Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Sound Source Localization using the Azure Kinect's Microphone Array on a Robot

Master Thesis Colloquium

Roland Fredenhagen

2023-06-20

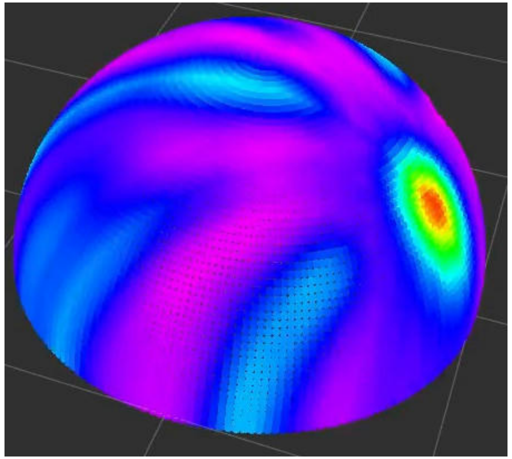
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Agenda

Introduction 3
Localization (SRP-PHAT) 6
Tracking 11
Separation 14
Implementation 17

Sound Source Localization

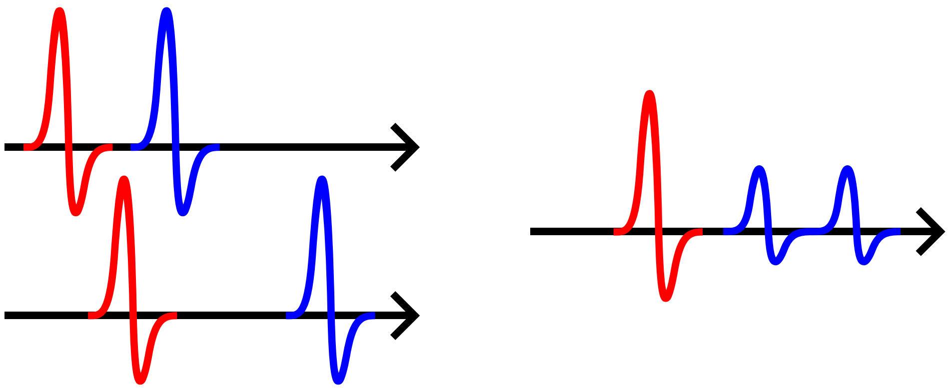
- approximating the 3D location of a sound source
- input: multichannel audio produced by microphone array
- output: direction of arrival (DoA)



Strength of cross correlation as point cloud

Signal Separation

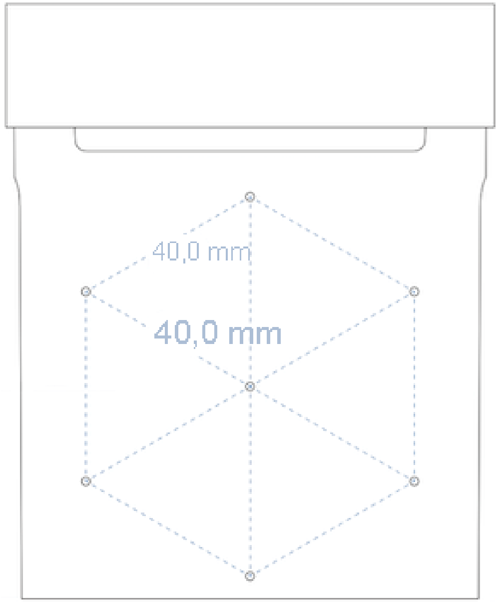
- isolating a source's signal
- input: multichannel audio and source location (DoA)
- output: single channel audio



Signal Separation using Delay-and-Sum Beamformer (DaS) adapted from [1]

