Dia: AutoDirective Audio Capturing Through a Synchronized Smartphone Array

Sanjib Sur

Teng Wei and Xinyu Zhang

University of Wisconsin - Madison

Multimedia applications in smartphones

- Growing mobile multimedia applications due to pervasiveness of smartphones
- Most apps are standalone


- Standalone smartphones are not suitable for *demanding* multimedia applications

Demanding multimedia recording applications

- Smart conferencing

- Autonomous lecture recording

Existing solutions

Polycom QDX 6000

Polycom CX5000

- Cost ~ $4000 - $5000
- Weight ~ 3 Kg.
- Requires dedicated infrastructure (lacks portability)

Synchronized smartphone array

- Goal

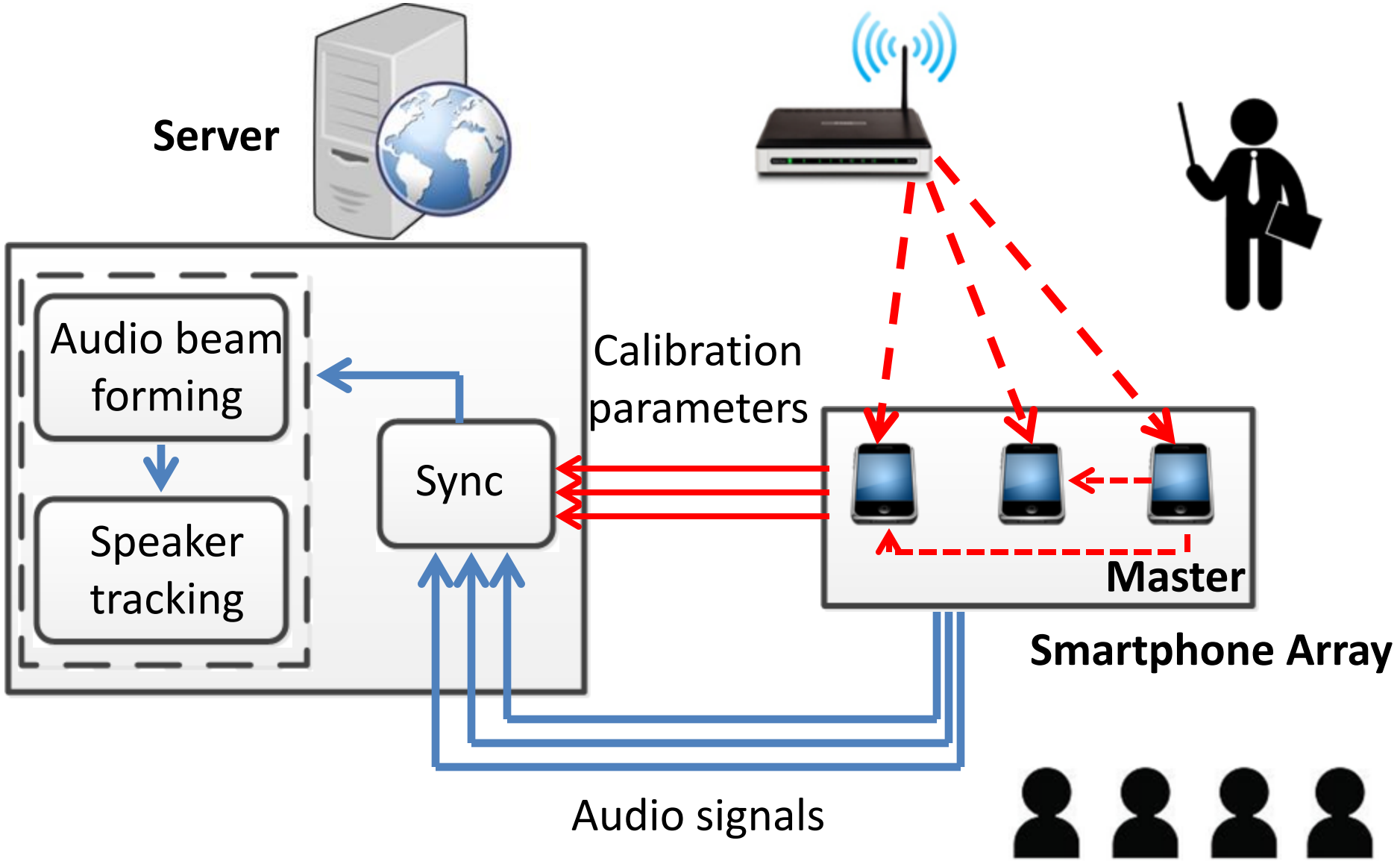
  Leverage multiple smartphones' microphones to realize smart conferencing/lecture room recording

