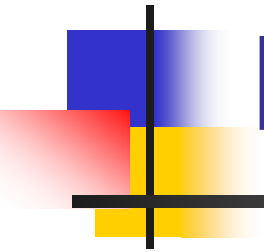
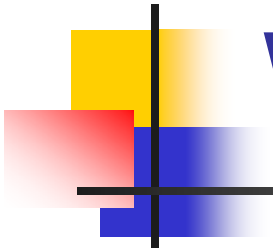


Improving Meetings with Microphone Array Algorithms

Ivan Tashev
Microsoft Research

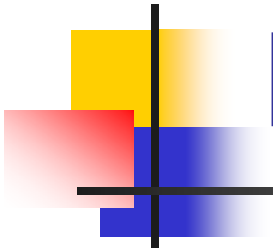


Why microphone arrays?

- They ensure better sound quality: less noises and reverberation
- Provide speaker position using sound source localization algorithms
- These technologies are used in the upper levels of meeting recording and broadcasting systems:
  - Speaker position awareness for better UI
  - Assisting speaker clustering and segmentation
  - Better speech recognition for meeting annotation and transcribing
  - Provide input data for machine learning enabled applications

Better audio quality and user experience with MicArrays

- Meeting attendees look awkward wearing microphones, nobody likes to be tethered
- Capturing sound from single point is difficult
  - A single microphone captures ambient noises and reverberation
  - Due to interference with reflected sound waves we can have some frequencies enhanced and some completely suppressed
- A microphone array is set of microphones positioned closely
  - The signals are captured synchronously and processed together
- Beamforming is ability to make the microphone array to listen to given location, suppressing the signals coming from other locations. Electronically steerable.
- Another name for this type of processing is spatial filtering

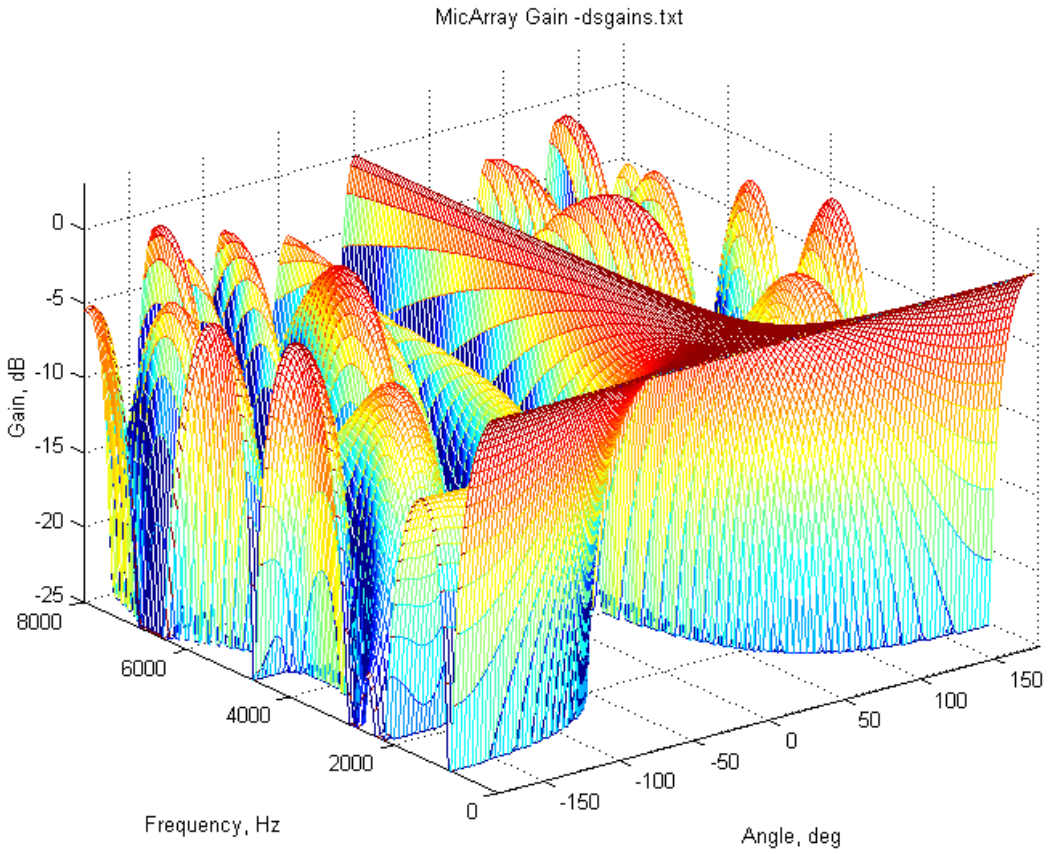


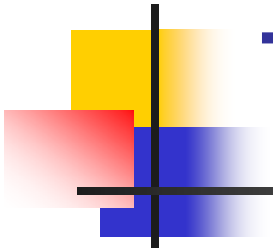
Delay and sum beamformer

- The most straightforward approach
  - As the sound from the desired direction reaches the microphones with different delay just delay properly the signals from the microphones and sum them
  - Supposedly the mismatched shifts (phases) for signals coming from other directions will reduce their amplitude
  - Fast and easy to implement
- Major problems
  - The shape of the beam is different for different frequencies
  - Almost no directivity in the lower part of the frequency band
  - Side lobes (one or more) appear in the upper part of the frequency band
- Used for comparison as a base line

