Device-Free Gesture Tracking Using Acoustic Signals

Wei Wang[†]

Alex X. Liu[†‡]

Ke Sun[†]

[†]Nanjing University, [‡]Michigan State University

MobiCom'16 Oct 4^{th}, 2016

Motivation

It is difficult to input on smart watches

Limitation of Prior Arts

Google Soli designs specialized 60GHz chips for gesture input

Problem Statment

Can we build software-based to replace specialized hardware?

... and meet these design goals

- High accuracy (mm-level)
- Low latency ($<$ 30 ms)
- Low computational cost (works on mobile devices)
- Low energy

Problem Statment

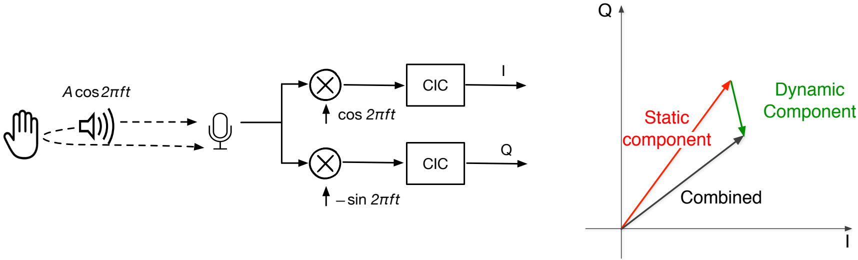
Can we build software-based to replace specialized hardware?

. . . and meet these design goals

- High accuracy (mm-level)
- Low latency ($<$ 30 ms)
- Low computational cost (works on mobile devices)
- Low energy

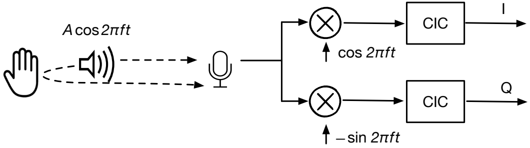
Can we do that?

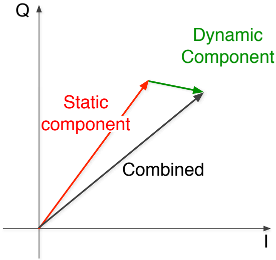
Basic Idea



- Use plain cos wave rather than impulses
- Measure the phase rather than Doppler shifts
- Decompose the received signal in vector space rather than in the time/frequency domain

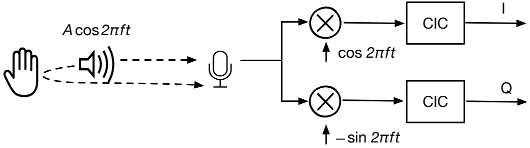
Basic Idea

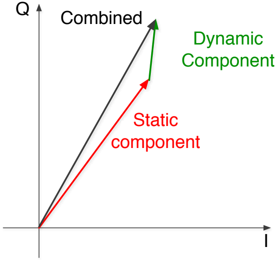




- Use plain cos wave rather than impulses
- Measure the phase rather than Doppler shifts
- Decompose the received signal in vector space rather than in the time/frequency domain

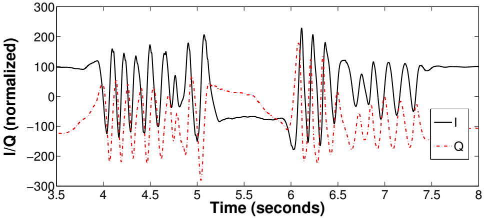
Basic Idea

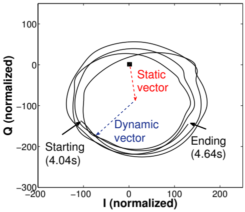




- Use plain cos wave rather than impulses
- Measure the phase rather than Doppler shifts
- Decompose the received signal in vector space rather than in the time/frequency domain