  - Practical audio beamforming
  - Robust speaker localization

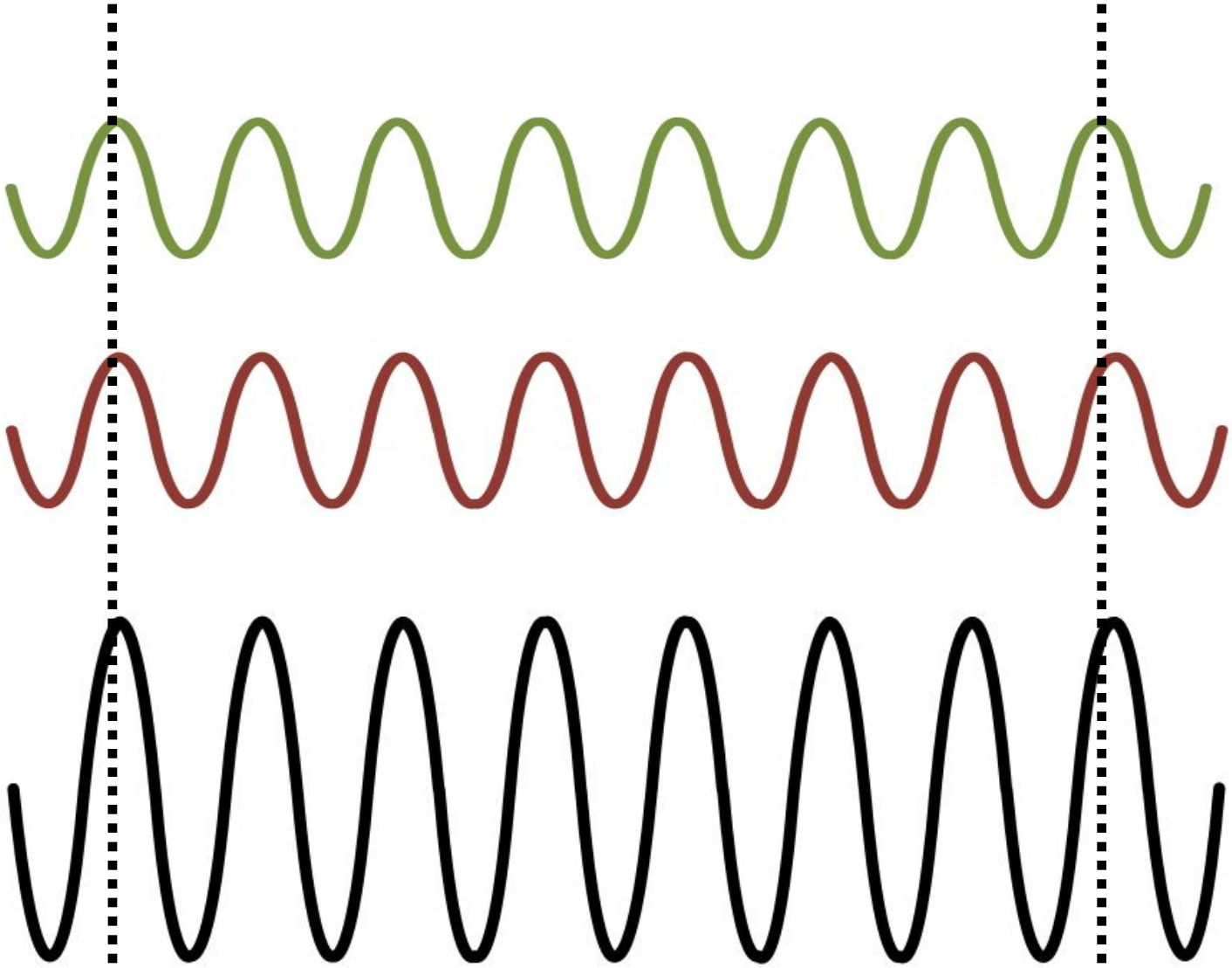
- Challenge

  Precise audio I/O synchronization

Dia: System overview

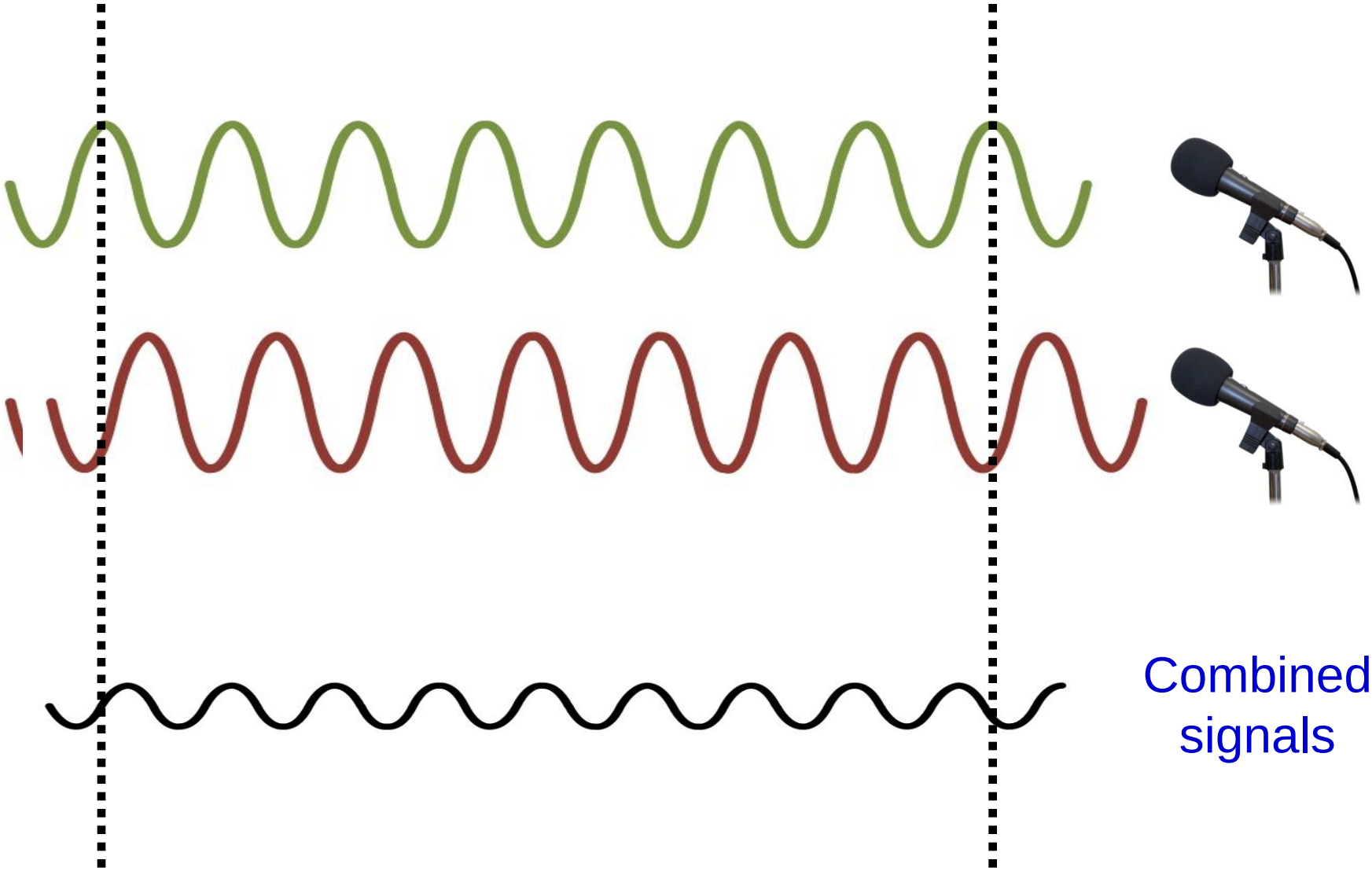


Why synchronization?