Delay and sum beamformer

Delay and sum beamformer gain vs. frequency and angle





Time vs. Frequency domain

- Time domain processing
  - More “natural”, used in most of the common beamforming algorithms (GSC etc.)
  - No time spent for conversion
  - Requires long filters (200 – 2000 taps), very slow!
- Frequency domain processing
  - CPU time for conversion
  - Long filters are vector multiplications, much faster!
  - Many other types of audio signal processing are faster as well

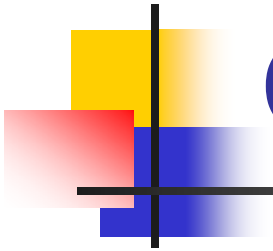
Generalized beamformer

- All time domain algorithms for beamforming can be converted to processing in frequency domain
- Canonical form of the beamformer:

$$Y(f) = \sum_{i=0}^{M-1} W(f,i) X_i(f)$$

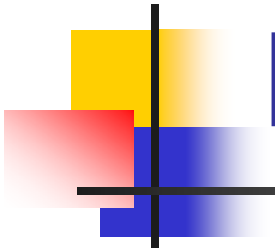
M – number of microphones
Xi(f) – spectrum of *i*-th channel
W(f,i) – weight coefficients matrix
Y(f) – output signal

- Fast processing: M multiplications and M-1 additions per frequency bin
- For each weight matrix we have corresponding shape of the beam $B(\varphi, \theta, f)$ - the array gain as function of direction



Calculation of the weights matrix

- The goal of the calculation is for given geometry and beam direction to find the optimal weights matrix
- For each frequency bin find weights to minimize the total noise in the output
- Constrains: equalized gain and zero phase shift for signals coming from the beam direction



Known approaches

- Using multidimensional optimization
  - The multidimensional surface is multimodal, i.e. have multiple extremes
  - Non-predictable number of iterations, i.e. slow
  - Multiple computations lead to losing precision
- Using the approach above with different optimization criterion:
  - Minimax, i.e. minimization of the max difference
  - Minimal beamwidth, etc.
- In all cases the starting point of the multidimensional optimization is critical

Array noise suppression

- Noise = ambient + non-correlated + correlated (jammers and reverberation)
- Ambient noise suppression

$$20\log\int_0^{\frac{f_S}{2}}\int_0^{2\pi}\int_{-\frac{\pi}{2}}^{+\frac{\pi}{2}} N(f)B(\varphi,\theta,f)\,d\theta\,d\varphi\,df$$

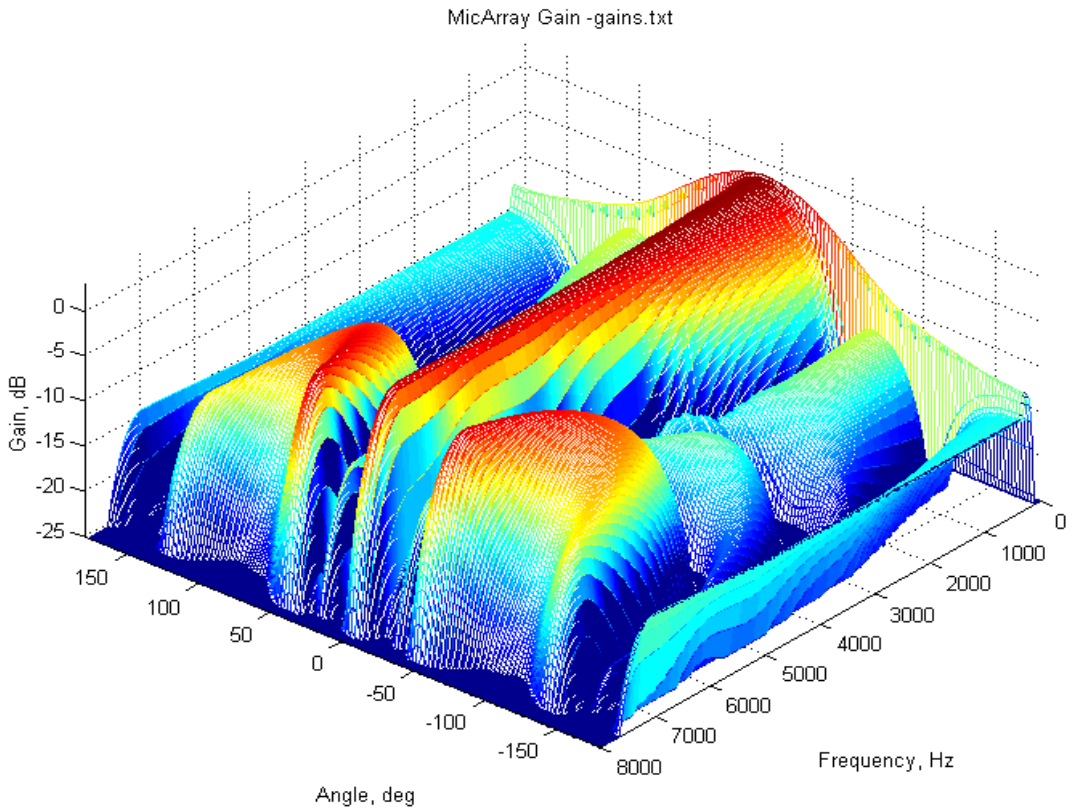
- Non-correlated noise: $20\log\left[\int_0^{\frac{f_S}{2}}\sqrt{\sum_{i=0}^{M-1}W(f,i)^2}\,df\right]$
- Correlated (from given direction): $20\log\dfrac{\int_0^{\frac{f_S}{2}} S(f)B(\varphi_S,\theta_S,f)\,df}{\int_0^{\frac{f_S}{2}} J(f)B(\varphi_J,\theta_J,f)\,df}$

