.
- Switch to ICS-43434 and incorporate MCU.

# Chapter 2 – Low-cost HOA SMA

<u>Following that:</u>

* Design a PCB for the ICS-41350 (PDM v. I2S)

* Build 3OA design
    * (icosahedron -4 mics or Fliege grid)

* Try hollow/rigid designs, radial filters, different encoding methods.

# End of chapter 2

- Questions?

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

In Chapter 1 we looked at systems we can use to mix and master ambisonic music.

In Chapter 2 we looked at HOA SMAs, which are a convenient way to record in this format.

*Once we have created this music, how can we present it?*

* **Films**
* **HDLA concerts**
* **Surround sound systems (at home)**
* **“Fixed Perspective Binaural Audio”**
* **VR experiences (“Head-Tracked Binaural Audio”)**

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

*** Films**
The composer is beholden to the desires of the director and producers.

*** HDLA concerts**
Very few Universities have access to these.

*** Surround sound systems**
Expensive, only a few sound engineers can afford to buy them.

*** Present in “Fixed Perspective Binaural Audio”**
Does not *rely* convey much about the musical space. Can’t rotate SF.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- Still considering FOSS, low-cost, and accessibility - unifying theme.
  - Dedicated HMDs
    - Too expensive, proprietary and not ubiquitous yet. For now.
  - CAVEs
    - Too expensive.
  - Mobile HMDs (2 flavors)
    - Native apps via SDKs (such as Cardboard SDK
    - WebXR solutions

CAVE

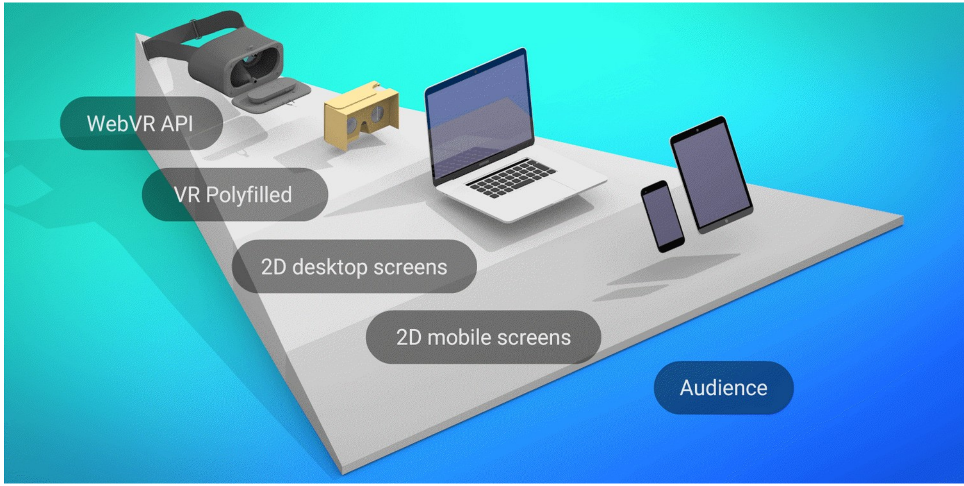
# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

The DayDream is dead

## Mobile HMDs (2 flavors)

- Native apps via SDK (such as Cardboard SDK)
  - **Pros**: stable, x-compatible (multiple compilations), can be larger in size.
  - **Cons**: not as flexible, slowly being deprecated.
- **WebXR solutions**
  - **Pros**: work on desktop, mobile, and HMDs. No compilation required.
  - **Cons**: less stable (need internet), need to be simpler (data stored in RAM).