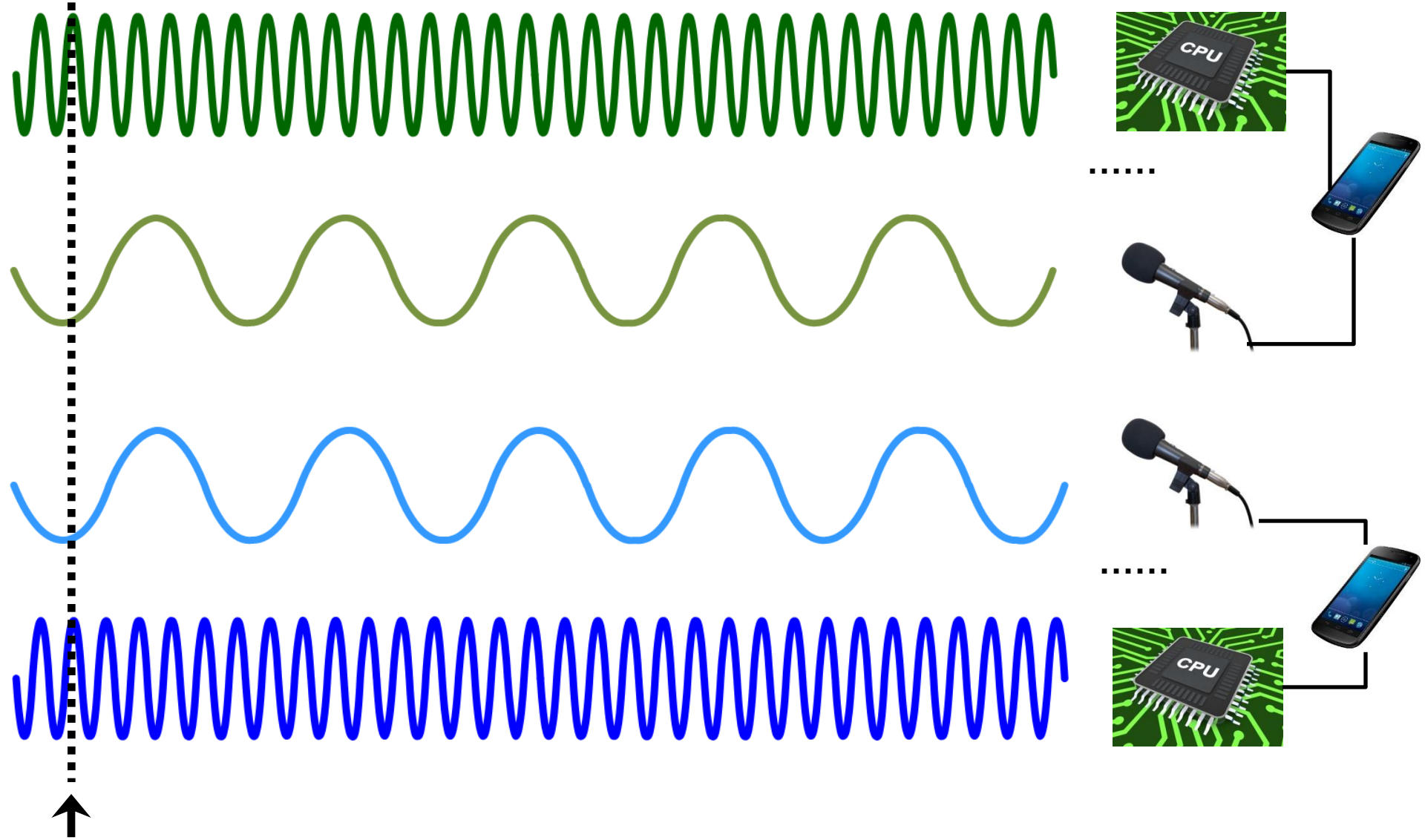
Combined signals

Why synchronization?



Combined signals

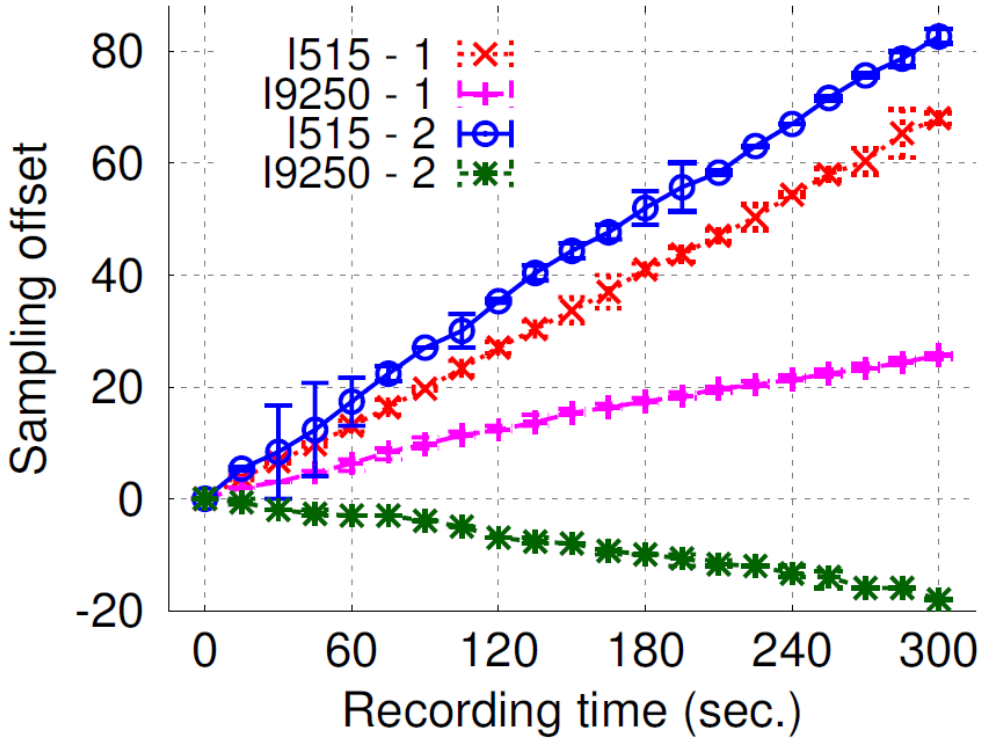
Independent CPU and audio I/O clocks



Clock synchronization

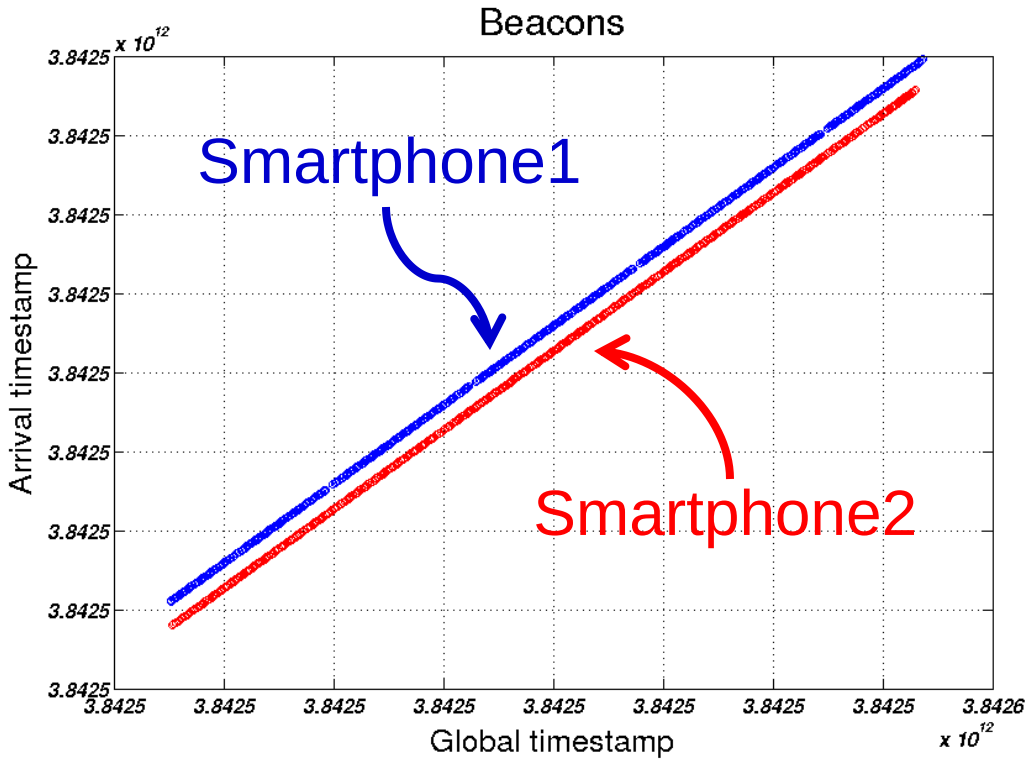
- Problem

  Because of independent CPU and Audio I/O clock, synchronizing only CPU clock at application level is not sufficient