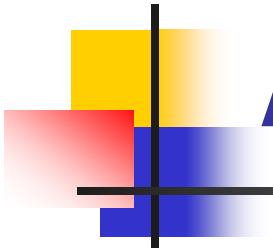
Microphone Array for meetings

- Number of microphones: 8
- Noise suppression, ambient: 12-16 dB
- Sound source suppression (up to 4000 Hz):
  - At 90^0: better than 12 dB
  - At 180^0: better than 15 dB
- Beam width at -3 dB: 40^0
- Work band: 80 – 7500 Hz.
- Principle of work: points a capturing beam to the speaker location

Microphone Array for meetings

MicArray gain vs. frequency and angle



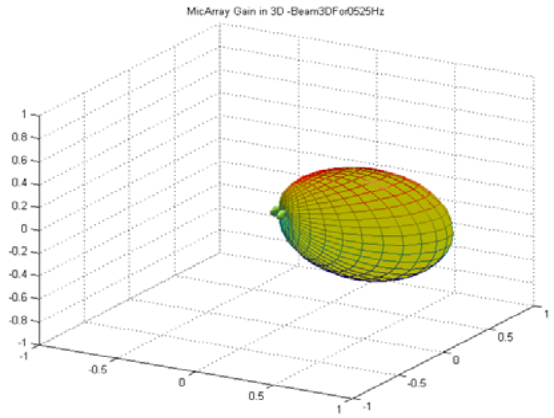


Additional goodies

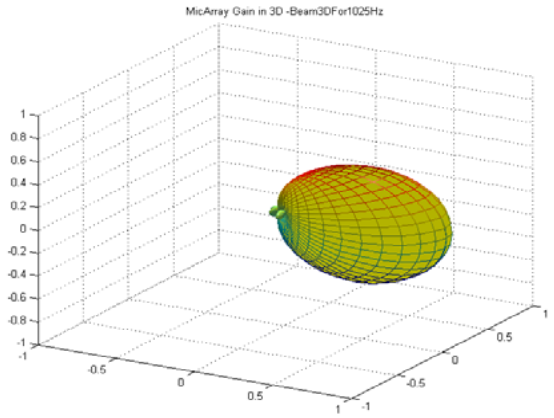
- Linear processing
  - Beamforming doesn't introduce non-linear distortions making the output signal suitable not only for recording/broadcasting, but for speech recognition as well
- Integration with Acoustic Echo cancellation
  - Requirement for real-time communication purposes
- Better noise suppression
  - The initial noise reduction from the beamformer allows using better noise suppression algorithms after it without introducing significant non-linear distortions and musical noises
- Partial de-reverberation
  - The narrow beam suppresses reflected from the walls sound waves making the sound more "dry" and better accepted from live listeners and speech recognition engines, it makes the job of potential de-reverberation processor easier

Beamshapes

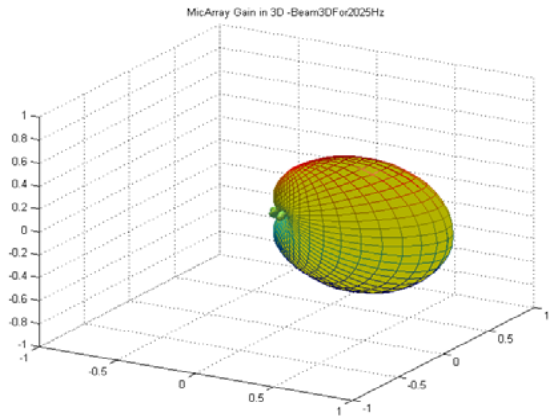
525 Hz



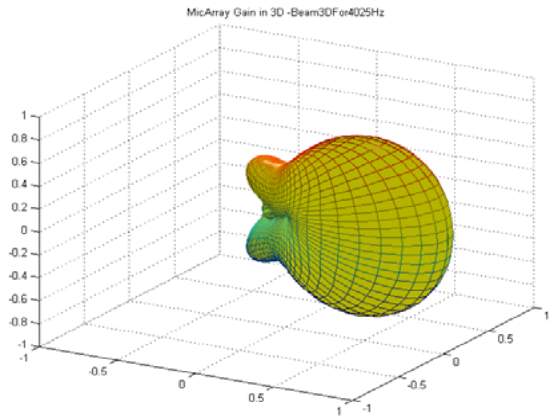
1025 Hz



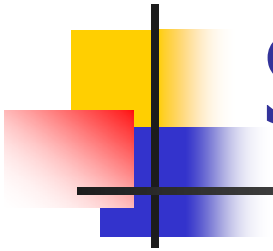
2025 Hz



4025 Hz

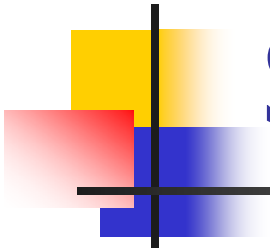


The beam shape in 3D proves frequency independent beamforming



Sound source localization

- Provides the direction to the sound source
- In most of the cases works in real-time
- Goes trough three phases:
  - Pre-processing:
  - Actual sound source localization
    - Provides a single SSL measurement (time, position, weight)
  - Post-processing of the results:
  - Final result: position, confidence level