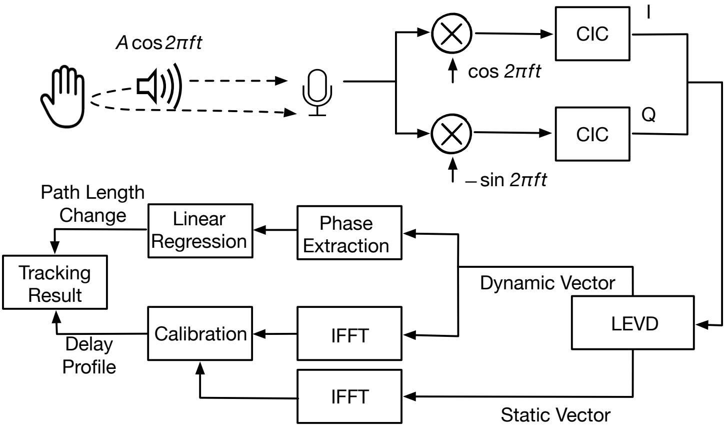
Real World Signals



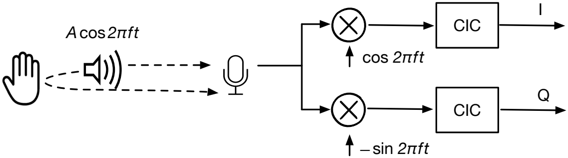


- Sound frequency 17 $\sim$ 22 kHz $\rightarrow$ wavelength 1.5 $\sim$ 2 cm
- Path length change of wavelength $\rightarrow$ phase change of 2π
- Move 1.25 mm $\rightarrow$ path length change 2.5 mm $\rightarrow$ phase change $\pi/4$
- Phase change direction $\rightarrow$ movement direction

System Architecture

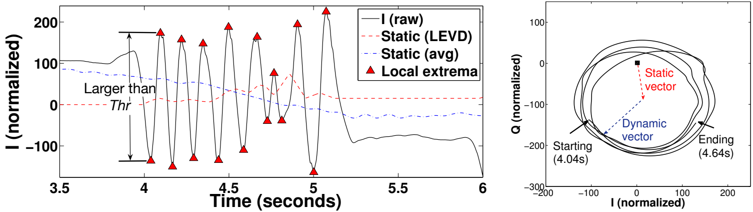


Baseband Conversion



- Sound generation
  - 16 single tones 17 $\sim$ 22 kHz (350 Hz interval)
  - Sample rate 48 kHz
- Recording using two microphones at 48 kHz
- Mix (multiply) with the transmitted frequency
- CIC filtering
  - Low computation (no multiplication)
  - Kills neighboring frequencies

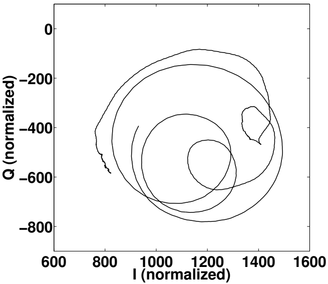
Separating the Static and Dynamic



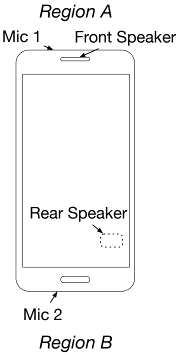
- Static component
  - LOS path, static objects (tables, walls)
  - Change slowly
- Dynamic component
  - Hand movements
  - A few to tens of Hz
- Use LEVD to detect peaks and estimate static component

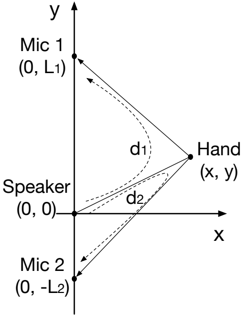
Error Sources

- Multipath effect
  - Refections of nearby objects
  - Mitigated through frequency diversity



2D tracking

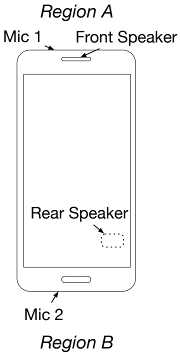


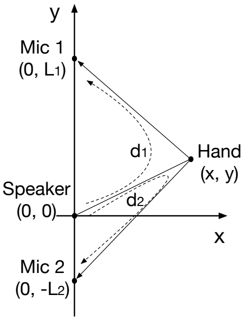


- Measure distance from two microphones
- Solve the location of the hand

Initial position?