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR



Src: W3.org

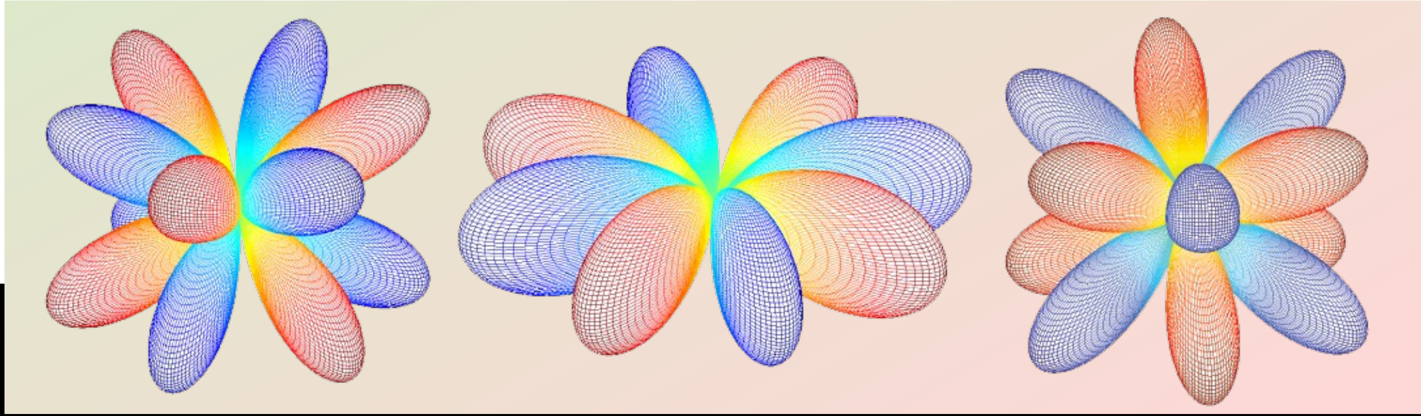
WebXR solutions

- Ultimately, more appealing as a FOSS solution.
- Ability to view content anywhere. No **need** for a smartphone.

- Head-tracked Binaural Audio gives impression of sound changing as we rotate.
- Ambisonics is a great system for this, linear transform can rotate SF.
- Binaural decoding is cheap.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- In this chapter we explore 5 projects that relate to WebAudio and ambisonics.
  - **Resonance:** #library
  - **JSAmbisonics:** #library
  - **HOAST:** #player
  - **WHAM:** #other
  - **POI:** #piece



# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

## Resonance

- Comprehensive solution (multiple platforms).
  - C++, C#, JS, etc.
- Optimizations for low-end mobile devices.
- Open source licensing but developed by Google.
- Gorzel, Marcin, et al. "Efficient encoding and decoding of binaural sound with resonance audio." Audio Engineering Society Conference: 2019 AES International Conference on Immersive and Interactive Audio. Audio Engineering Society, 2019.

Resonance Audio
by Google

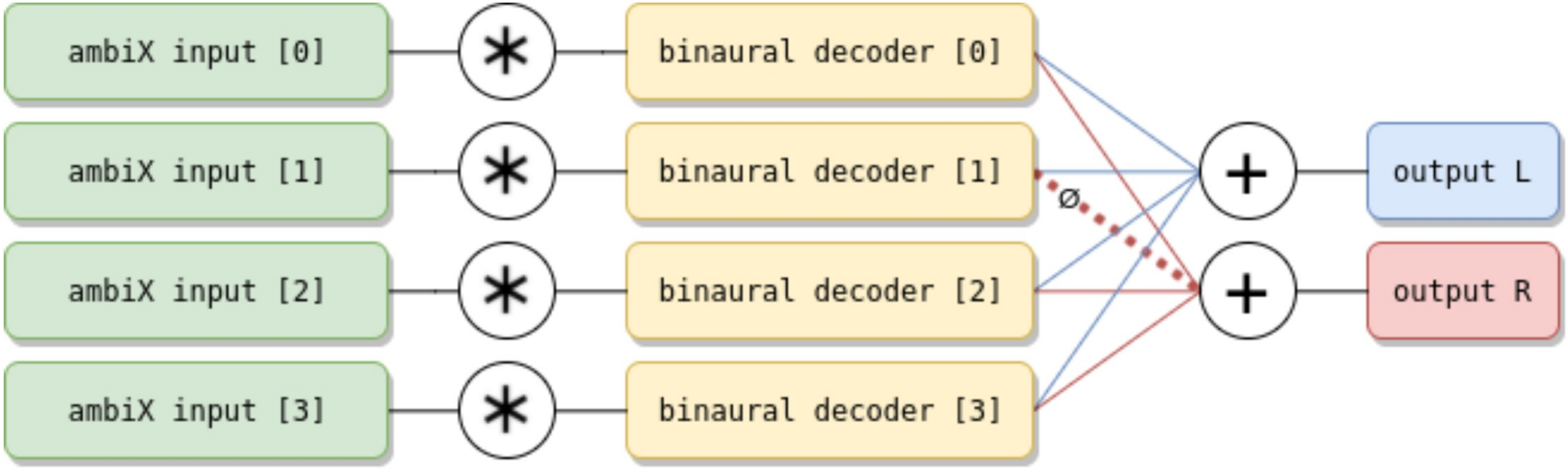
# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- **Dual-band decoding:** max-Re decoding above critical frequency.
  - HRTFs pre-filtered to reduce computations (CPU optimization).
- **Efficient encoding:** LUT to reduce calculations.
  - Reduce memory of LUT by calculating a single quadrant of sphere (exploit symmetry)
- **Control sound source spread:** energy-preserving max-Re using fractional ambisonic orders.
  - We saw this feature in IEM's Directivity Shaper as well.
  - Normalized max-Re coefficients also exported as LUT.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

## Resonance

- **Near-field sources:** not yet published.
- **Efficient Binaural Decoding:** LS expansion of SADIE HRTF dataset.
  - Assuming head-symmetry to reduce number of convolutions.
    - 26-point Lebedev grid to 16 harmonics for 3OA.



# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- Resonance:
  - Room modeling algorithm (not fully published yet) [**reverb**].
  - Future work proposed by *Gorzel et al*:
    - SOFA HRTFs (LS?)
    - efficient decoding for regular and irregular (physical loudspeaker setups)
    - efficiency improvements to existing algorithms.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

## Resonance

- Conclusion:
  - Requires in-depth expertise of web-audio to be used.
    - For many composers this is not feasible.
  - Create A-Frame/Babylon.js examples?
    - Two popular WebXR frameworks.
  - While the project is open source, it is also tied to a commercial entity…

```
  init: function() {
    this.stage.elem.width = this.stage.w;
    this.stage.elem.height = this.stage.h;
    this.ctx = this.stage.elem.getContext('2d');

    for (var i = 0; i < grid.c; i++) {
      for (var j = 0; j < grid.r; j++) {
        this.circles.push(new Circle(i * grid.dim + 20, j * grid.dim + 30));
      };
    };

    this.update();
  },
  update: function() {
    var self = this;
    window.requestAnimationFrame(function() {
      self.update();
    });

    var now = new Date().getTime();
    var dt = now - (this.time || now);
```

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

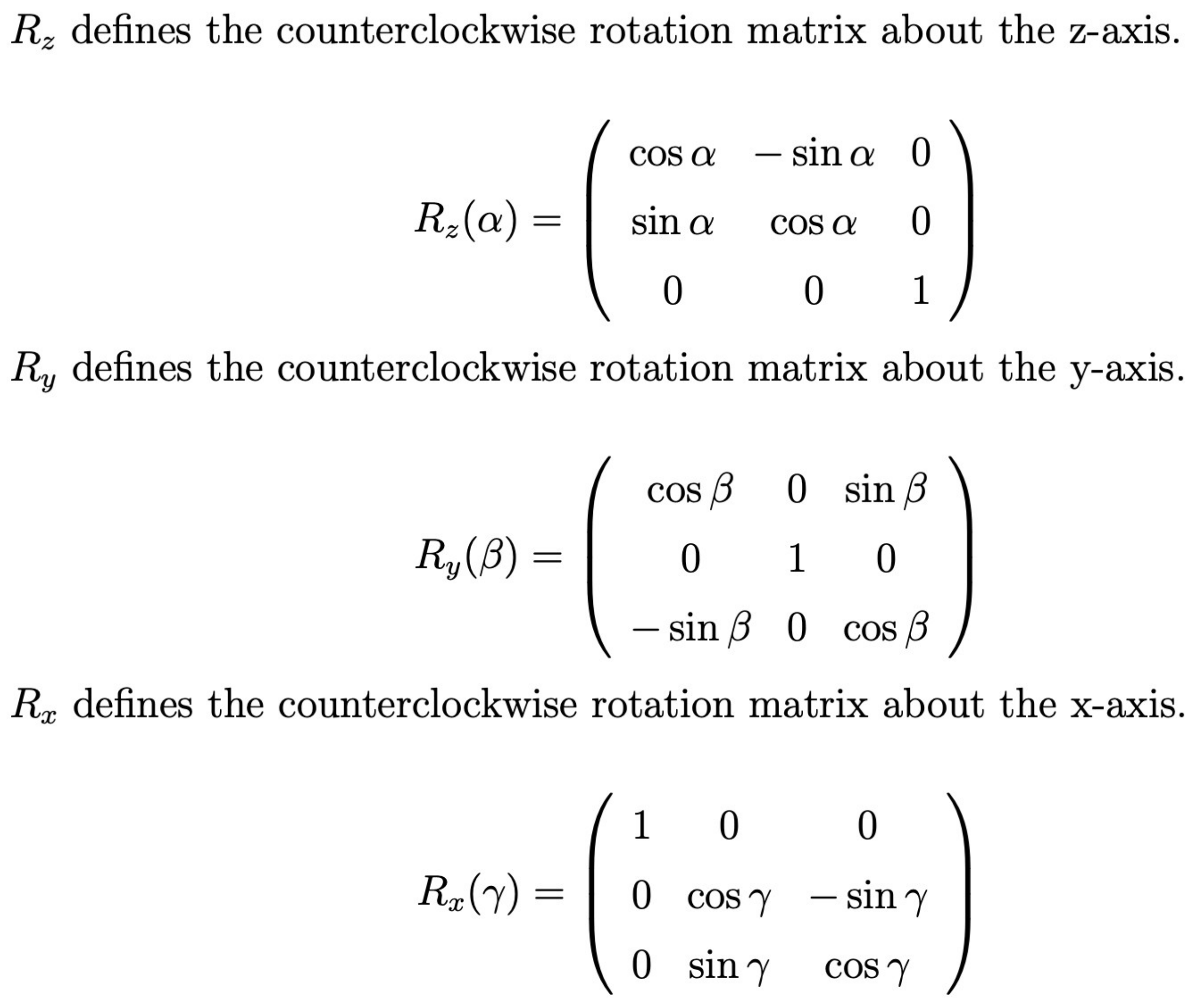
## JSAmbisonics

- From academic researcher (**Politis**).
- Not as powerful as Resonance (admittedly)
- Still comprehensive and has everything you **need.**
  - Rotation
  - Reflection
  - Beam-forming
  - Etc.
- Used in **HOAST (by IEM)** and **WHAM (by Derby)**.

Aalto University
Finland

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- JSAmbisonics
  - SF Rotation is vital
  - Rotation in FOA is very simple. If harmonics are ordered as [x, y, z] then:
    - Alpha, beta, gamma are yaw, pitch and roll.



$R_z$ defines the counterclockwise rotation matrix about the z-axis.

$$R_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$R_y$ defines the counterclockwise rotation matrix about the y-axis.