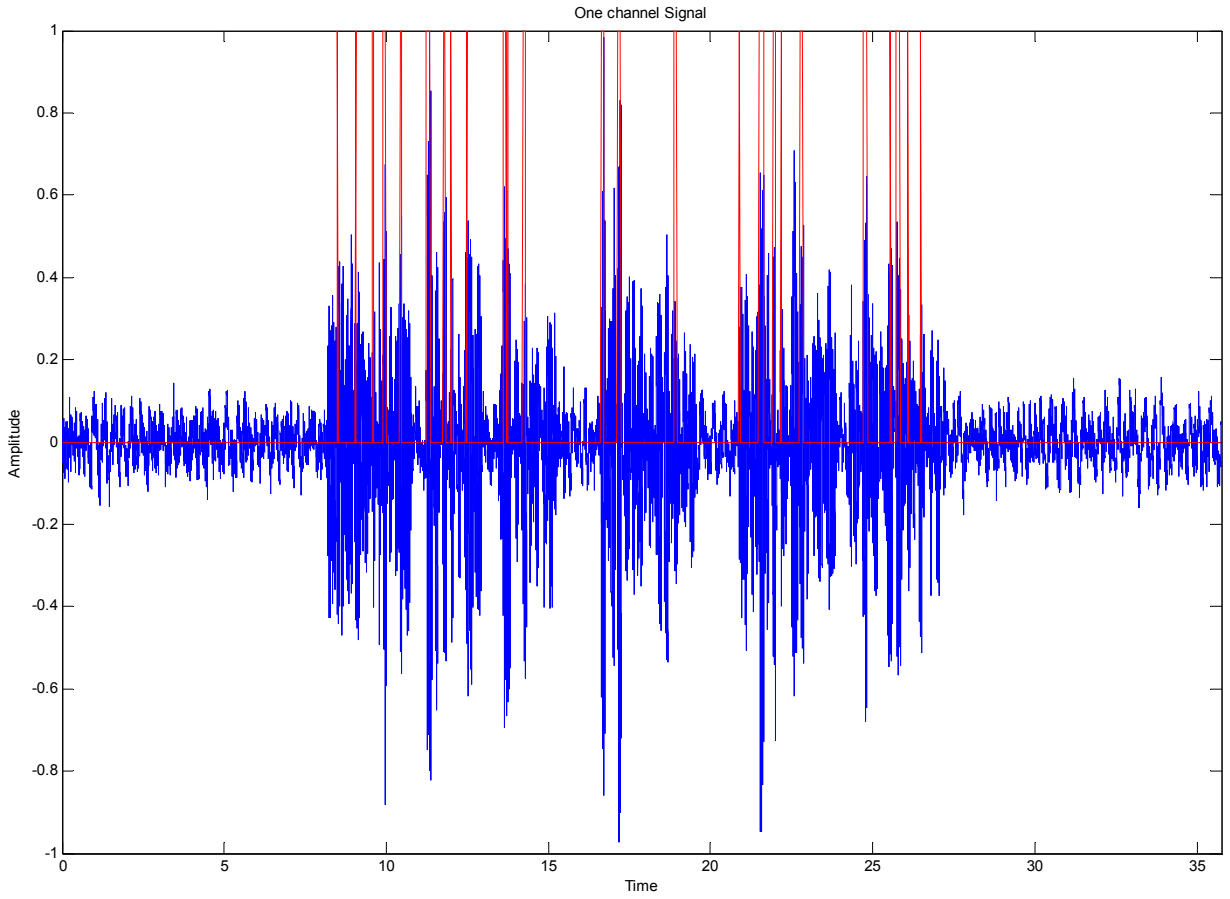


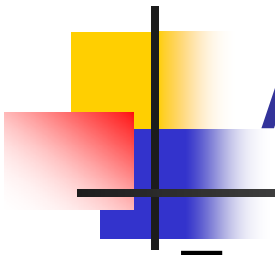
SSL pre-processing

- Pre-processing
  - Packaging the audio signals in frames
  - Conversion to frequency domain
  - Noise suppression
  - Classification signal/pause
  - Rejection of non-signal frames

SSL pre-processing (example)