Our solution: Two-level synchronization

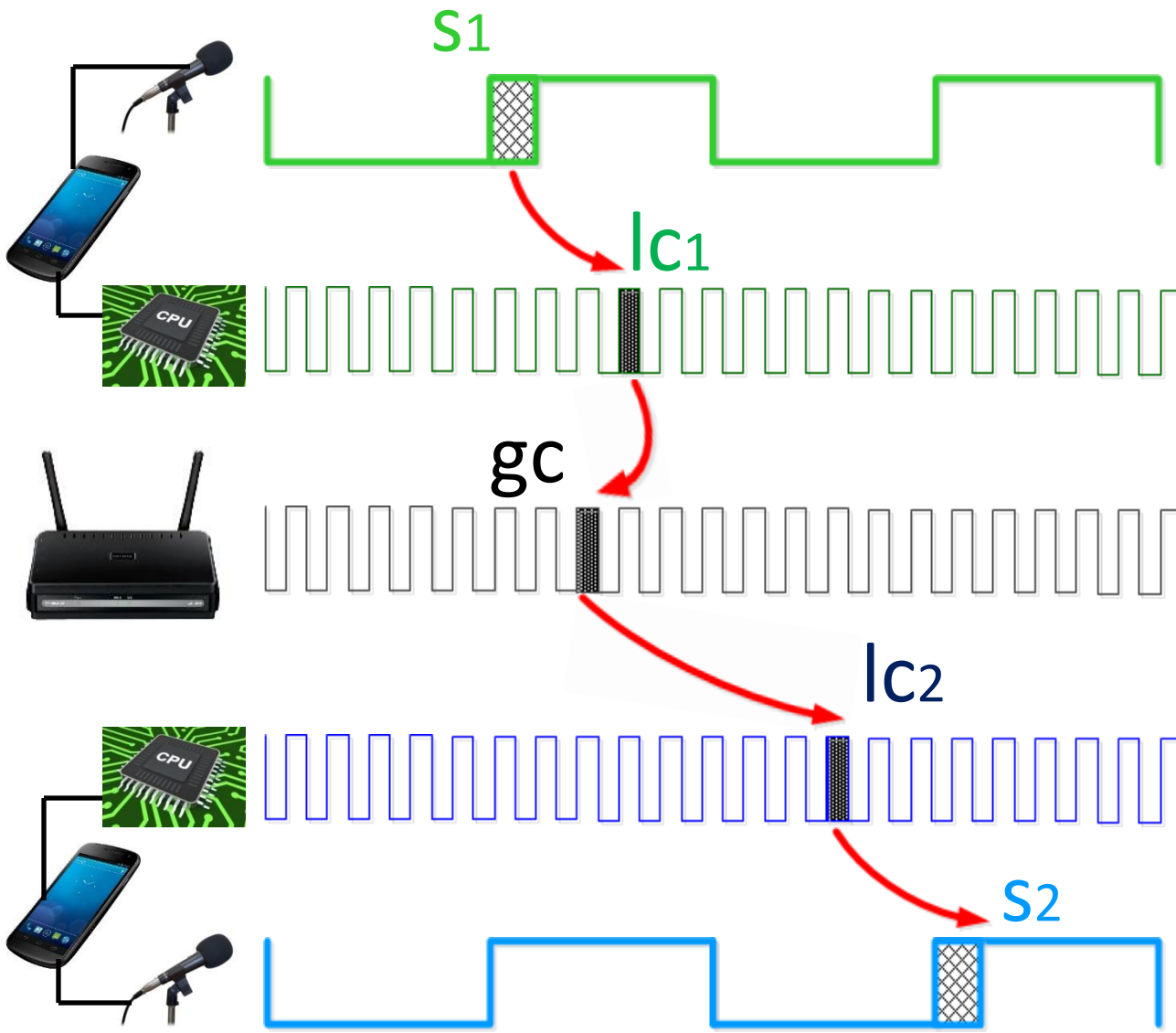
- Observations
  - CPU clock ∞ Global clock
  - Audio I/O clock ∞ CPU clock



$$LC^{cpu}(t) = a(\delta) \, . \, GC(t) + b(\delta)$$

$$LC^{audio}\left(LC^{cpu}(t)\right) = \alpha(\delta) \, . \, LC^{cpu}(t) + \beta(\delta)$$

Two-level synchronization



$$s_1 = \alpha_1 . lc_1 + \beta_1$$

$$lc_1 = a_1 . gc + b_1$$

$$lc_2 = a_2 . gc + b_2$$

$$s_2 = \alpha_2 . lc_2 + \beta_2$$

Estimating the timing models

$$LC^{cpu}(t) = a(\delta) \, . GC(t) + b(\delta)$$

- Observe first *n* tuples, $\{GC(t_j), LC^{cpu}(t_j)\}\ j = 1, 2, \ldots, n$
- Run a linear regression to estimate $a \ and \ b$

$$LC^{audio}\left(LC^{cpu}(t)\right) = \alpha(\delta) \, . LC^{cpu}(t) + \beta(\delta)$$

- Observe *m* tuples, $\{LC^{cpu}(t_j), k\} \quad k = 1, 2, \ldots, m$
- Run another linear regression to estimate $\alpha \ and \ \beta$

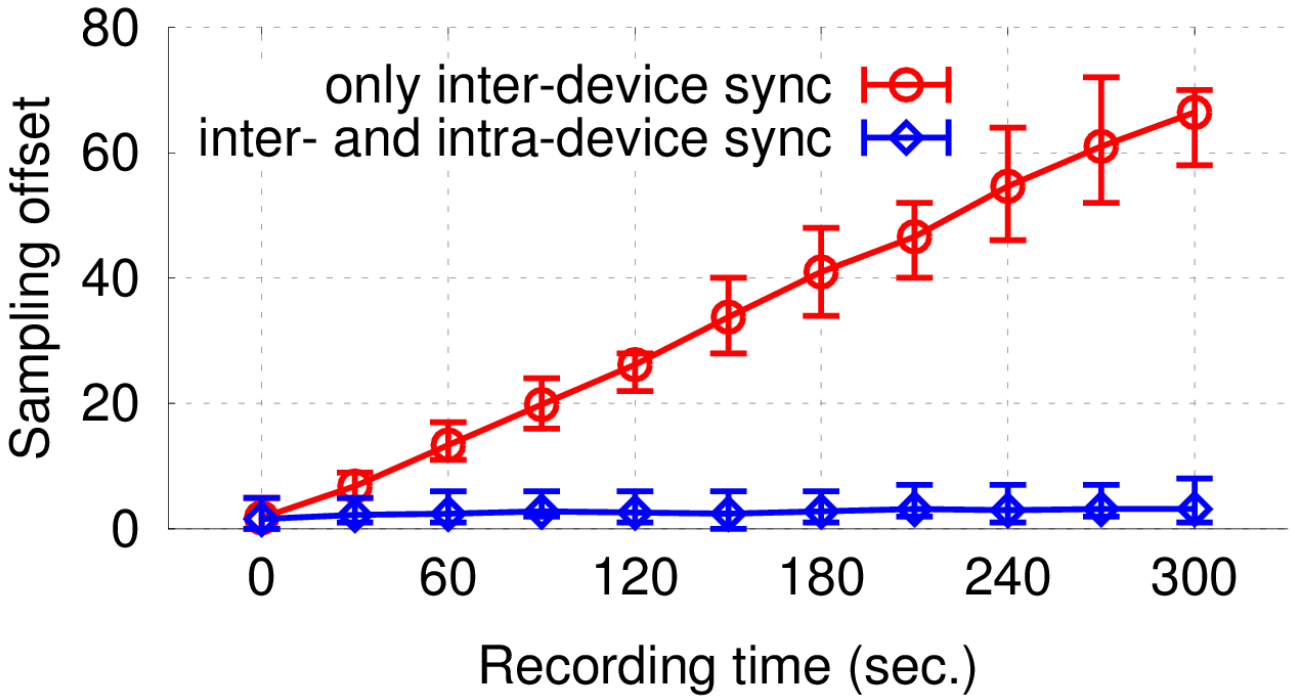
Implementation

- Implemented in 8 Samsung Galaxy Nexus smartphones

- Modified Broadcom wireless drivers and Tinyalsa audio drivers in Linux kernel of Android OS