$$R_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}$$

$R_x$ defines the counterclockwise rotation matrix about the x-axis.

$$R_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

## JSAmbisonics

- We can combine these into single rotation matrix.
  - Two common conventions.
  - Here we use roll, pitch, yaw.
- Clockwise user rotation = counterclockwise SF rotation (compensate)!

$$R(\alpha,\beta,\gamma) = R_z(\alpha)R_y(\beta)R_x(\gamma) =$$

$$\begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

## JSAmbisonics

- Rotations in HOA are more complicated
- Three methods:
  - **Zotter/Kronlachner method**: easiest to understand.
  - **Recursive calculation**: from molecular chemistry, more “accurate”.
    - Used in JSAmbisonics and Resonance.
  - **Complex domain rotation**: requires FFT on ambisonic signals.
    - used in Sparta’s ambiBin

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

**Zotter/Kronlachner method**:

- HOA rotation matrices for z-axis are easy to compute.
- Just need one rotation along the y-axis to change orientation of harmonic.
  - 90 degrees.
- Re-use R_y matrix to return SH to original orientation
  - with modified yaw, pitch, roll.

$$R(\alpha, \beta, \gamma) = R_z(\alpha + 90) \cdot R_y(90) \cdot R_z(\beta + 180) \cdot R_y(90) \cdot R_z(\gamma + 90)$$

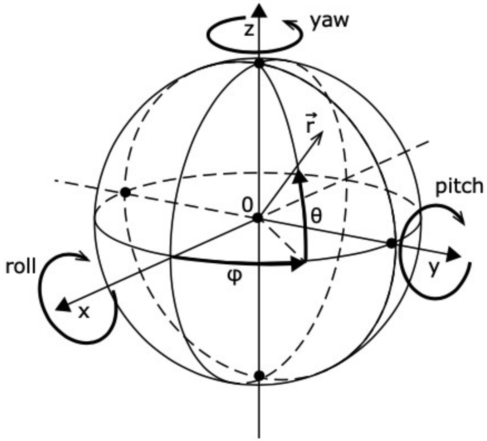
$$\boldsymbol{R}_z(\alpha) = \left(\begin{array}{c|ccc|ccccc|c}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} \\
\hline
0 & \cos\alpha & 0 & \sin\alpha & 0 & 0 & 0 & 0 & 0 & \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} \\
0 & -\sin\alpha & 0 & \cos\alpha & 0 & 0 & 0 & 0 & 0 & \\
\hline
0 & 0 & 0 & 0 & \cos 2\alpha & 0 & 0 & 0 & \sin 2\alpha & \\
0 & 0 & 0 & 0 & 0 & \cos\alpha & 0 & \sin\alpha & 0 & \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \mathbf{0} \\
0 & 0 & 0 & 0 & 0 & -\sin\alpha & 0 & \cos\alpha & 0 & \\
0 & 0 & 0 & 0 & -\sin 2\alpha & 0 & 0 & 0 & \cos 2\alpha & \\
\hline
\mathbf{0} & & \mathbf{0} & & & & \mathbf{0} & & & \cos 3\alpha \\
 & & & & & & & & & \ddots
\end{array}\right)$$