SSL measurements vs. time



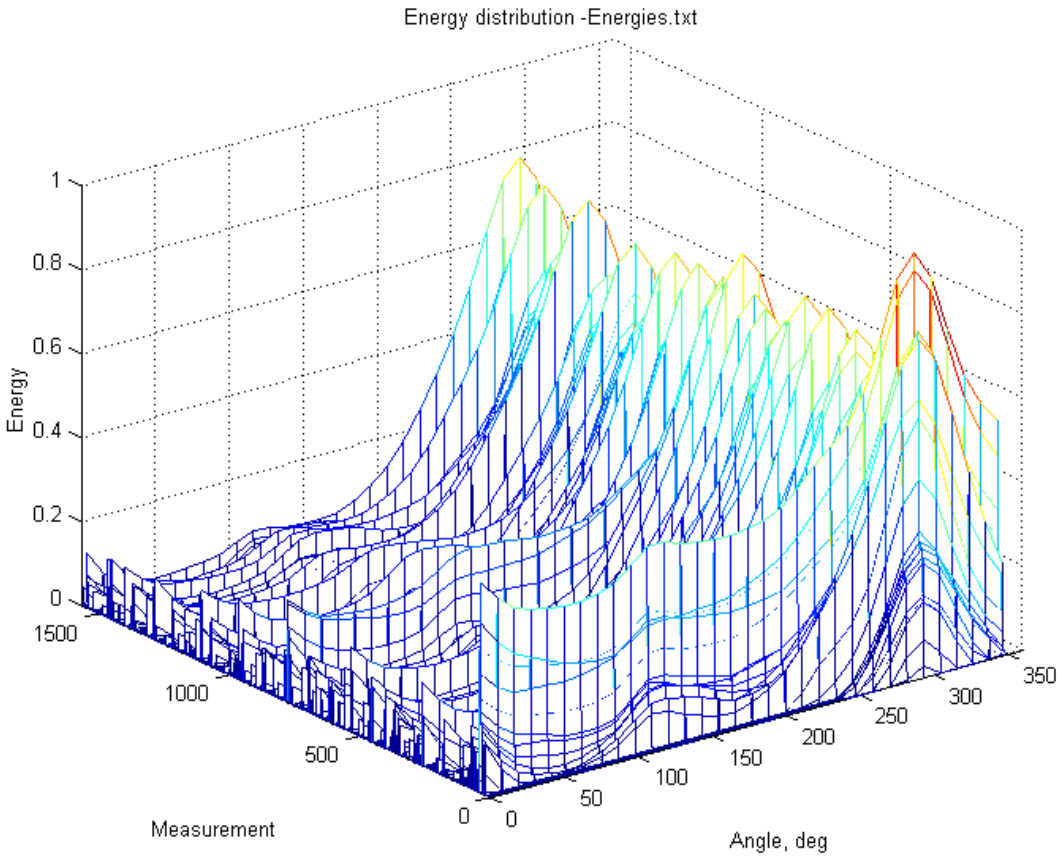


Actual SSL - known algorithms

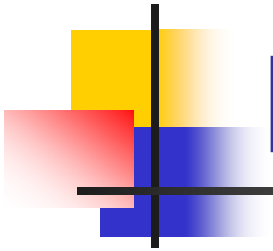
- Two step time delay estimates (TDOA) based
  - Calculate the delay for each microphone pair
  - Convert it to direction
  - Combine the delays from all pairs for the final estimation
- One step time delay estimates (Yong Rui and Dinei Florencio, MS Research)
  - Calculates the correlation function for each pair
  - For each hypothetical angle of arrival, accumulate corresponding correlation strength from all pairs, and search for the best angle
- Steered beam based algorithms
  - Calculate the energy of beams pointing to various directions
  - Find the maximum
  - Interpolate with neighbors for increased resolution
- Others: ICA based, blind source separation, etc.
  - Most of them non real-time

Beamsteering SSL (example)