Setup

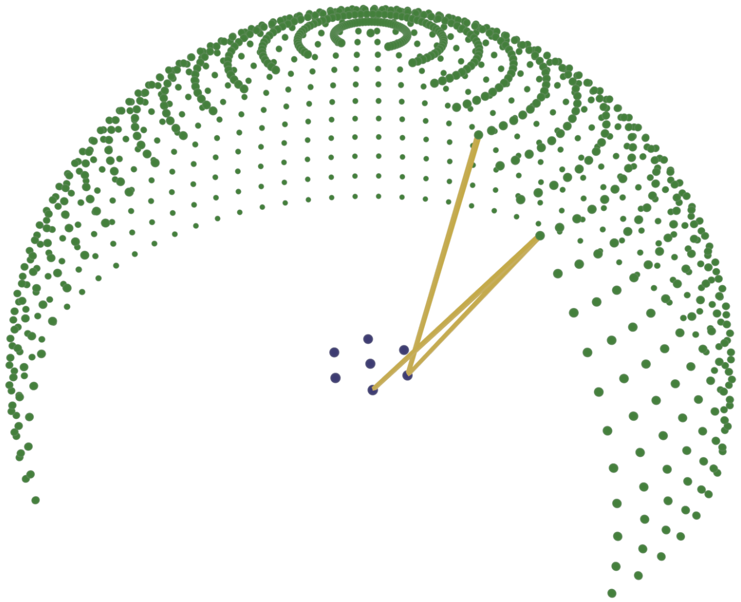
- Azure Kinect microphone array on robot head
- 7 upwards facing microphones in circle



Microphones in Azure Kinect [2]

Steered-Response Power with Phase Transform [3]

- iterate over solution space (sampled DoAs)
- find corresponding time difference of arrival (TDoA) between microphones
- measure SRP at TDoA



Grid of possible DoA on 1m “unit-sphere”

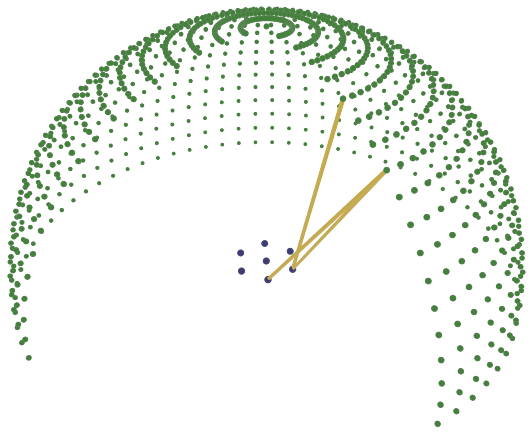
Steered-Response Power with Phase Transform [3]

- find DoA s (position on sphere) with maximum SRP P

$$\hat{s} = \arg\max_{s \in G} P(s)$$

$$P(s) = \sum_{\{m_1, m_2\} \in [M]^2} R_{m_1 m_2}\left(\tau_{m_1}(s) - \tau_{m_2}(s)\right)$$

$$\tau_m(s) = \frac{|s - m|}{c}, c = \text{speed of sound}$$



Grid of possible DoA on 1m "unit-sphere"

Generalized Cross-Correlation with Phase Transform [4]

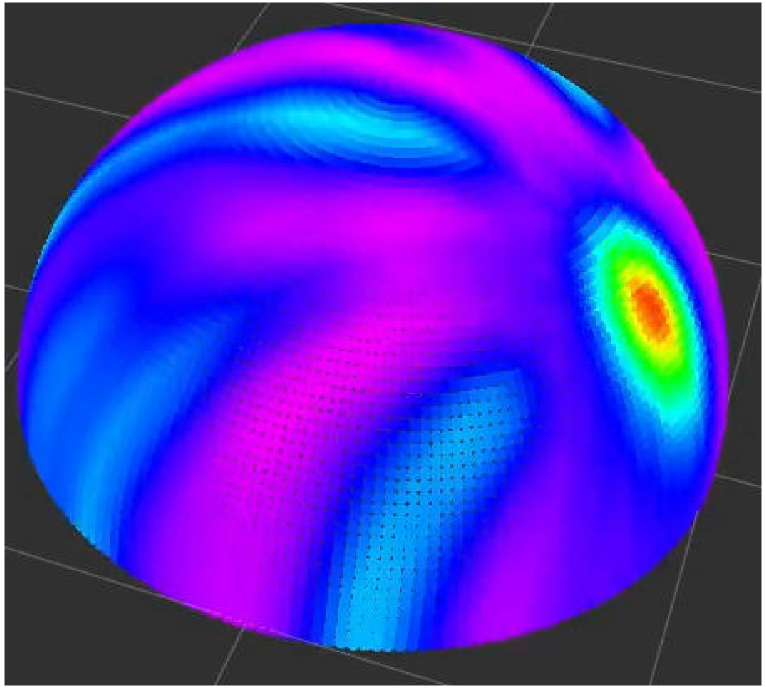
- cross correlation at delay τ using *PHAT* weighting