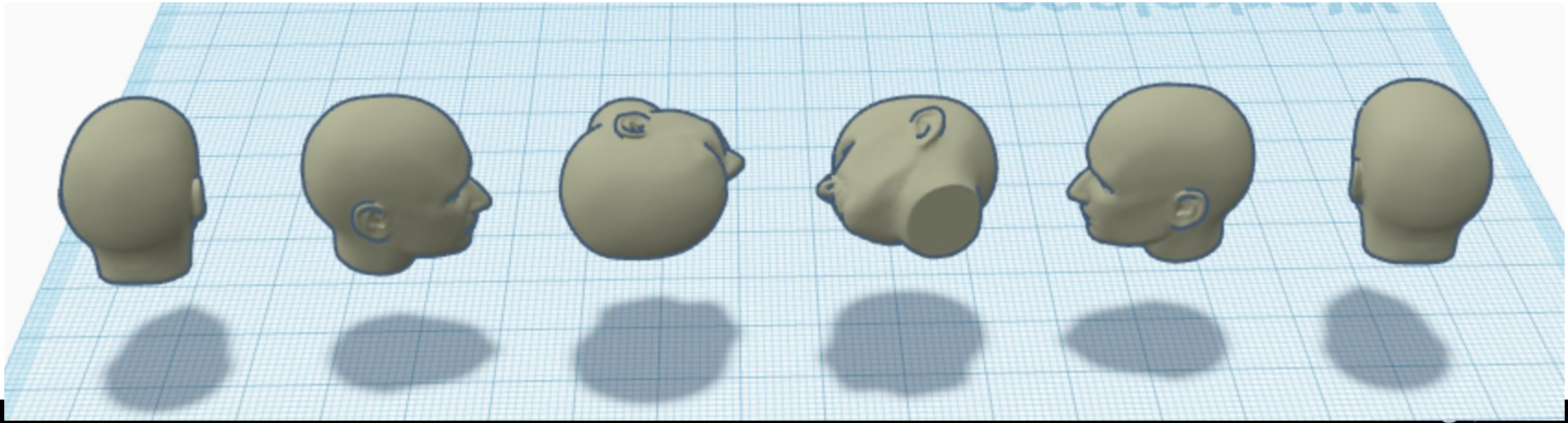
$$R(\alpha, \beta, \gamma) = R_z(\alpha + 90) \cdot R_y(90) \cdot R_z(\beta + 180) \cdot R_y(90) \cdot R_z(\gamma + 90)$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

If yaw, pitch and roll are all zero (from left to right):

- Rotate about z axis 90 degrees.
- Rotate about y axis 90 degrees.
- Rotate about z axis 180 degrees.
- Rotate about y axis 90 degrees.
- Rotate about z axis 90 degrees.
  - Which returns the model to it’s original position.





$$R(\alpha, \beta, \gamma) = R_z(\alpha + 90) \cdot R_y(90) \cdot R_z(\beta + 180) \cdot R_y(90) \cdot R_z(\gamma + 90)$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- **Zotter/Kronlachner method**:
  - R_y matrices provided by Zotter on IEM website.
- **Recursive calculation**:
  - Use solutions by Politis or Gorzel. Code based on paper by Blanco et al.
    - Miguel A Blanco, Manuel Fĺorez, and Margarita Bermejo. Evaluation of the rotation matrices in the basis of real spherical harmonics. **Journal of Molecular Structure**: THEOCHEM, 419(1-3):19–27, 1997.
- **Complex domain rotation**:
  - Use the SAF by Aalto. Has both real and complex rotations in HOA
    - (C++)

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- JSAmbisonics
  - Beam-forming
    - Create beam-patterns from SH signals.
    - Create custom decoders, **isolating sound sources**, or modifying a particular region of SF.
      - **b** is a vector with one sample of all harmonics (at time t),
      - **w** is a vector of beam-forming weights (calculated based on direction gamma_0).

$$x_{\mathrm{vm}}(t, \gamma_0) = \mathbf{w}^{\mathrm{T}}(\gamma_0)\mathbf{b}(t)$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

$$x_{\mathrm{vm}}(t, \gamma_0) = \mathbf{w}^{\mathrm{T}}(\gamma_0)\mathbf{b}(t)$$

Beam-forming

- The **w** vector has a “pattern-dependent” part and a “rotation-dependent” part.
  - There are (N+1) “pattern-dependent” coefficients, corresponding to the ambisonic order n.
  - There are (N+1)^2 “rotation-dependent” coefficients, calculated using the real-valued SH equation.

$$[\mathbf{w}(\gamma_0)]_q = w_{nm} = c_n Y_{nm}(\gamma_0)$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