- Used Desktop PC as server to process the synchronized audio signals

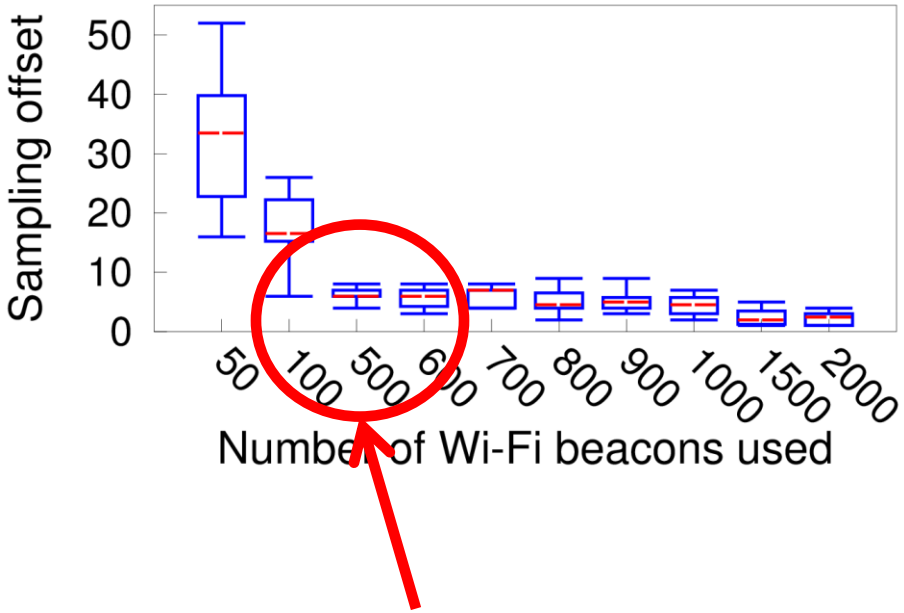
Synchronization performance



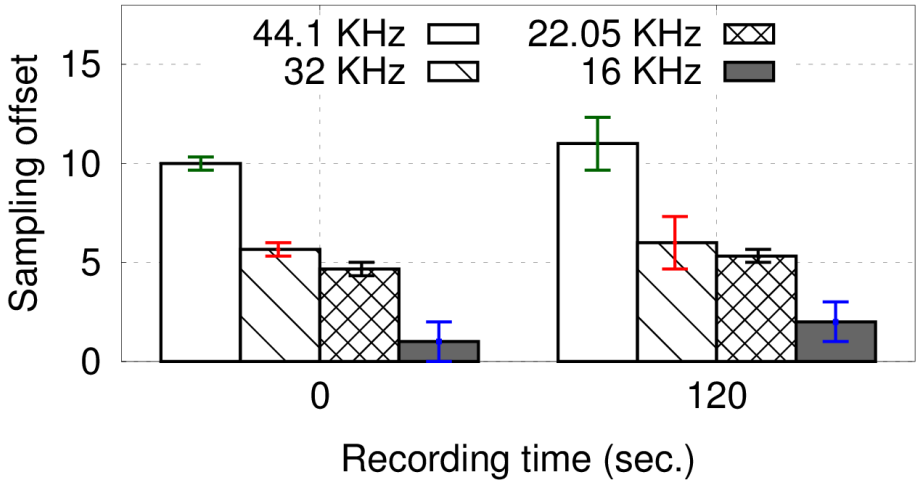
Accuracy within 2 ~ 3 samples at 16 kHz = 187.5 μS

Synchronization performance

- Impact of initialization time
- Impact of audio sampling frequency

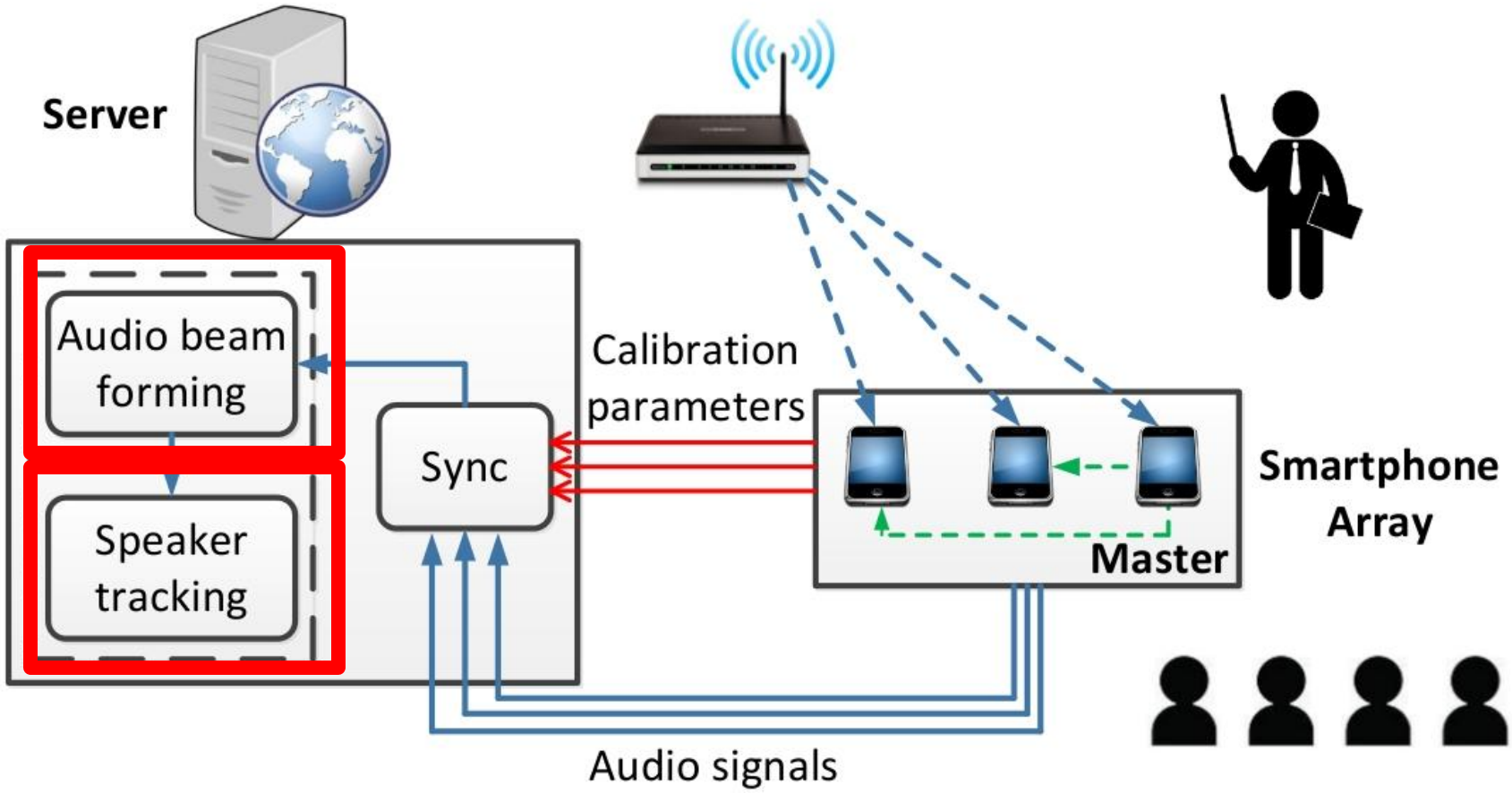


Only 500 beacons, initial setup time ≈ 50 sec.