$$R_{m_1 m_2}(\tau) = \int_{-\infty}^{+\infty} \Psi_{m_1 m_2}(f) X_{m_1}(f) X^*_{m_2}(f) e^{j2\pi f \tau} \, \mathrm{d}f$$

$$\Psi_{m_1 m_2}(f) = \frac{1}{|X_{m_1}(f) X^*_{m_2}(f)|}$$

Sound Source Localisation

- skipping the maximization returns an “intensity” for all DoAs
- multiple local maxima
  $\Longrightarrow$ possibly multiple sound sources
- possibility to filter by threshold

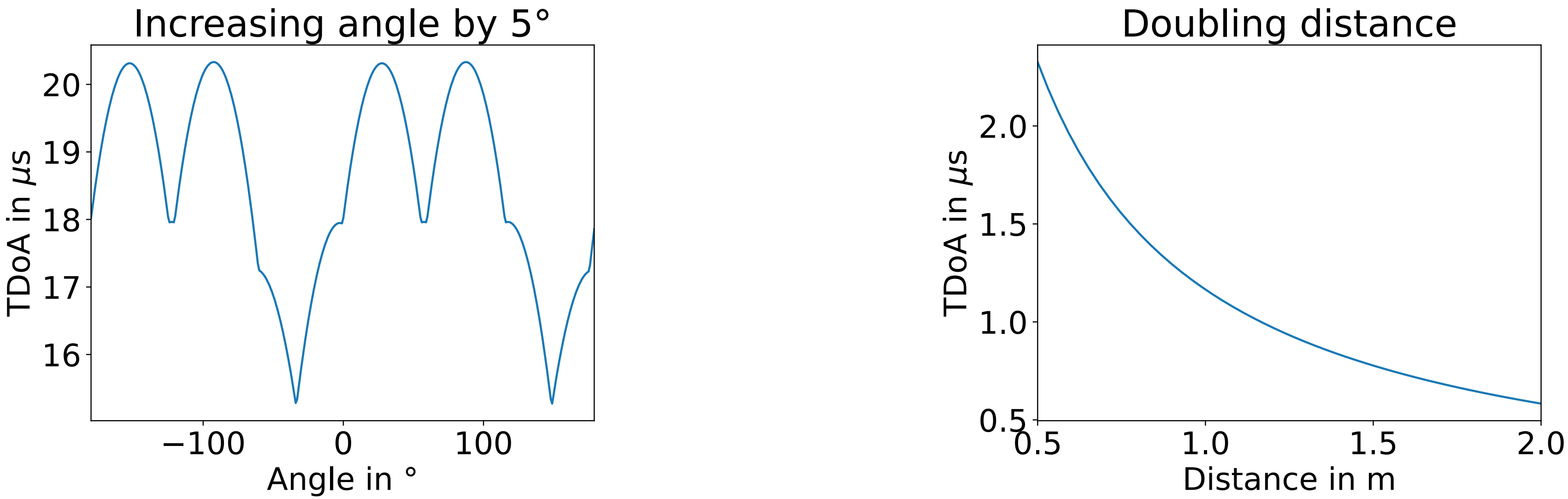


Grid of possible DoA on 1m “unit-sphere”