Energy vs. angle and time, single sound source



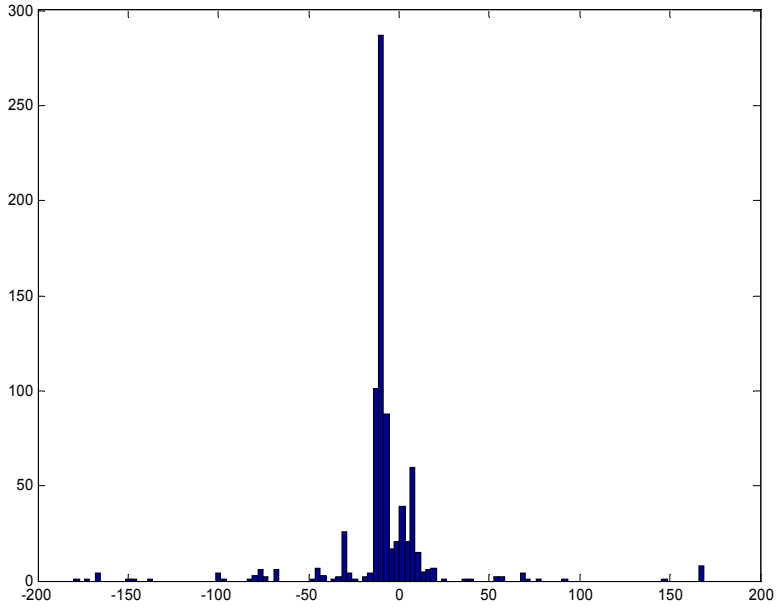
Major factors harming the precision

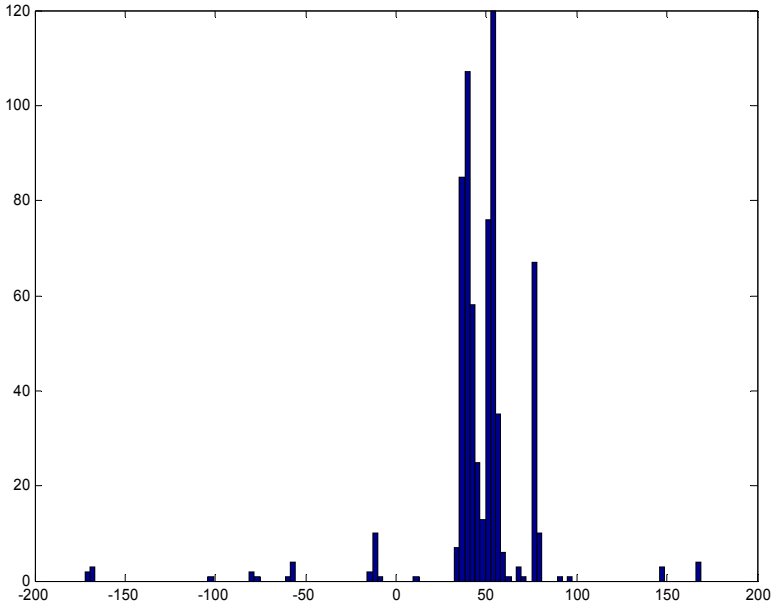


- Ambient noise
  - Smoothes the maximums
  - Hides low-level sound sources
- Reverberations
  - Create additional peaks
  - Lift the noise floor
  - Suppress/enhance some frequencies
- Reflections
  - Create distinct fake peaks with constant location
- All above justify the post-processing phase

SSL with reflections and reverberation – raw data

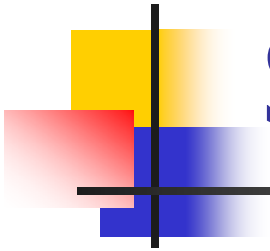
Speakers in conference room (SSL results histogram)





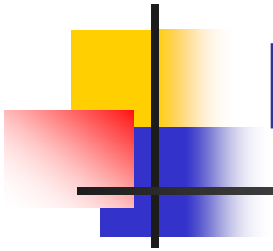
Speaker 1 at -8^{O}: louder voice, less reflections

Speaker 2 at 52^{O}: quieter voice, strong reflections from the white boards



SSL post-processing

- The goals are:
  - To remove results from reflections and reverberation
  - To increase the SSL precision (standard deviation)
  - To track the sound source movement/change dynamics
  - Eventually to provide tracking of multiple sound sources
- Approaches for post-processing of the SSL results
  - Statistical processing
  - Real-time clustering
  - Kalman filtering
  - Particle filtering
- Provides the final result: time, position, confidence level



Real-time clustering of SSL data

- Put each new SSL measurement (time, direction, weight) into a queue
- Remove all measurements older than given life time (~4 sec)
- Place all measurements into a spatially spread 50% overlapping buckets
- Find the bucket with largest sum of weights
- Weighted average the measurements in this bucket
- Calculate the confidence level based on last time, number of measurements, standard deviation

Post-processing results

Single speaker in various positions

Recording conditions:

- Sound room (no noise and reverberation)
- Office (high noise, shorter reverberation, reflections)
- Conference room (less noise, longer reverberation, reflections)

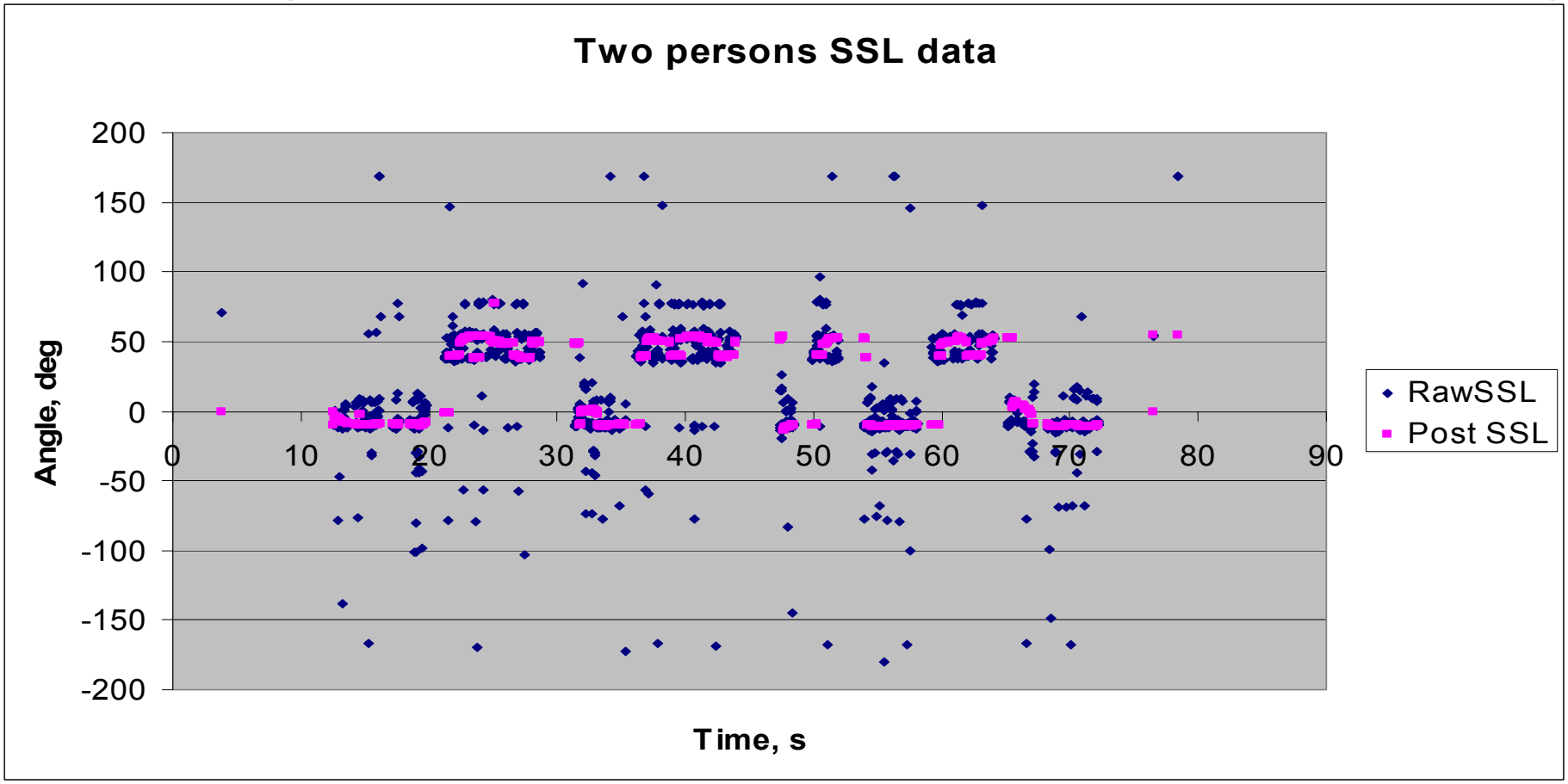
Conditions	Speaker, deg	Bias, deg	StDev, deg	#results
Sound Room	36	-1.6054	0.3857	334
Sound Room	0	1.8722	2.087	319
Sound Room	-21	5.6932	2.4788	292
Office	38	-4.7539	1.3155	407
Office	0	1.6181	0.9687	391
Office	-29	4.7209	0.7511	405
Conf. Room	35	3.4657	0.9699	226
Conf. Room	0	-4.207	2.4308	271
Conf. Room	-43	-5.1692	0.8766	383