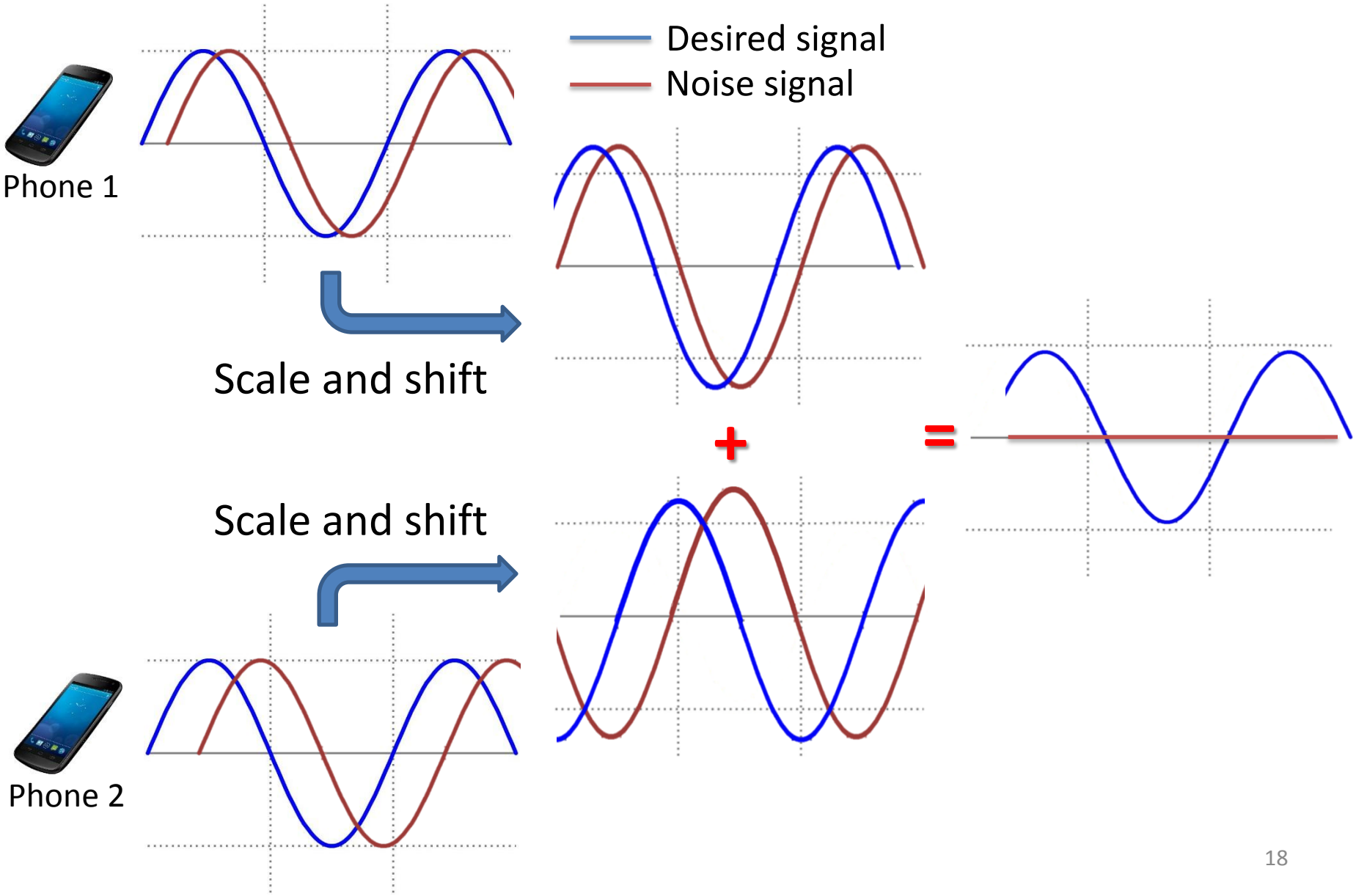


Audio sampling rate invariant

Application to autodirective audio capturing

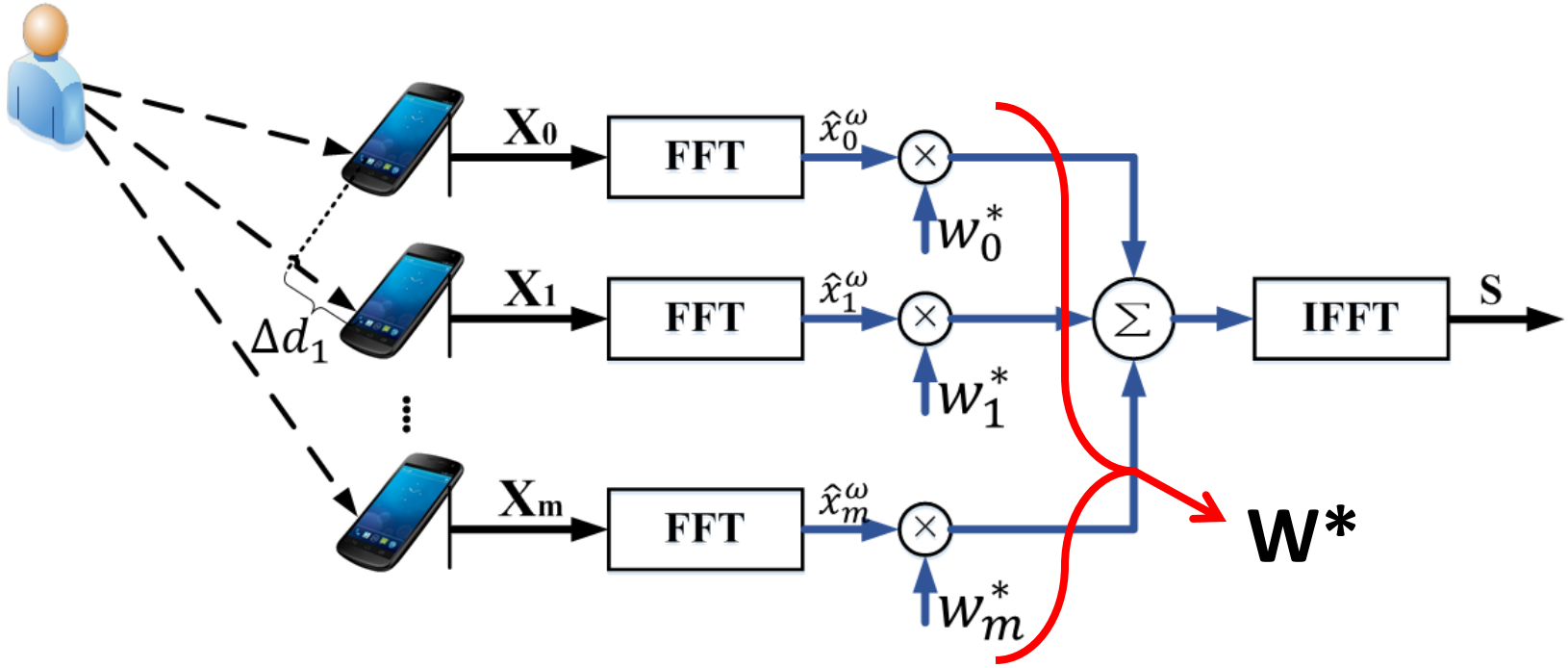


Audio Beamforming



Audio Beamforming Algorithm

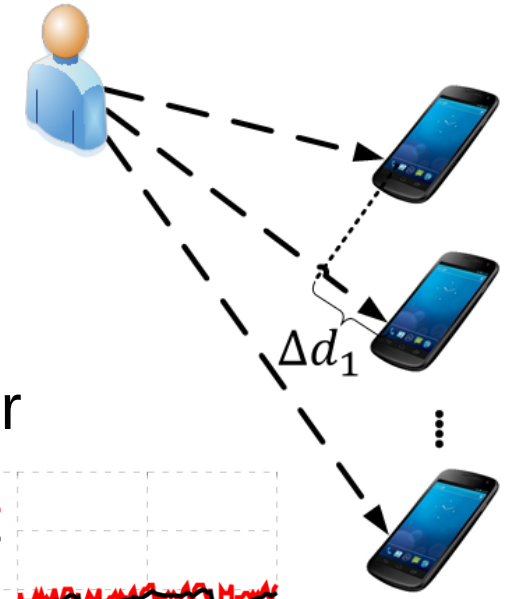
- Minimum Variance Distortionless Response (MVDR)



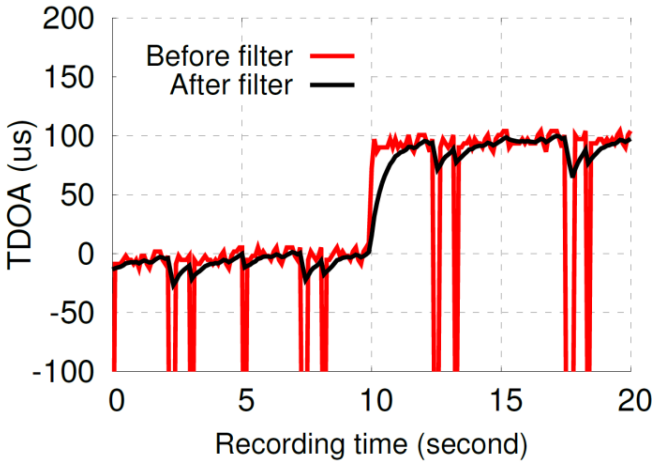
  - MVDR theorem: to find the W* that
    - Steers the “beam” to the desired direction
    - Minimizes the output energy of noise signals

Speaker tracking using Time Difference Of Arrival

- TDOA estimation
  - Generalized cross-correlation with phase transformation (GCC-PHAT)
  - The phase maximizing the cross-correlation corresponds to the TDOA