Distance

- changes in distance result in very small TDoA
- sound approximately plane wave

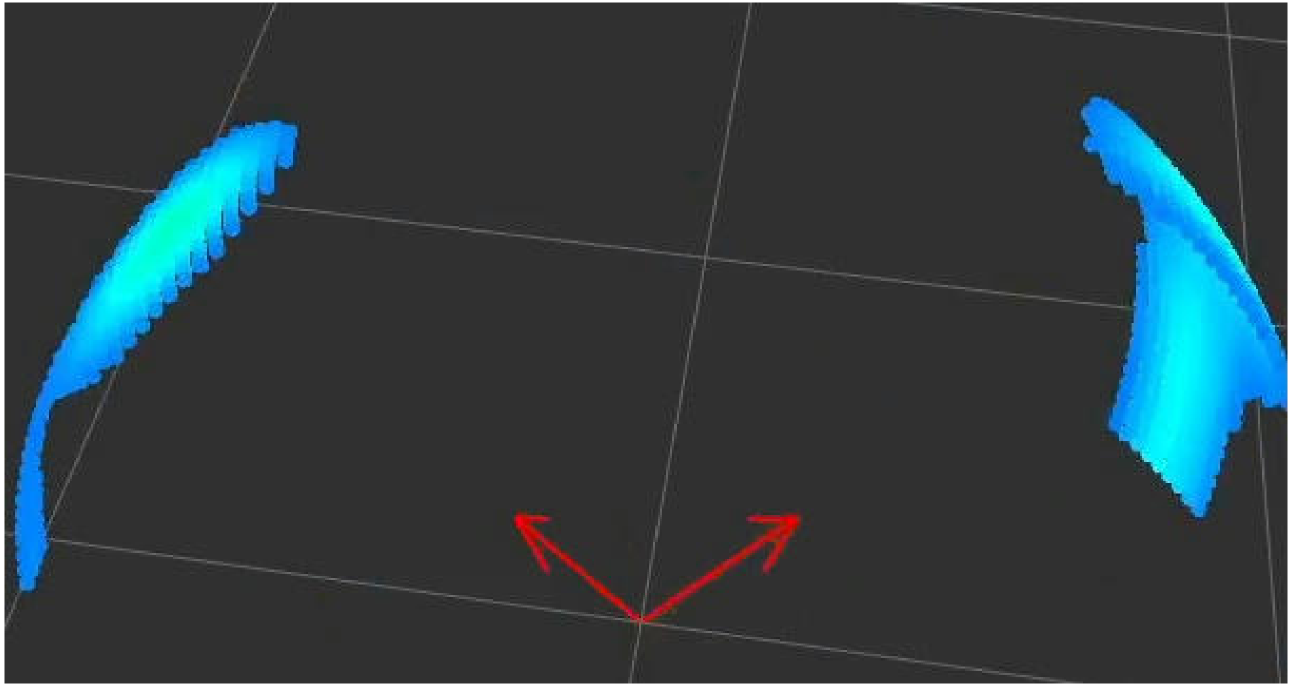
Maximum change in TDoA



Tracking

- real sound source not permanently outputting
- consistent IDs improve usability of e.g. Source Separation
- when source s is detected:
  - find already tracked sources close to s
  - if none, track s with new id
  - else, track s with neighbor's id

Tracking



Poses produced by tracking

Possible Future Tracking Improvement

could be improved by taking into account:

- robot movement and rotation
- velocity/acceleration of sources
- spectral properties of sound sources