All records done with 8 element circular microphone array for meetings recording

Post-processing results (2)

Two speakers in fixed positions

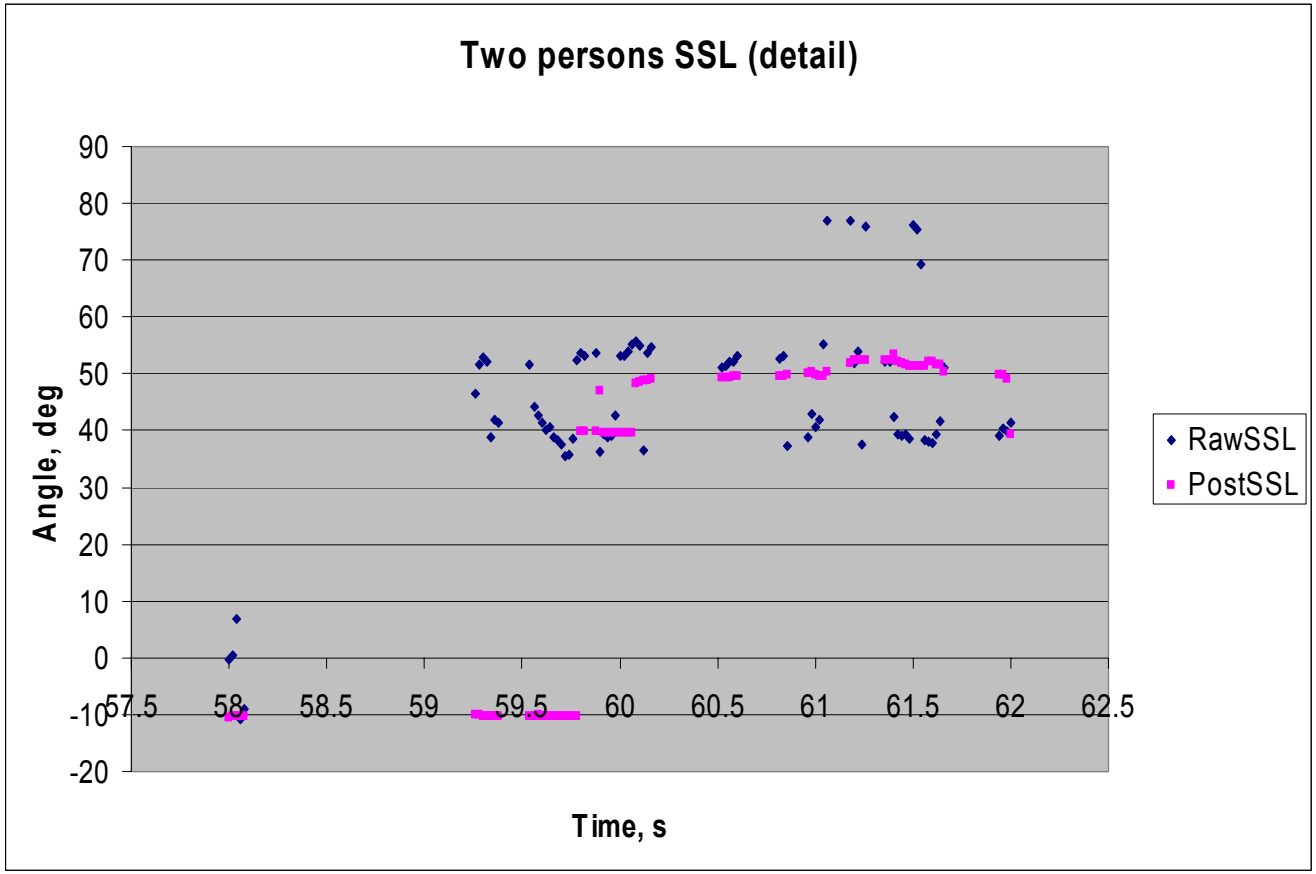
Recording conditions: conference room, speakers at -8 and 52 deg