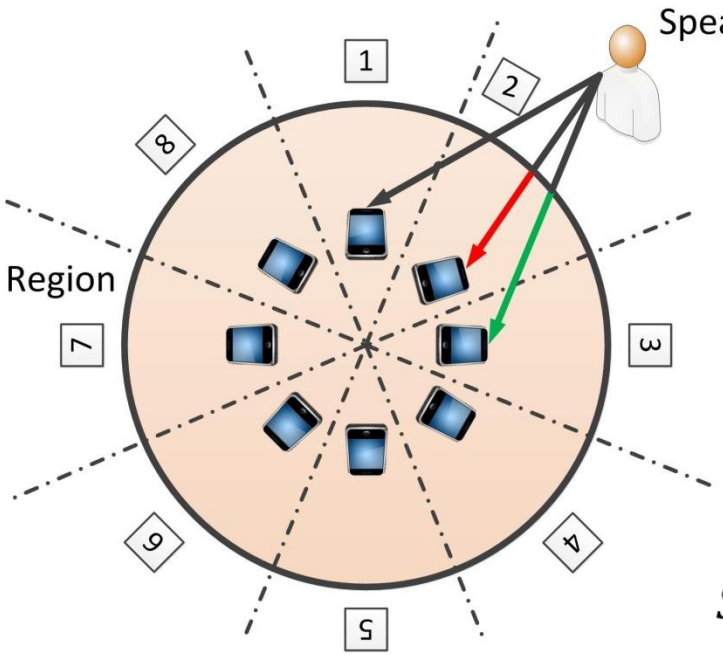


- Problem:
  - Instability of the estimation result
- Solution: Adaptive time-domain linear filter
  - Remove TDOA outliers
  - Tradeoff between Robustness and Latency



Speaker tracking: Binary Mapping Algorithm

How do we find the location of speaker without knowing the distance between smartphones?



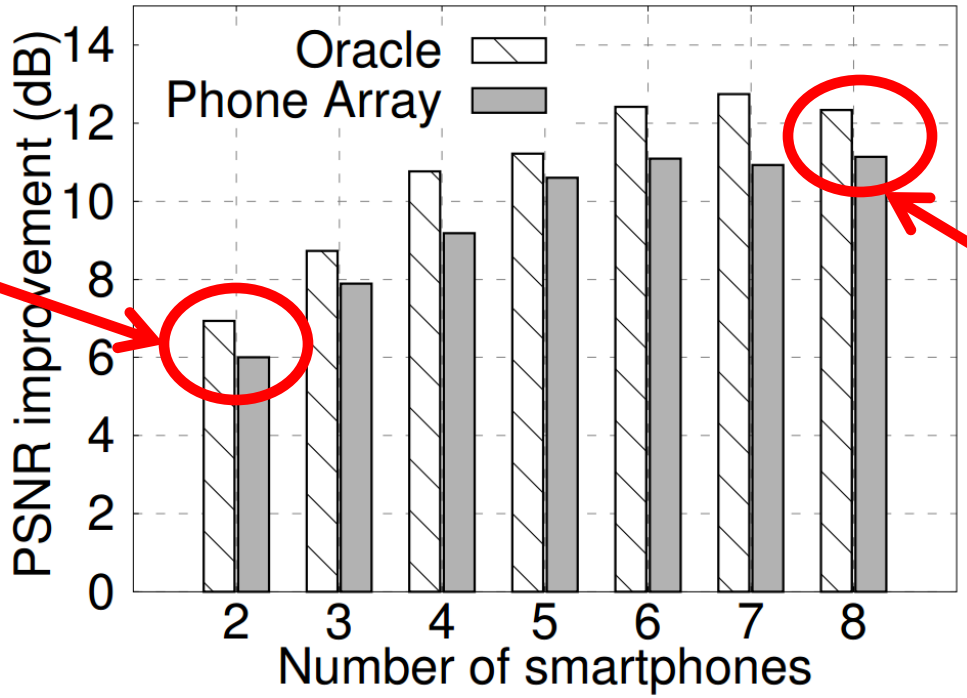
$$S^2 = [-1, \pm 1, 1, 1, 1, 1, 1]$$

- Region Signature S: unique binary vector for each region
- Use minimum Euclidean distance for region signature matching

Beamforming performance

- Beamforming gain from a smartphone array
  - Gain scales with the growing number of microphones