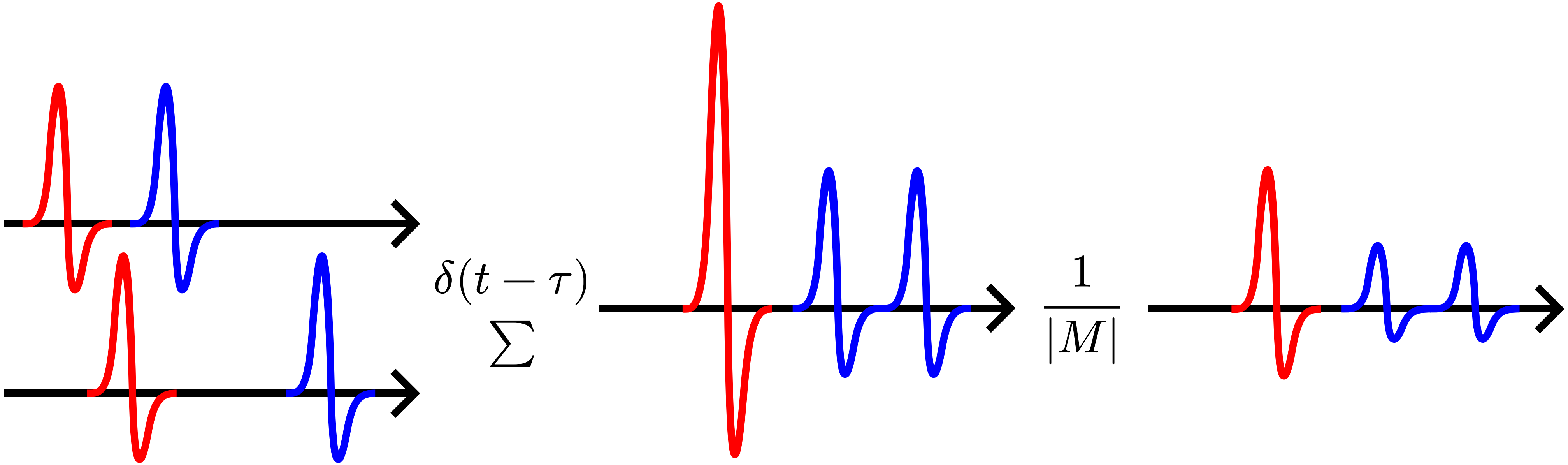
Delay-and-Sum Beamformer

- amplify one source s by summing delayed inputs $x_m(t-\tau)$
- normalise by number of tracks $|M|$

$$y_s(t) = \frac{1}{|M|}\sum_{m\in M} x_m\Big(t - \tau_{m_n}(s) + \tau_m(s)\Big), m_n = \text{furthest mic from } s$$

- constructive interference for the wanted source

Delay-and-Sum Beamformer



Delay-and-Sum Beamformer with two inputs adapted from [1]

Minimum Variance Distortionless Response Beamformer *(Not yet Implemented)*

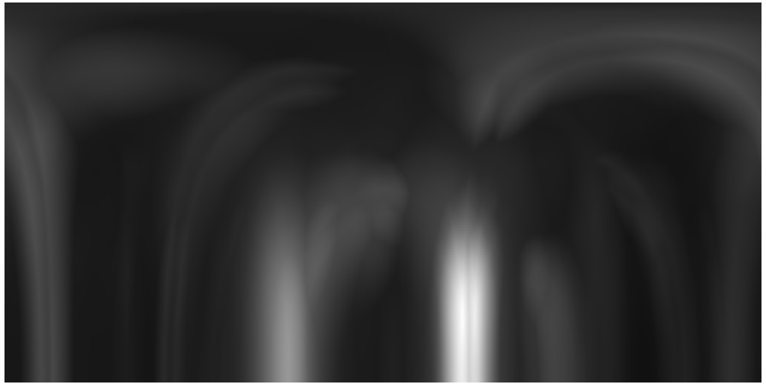
- takes into account both DoA and input signal to dynamically reduce interference
- adds weighing w:
  - **minimize** overall variance of output $y \Rightarrow$ interference minimized
  - **subject to** signal in direction of source s stays undistorted

ssloc [6]

- FOSS Rust crate providing both a library and CLI application
- highly configurable sound source localisation implementation (exposed by `ssloc_ros` via `dynamic_reconfigure`) based on *SRP-PHAT*
- Delay-and-Sum Beamformer
- capable of processing real time and recorded inputs

ssloc [6]

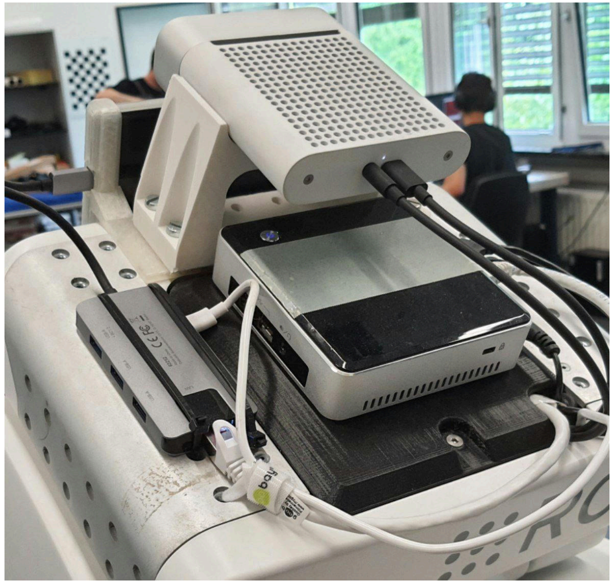
- Audio Source:
  - ALSA (Linux audio hardware)
  - WAVE or generic PCM-Audio data
- Output:
  - Spectrum as CSV or Image
  - Probable audio sources in azimuth and elevation