Three equations are provided in the associated publication for: cardioid, hyper-cardioid, and max-Re beam-formers.

$$\text{cardioid} \; : c_n = \frac{N!N!}{(N+n+1)!(N-n)!}$$

$$\text{hypercardioid} : c_n = \frac{1}{(N+1)^2}$$

$$\text{max} - \text{rE} : c_n = \frac{P_n(\cos \kappa_N)}{\sum_{n=0}^{N}(2n+1)P_n(\cos \kappa_N)}$$

K_N = cos(2.407/(N + 1.51))
P_n = ass. Legendre functions

$$[\mathbf{w}(\gamma_0)]_q = w_{nm} = c_n Y_{nm}(\gamma_0)$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

Decoding – the JS library does not actually implement these (only *binDecoder()* VLS method).

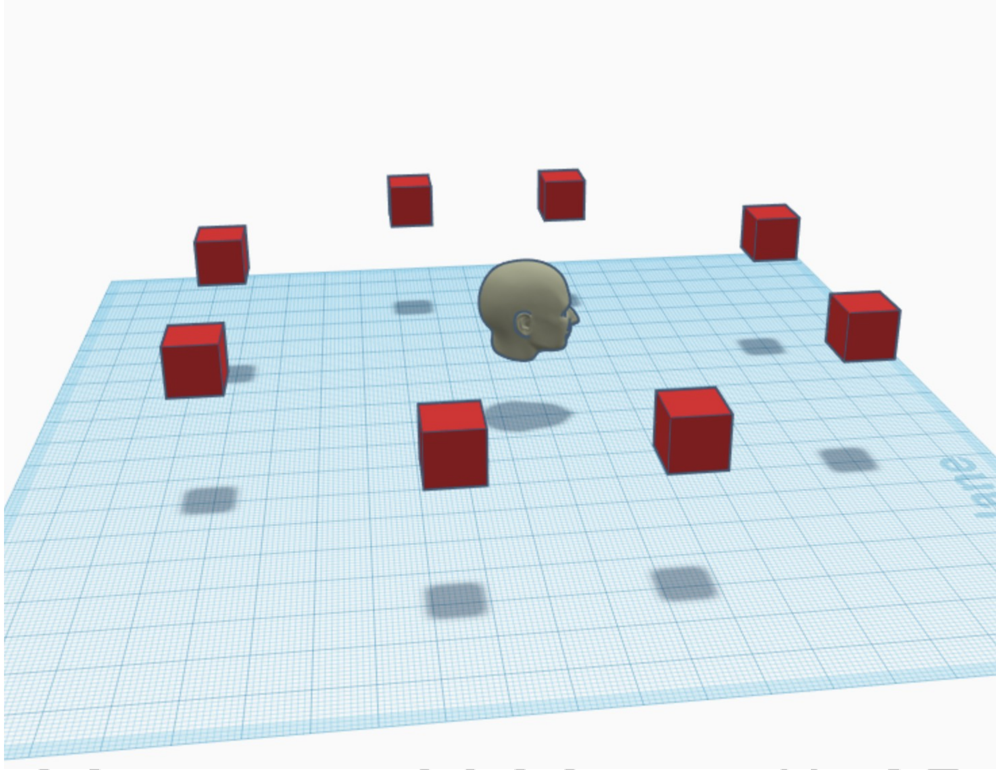
- **Sampling**: only works well for regular layouts. Simplest decoder.
- **Mode-Matching**: based on a regularized LS solution.
- **All-RAD**: selected decoder method in JSAmbisonics. **G** is decoding weights from VBAP algorithm. (td = t-design).
  - Decoder matrix pre-computed (MATLAB).

$$
\begin{aligned}
\text{Sampling} &: \mathbf{D}_{\text{ls}} = \frac{1}{L}\mathbf{Y}_L^{\text{T}} \\
\text{Mode} - \text{matching} &: \mathbf{D}_{\text{ls}} = \left(\mathbf{Y}_L^{\text{T}}\mathbf{Y}_L + \beta^2\mathbf{I}\right)^{-1}\mathbf{Y}_L^{\text{T}} \\
\text{AllRAD} &: \mathbf{D}_{\text{ls}} = \frac{1}{N_{\text{td}}}\mathbf{G}_{\text{td}}\mathbf{Y}_{\text{td}}^{\text{T}}
\end{aligned}
$$

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

Decoding – the JSAmbisonics library utilities folder has an AllRAD MATLAB example.