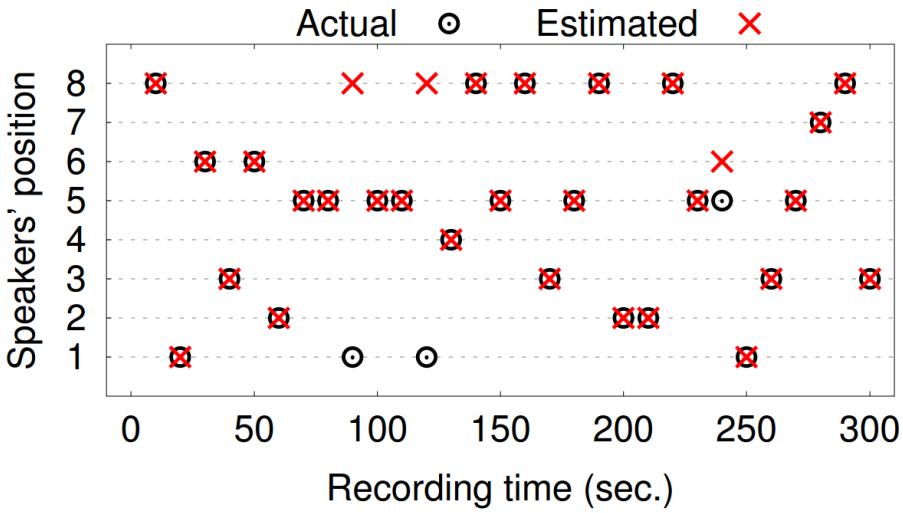
6dB of improvement!

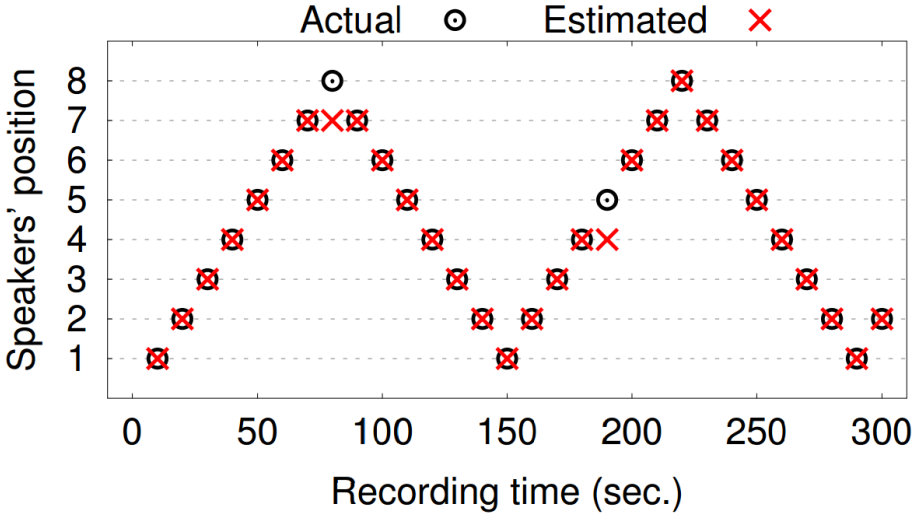


11dB of PSNR improvement!

Accuracy of speaker tracking

- In round-table conference scenario
- In lecture room scenario





~90 - 93% of accuracy!

Conclusion

- Precise audio sample synchronization enables many distributed audio sensing applications in smartphones
- Practical autodirective audio capturing and speaker tracking by ad-hoc cooperative smartphones
- Future work
  - Cooperative audio and visual recording system

Thank you!

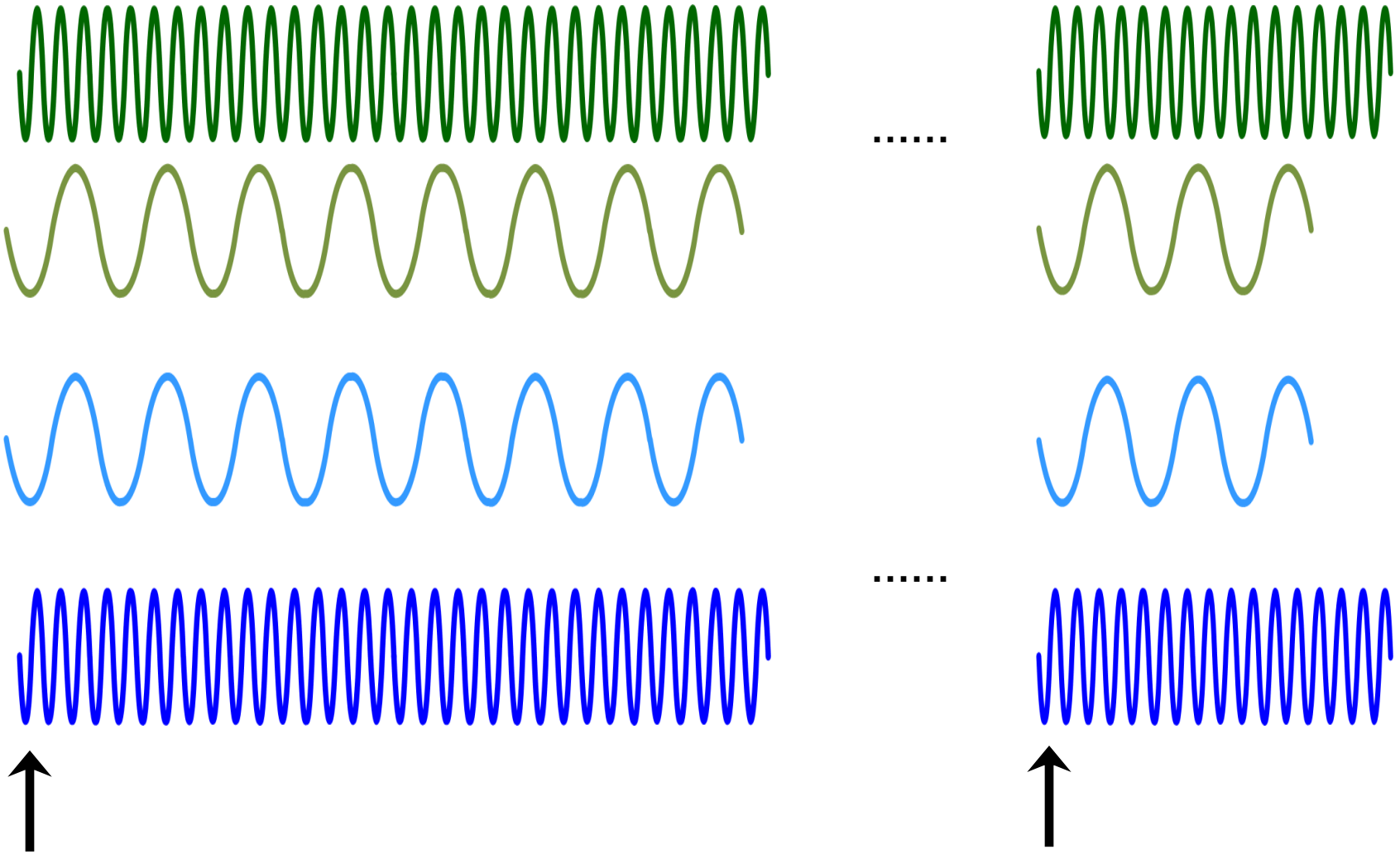
Backup slides

How precise?

- Human voice can range from 500 Hz – 2 kHz
- Equivalently the synchronization timing offset between two microphones ≤ 1/(2 * 2000) = 250 µS

Requirement: Audio synchronization timing offset should be below 250 µS!

Independent CPU and audio I/O clocks



Independent CPU and audio I/O clocks