Spectrum as image

ssloc – Setup

- runs on Intel NUC — low performance hardware
- localization “frame” $\approx 0.1\mathrm{s}$
- grid resolution $\approx 4°$
- Azure Kinect records at $48000\mathrm{Hz}$
  - FFT-Window of $0.64\mathrm{s} \Rightarrow 4096$ Samples
  - Frequencies $11\mathrm{Hz}$ – $24\mathrm{kHz}$



ssloc_ros [7]

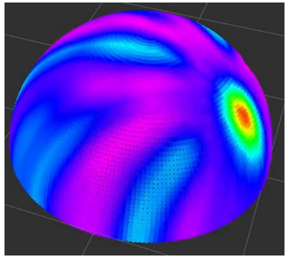
- ROS package wrapping `ssloc`
- requires `cargo` to be installed as built dependency
- `cargo` controlled through `cmake` to work with `catkin`
- ROS node either recording on robot or separately through messages

ssloc_ros – Node

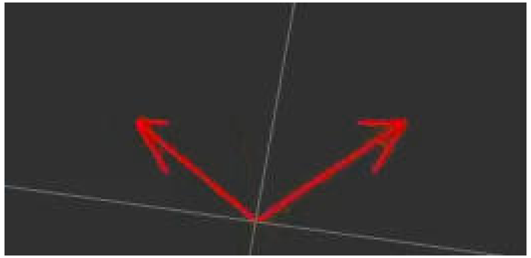
- audio
  - record directly
  - receive audio over messages (`audio_capture`)
  - separate thread putting “frames” into queue
- ssloc:
  - multithreaded – worker threads take “frames” from queue
  - produces messages with computation results to ROS network

ssloc_ros – Messages

- ROS timestamp of recording
- data from localization, tracking and source separation
- source audio and separated sources (`audio_common` [8])
- visualization (rviz): point-cloud and pose array
- spectrum as image



Point Cloud



Pose Array

ssloc_ros_msgs [9]

- ROS messages for `ssloc_ros`
- published separately to not propagate `ssloc_ros`'s built dependencies
  - `Ssl.msg`, `SslArray.msg`
  - `Sst.msg`, `SstArray.msg`
  - `SssMapping.msg`

ssloc_ros_msgs [9]

- sound separation data as multi channel audio messages:
- and channel to id mapping with custom message

```
Header header
int64[] sources
```

- `sources[channel_index] = source_id`

ssloc_ros_msgs [9]

- localization and tracking data at `~ssl` and `~sst` respectively

```
float64 x
float64 y
float64 z
float64 azimuth
float64 elevation
float64 P
```

in `Sst.msg` additionally:

```
int64 id
```

`SslArray.msg`:

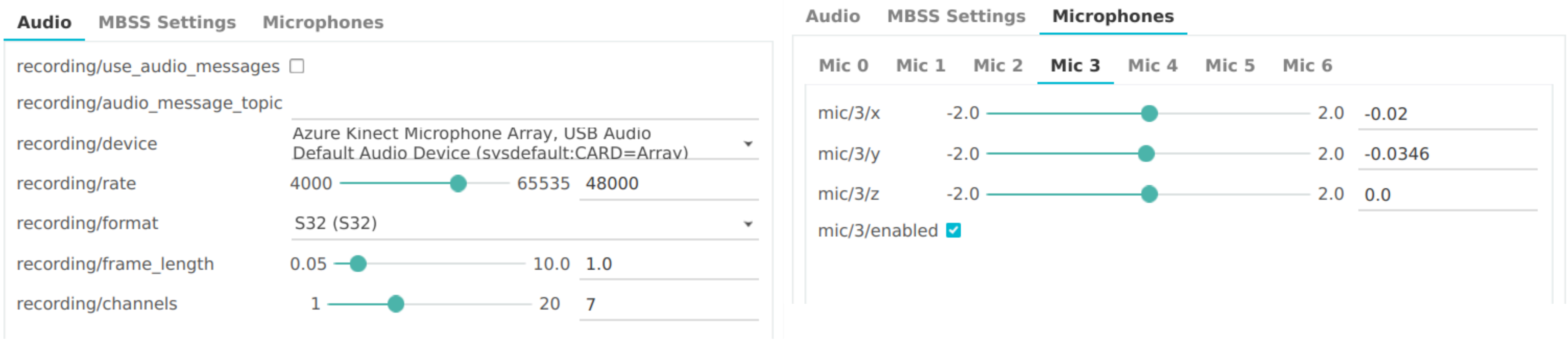
```
Header header
ssloc_ros_msgs/Ssl[] sources
```