Post-processing results (3)

Two speakers in fixed positions

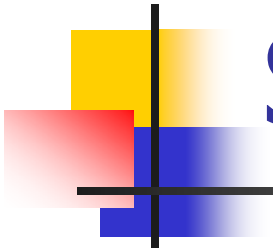
Recording conditions: conference room, speakers at -8 and 52 deg



Speaker switching at second 59

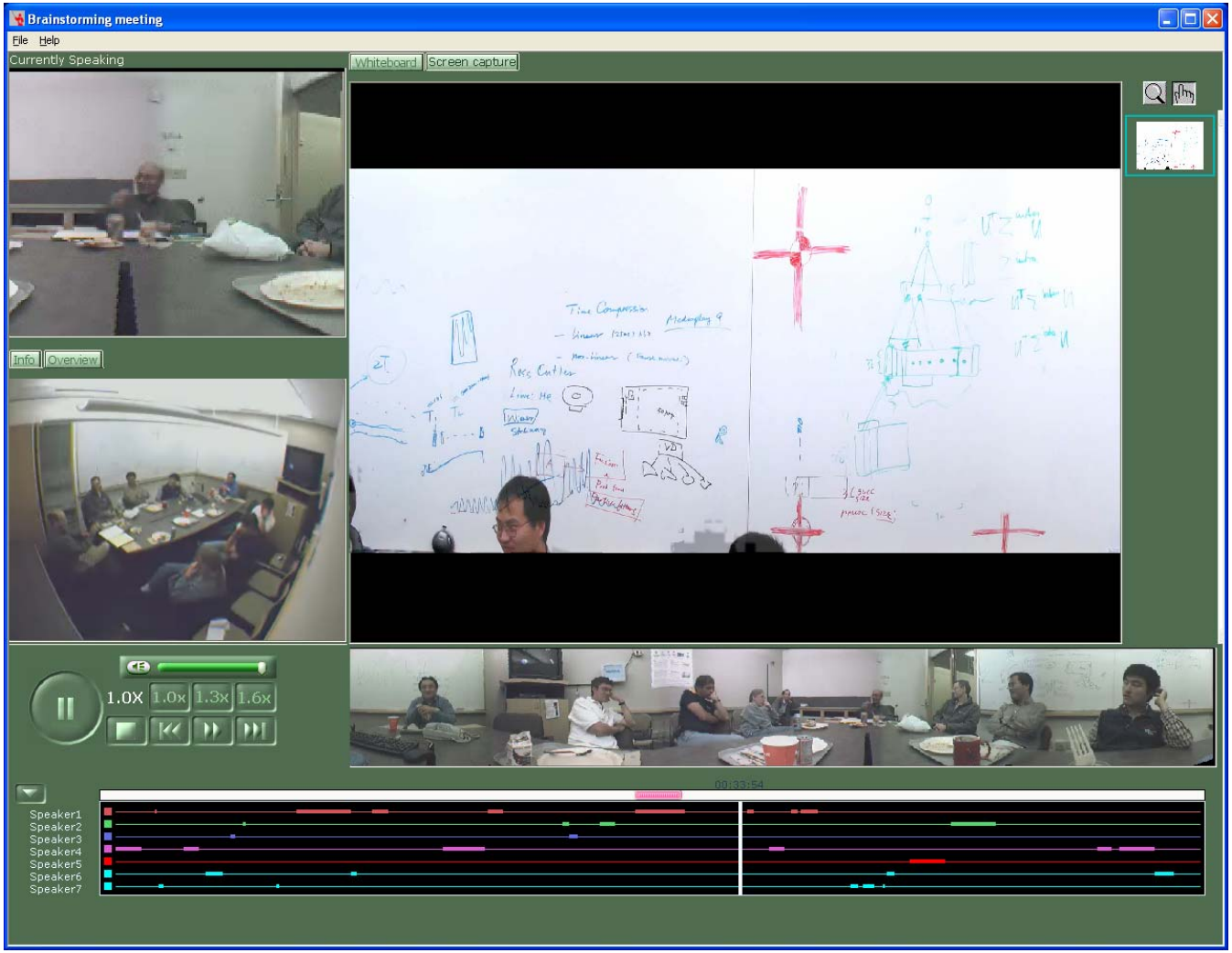
Post-processing delay: ~400 ms

Applications for MicArrays and Sound Source Localization



- Sound capturing during meetings
  - Provides direction to point the capturing beam
  - Assists the Virtual director for speaker view (real-time)
- Meeting post-processing
  - Assists speaker clustering
  - Meeting annotation using rough ASR (requires good sound quality)
  - Meeting transcription with precise ASR
- Recorded meetings viewing/browsing
  - Audio timeline: suppress some audio tracks, navigation by speaker (based on the speaker clustering)
  - Good sound quality - better user experience
  - Good sound quality – search by phrases or keywords with ASR
  - SSL data assisted virtual director for speaker view (play-time)

Meetings browser (example)



Meetings browser (detail)

- Audio timeline