- In VLS we use "real" decoding matrix
- Then we convolve the HRTFs with virtual speaker signals
- Number of convolutions depends on size of virtual array
  - Whereas in SH expansion, it depends on order

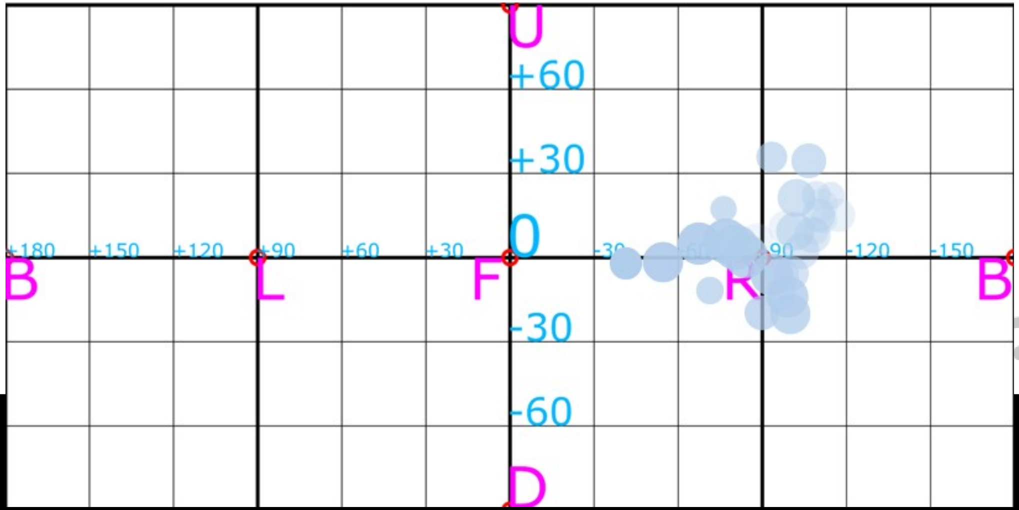


Virtual LoudSpeaker = VLS

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

Visualization

- JSAmbisonics performs a SF visualization using parametric analysis - similar to Aalto Tool-kit.
- Calculates intensity and plots it real-time.
- Useful for artists who want to upload ambisonic music, but don't have visual accompaniment.
- Sound localization is bi-modal, seeing sources helps localize them.



# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- SOFA Decoding (Spatially Oriented Format for Acoustics)
  - Using VLS method.
    - Personalized HRTFs have been shown to aid in localization (especially for elevation angles).
- Personalized HRTFs are difficult to acquire.
  - Ongoing area of research.
  - See Corentin Guezenoc and Renaud Seguier. Hrtf individualization: A survey. arXiv preprint arXiv:2003.06183, 2020.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- VR experiences (“Head-Tracked Binaural Audio”)
    - HOAST [HOA Streaming]
        - JSAmbisonics not easy to use for non-programmers
        - HOAST employs JSAmbisonics
        - Allows up to 4OA. Developed by IEM
            - YT – FOA
            - FB – 2OA.
        - OPUS CODEC (.ogg) + VP9 (video codec) + WebM container.
            - Open source.

# HOAST [HOA Streaming]

- MPEG-DASH = **D**ynamic **A**daptive **S**treaming over **H**TTP (dash.js)
  - Adapts to bandwidth!
- MagLS binaural decoder (received code from Deppisch!)
- Acoustic zoom via Warping + Directional Loudness.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- VR experiences ("Head-Tracked Binaural Audio")
  - HOAST [HOA Streaming]
    - No back-end - only creators can upload videos.
    - Requires video!
    - Works with HMDs, desktop, mobile.
    - No ads (good)

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- VR experiences ("Head-Tracked Binaural Audio")
  - HOAST [HOA Streaming]
    - Publisher retains full rights of content (good)
    - Great sound quality (unless moving fast).
    - Visualizations not real-time.
    - Requires very specific encoding
      - FFMPEG hard for non-programmers to use.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

## WHAM [Webcam Head-tracked AMbisonics]

- Uses JSAmbisonics.
- Head-tracking using Beyond Reality Face (SDK)
- Alternative to head-trackers
  - OSC, serial, MIDI, Bluetooth
- Limited angular resolution
- Requires good lighting.