2D tracking



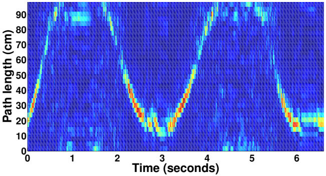


- Measure distance from two microphones
- Solve the location of the hand

Initial position?

Measuring Absolute Distance

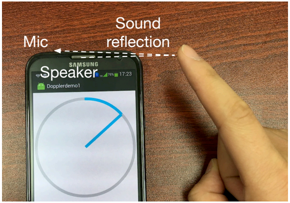
- Absolute path length for
    - Initial position
    - Prevent error accumulation



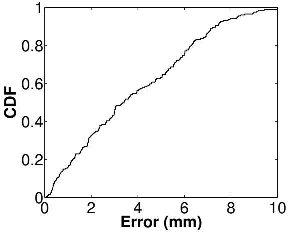
- Getting the absolute path length
    - Use IDFT on Dynamic component to get delay profile of the hand
    - Use IDFT on Static component to calibrate
    - Resolution $\sim$ 4 cm

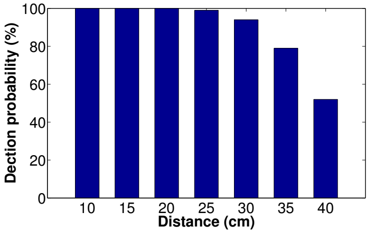
Implementation

- Implemented on
  - Android
    (C and Java with NDK)
  - iOS
    (Object C with vDSP)
- Parameters used
  - 48 kHz sampling rate
  - 16 frequencies
  - Only 1D for iOS



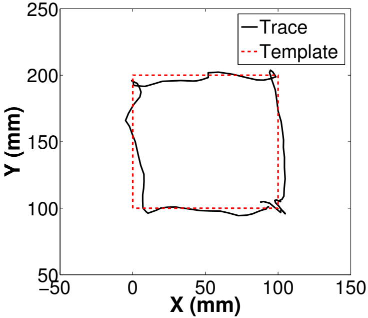
Result – 1D Tracking

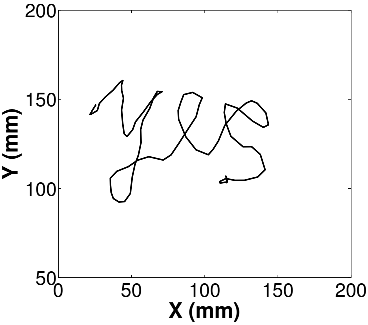




- Average error of 3.5 mm for movement of 10 cm
- Operational range $\sim$ 30 cm
- Detects single finger movements of 5 mm within 25 cm

Result – 2D Tracking



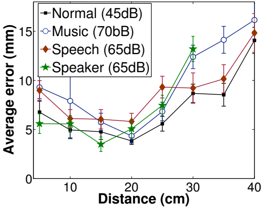


- 2D drawing error of 4.6 mm (with user compensation)
- Word recognition accuracy of 91.2% for 5 users

Robustness

- Robust to noise interferences
  - Resistant to normal noise of speech and music ($\sim$ 65 dB)
  - Can play sound from the same speaker used for tracking

- Limtations
  - Can be interfered by nearby movements
  - Can be interfered by nearby ultrasound devices



Latency and Energy Cost

- Latency and computational cost
  - Operates on 512 samples (10.7 ms segments)
  - Processing time
    - Samsung S5 (Android NDK) $\sim$ 4.32 ms
    - iPhone 6s (iOS vDSP) $\sim$ 0.3ms – 3% CPU
- Energy cost
  - iPhone 6s battery can sustain for 10.5 hours

Conclusions

- Software based solutions can do precise gesture tracking on existing devices
- Our system achieves design goals
  - High accuracy ($\sim$ 3.5 mm)
  - Low latency ($<$15 ms)
  - Low computation cost ($\sim$ 3% CPU cost on iPhone 6s)
  - Low energy (“Low” energy impact rated by Xcode)

Q & A

Thank you!

Questions?