`SstArray.msg`:

```
Header header
ssloc_ros_msgs/Sst[] sources
```

dynamic_reconfigure [10]

- libraries and applications
- runtime configuration of ROS nodes



rqt_reconfigure of *ssloc_ros* node

rosrust_dynamic_reconfigure [11]

- `dynamic_reconfigure` library only for Python and C++
- reverse engineered use of messages in `dynamic_reconfigure`
- Rust crate allowing manual use — no support for `.cfg`

Typst

- modern LaTeX alternative
- supports SVG images (but not PDF includes)
- small and young ecosystem
  - `typst-slides` doesn’t support page-breaks
  - references inside of slides were fixed last week

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Typst

```
#slide(title: [Typst], outlined: false, title-level: 1)[
#let LaTeX = style(styles => {
  let a = measure([#super[A]], styles); let e = measure([E], styles)
  [L]
  h(-.57*a.width); box(move(dy: -.6*a.height, text(size:.6em)[A]))
  h(-.2*a.width); [T]
  h(-.17em); box(move(dy: .3*e.height)[E])
  h(-.12em); [X]
})
- modern #LaTeX alternative
- supports SVG
- small and young ecosystem
  - `typst-slides` doesn't support page-breaks
  - references inside of slides were fixed last week
]
```

Typst

```
#new-section("Separation")
#slide(title: [Delay-and-Sum Beamformer], {
invis[@grythe2015beamforming]
line-by-line[
  - amplify one source by summing delayed inputs
  $ y_s_k (t) = sum_(m in M) x_m (t - tau_m (s_k) + tau_m_0 (s_k)),
    m_0 = "closest mic to" s_k $
  #[
    - constructive interference for the wanted source
    - destructive interference for the unwanted sources
  ]
  - only dependent on DoA of wanted source
]})
```

References

[1] gfai tech, “Delay-and-sum-beamforming.” https://www.gfaitech.com/de/wissen/faq/delay-und-sum-beamforming-im-zeitbereich (accessed: Jun. 10, 2023).

[2] Microsoft, “Azure kinect DK hardware specifications.” https://learn.microsoft.com/en-us/azure/kinect-dk/hardware-specification (accessed: Jun. 10, 2023).

[3] J. H. DiBiase, H. F. Silverman, and M. S. Brandstein, “Robust localization in reverberant rooms,” *Microphone Arrays: Signal Process. Techn. Appl.*, pp. 157–180, 2001.

[4] C. Knapp, and G. Carter, “The generalized correlation method for estimation of time delay,” *IEEE Trans. Acoustics, Speech, Signal Process.*, vol. 24, no. 4, pp. 320–327, 1976.

[5] J. Grythe, and A. Norsonic, “Beamforming algorithms-beamformers,” *Tech. Note, Norsonic aS, Norway*, 2015.

[6] Roland Fredenhagen, “Ssloc.” https://docs.rs/ssloc/

[7] Roland Fredenhagen, “Ssloc_ros.” https://github.com/ModProg/ssloc_ros

[8] Blaise Gassend, “Audio_common.” https://wiki.ros.org/audio_common

[9] Roland Fredenhagen, “Ssloc_ros_msgs.” https://github.com/ModProg/ssloc_ros_msgs

[10] Blaise Gassend, and Michael Carroll, “Dynamic_reconfigure.” https://wiki.ros.org/dynamic_reconfigure

[11] Roland Fredenhagen, “Rosrust_dynamic_reconfigure.” https://docs.rs/rosrust_dynamic_reconfigure/

[12] C. Blandin, A. Ozerov, and E. Vincent, “Multi-source tdoa estimation in reverberant audio using angular spectra and clustering,” *Signal Process.*, vol. 92, no. 8, pp. 1950–1960, 2012.