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

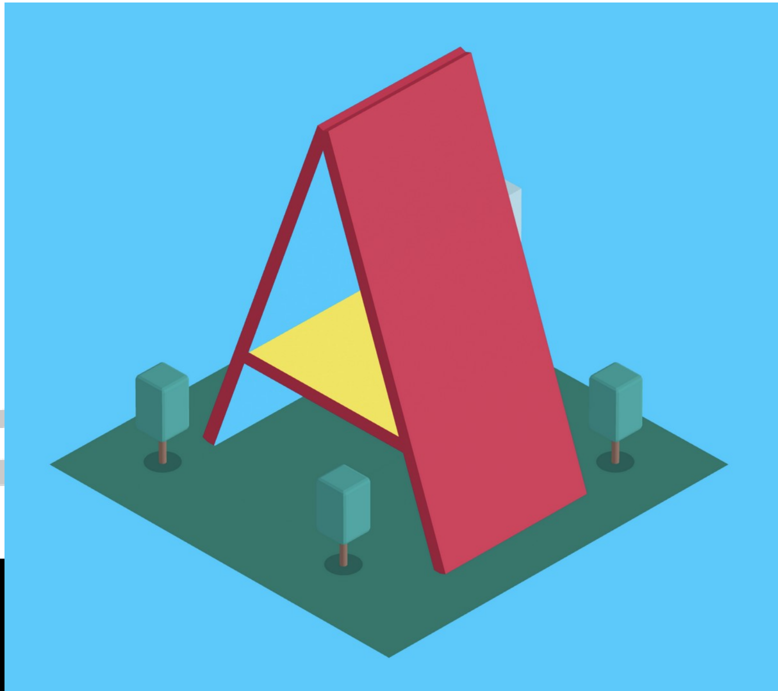
## WHAM [**W**ebcam **H**ead-tracked **AM**bisonics]

- Pros:
  - No need to wear HMD.
  - No need to wear tracker.
  - Share ambisonic music easily!
- Cons:
  - Plus/minus 40 degrees of angular resolution.
  - Latency high (128ms)
  - Currently not set-up as a music player.

UNIVERSITY of DERBY

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

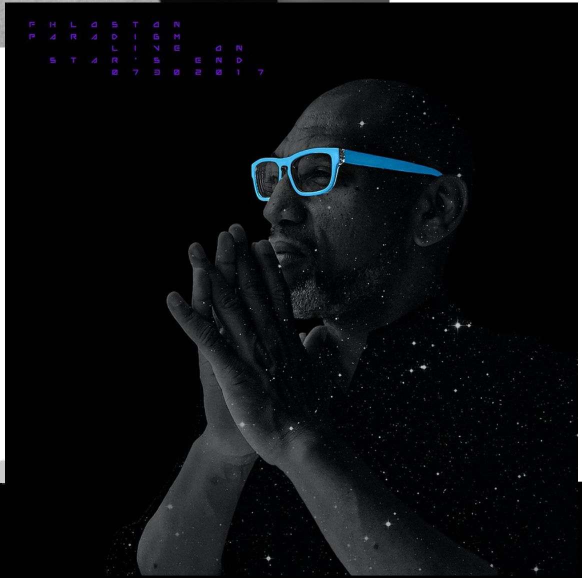
- VR experiences (“Head-Tracked Binaural Audio”)
  - POI
    - Collaboration with Tim Gmeiner and Eito Murakami (two ICAM students).
    - Explored various frameworks for WebXR + ambisonics
    - Decided to use A-Frame + Omnitone



# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- VR experiences (“Head-Tracked Binaural Audio”)
  - POI
    - VR project with FOSS + low-cost hardware.
    - Google Cardboard + binaural audio.
    - Reaper and IEM suite to create ambisonic mix.
    - Features King Britt and Stephanie Richards (UCSD Faculty).

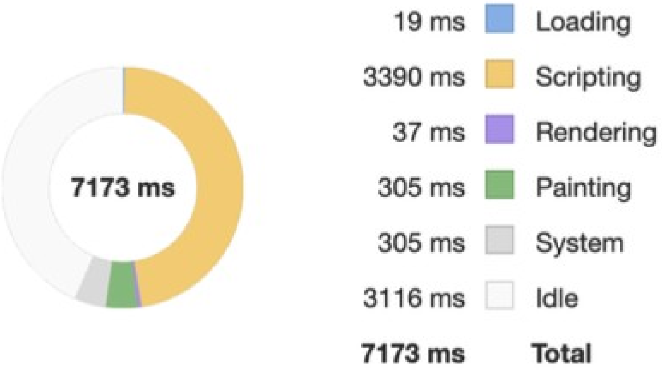
Stephanie Flugelhorn



King Synths

# Chapter 3 – Equitable Distribution of Spatial Music Using WebXR

- VR experiences ("Head-Tracked Binaural Audio")
  - POI
    - Neocities used to host project
    - Audacity to compress audio (OGG)
    - Creative commons assets: 360 photo, models, textures.
    - Created 3 versions HOA, FOA and binaural.
    - Analyzed load time & CPU/RAM usage.
    - Submitted to Audio Mostly.
      - Music track.



# Questions?