Millimeter-Wave Beamtracking in the COSMOS Testbed Using Analog AI Accelerators

Panagiotis Skrimponis

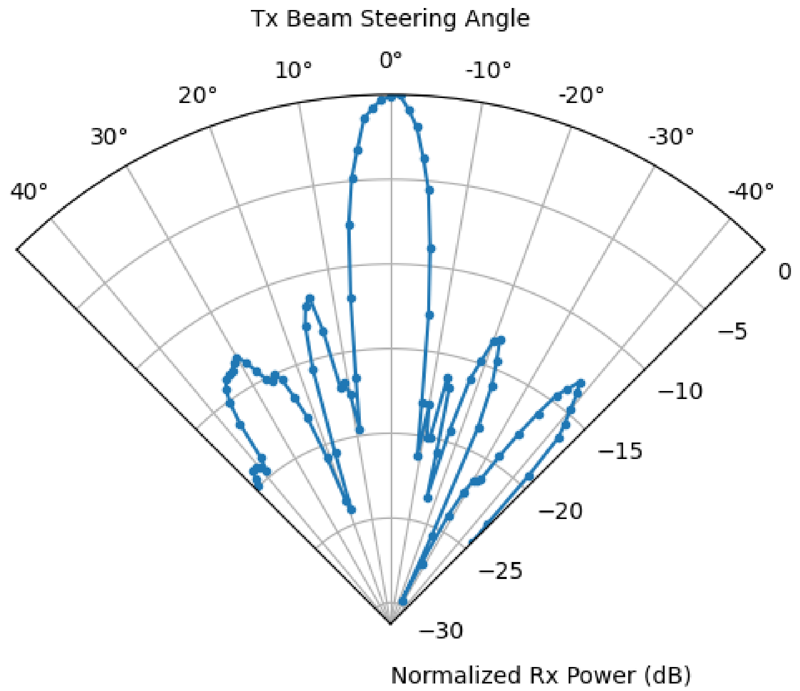
ps3857@nyu.edu

Overview

- Motivation
- COSMOS
- Mobility Model
- AI Beam Tracking

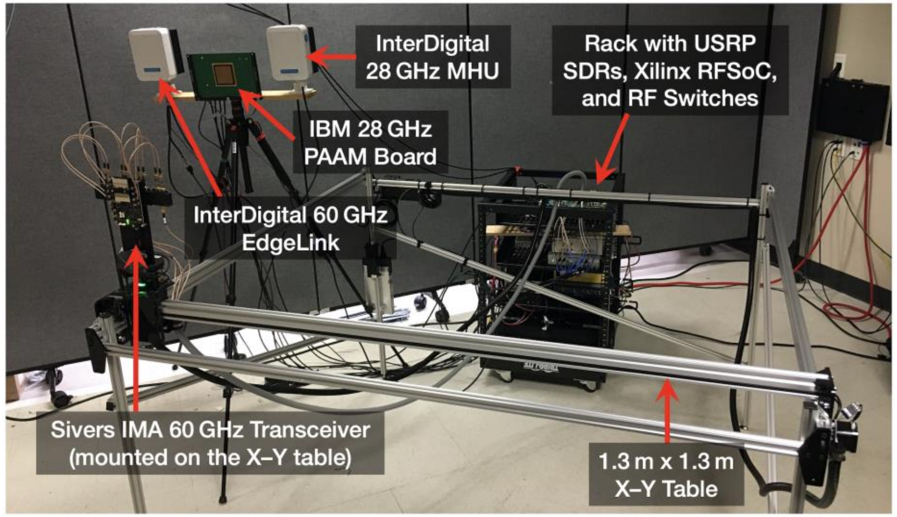
Motivation

- Power consumption: Key challenge for mmWave systems
  - Severe blockages and free-space path-loss (FSPL)
  - Large numbers of antenna elements
  - Poor device efficiency
- Beamforming and multi-beam tracking
  - Compensate for FSPL
  - Mitigate the effect of blockages
  - **High power and time overhead** ☹
- This work
  - Selective beam-tracking
  - Analog AI accelerators
  - **Low power and time overhead** ☺

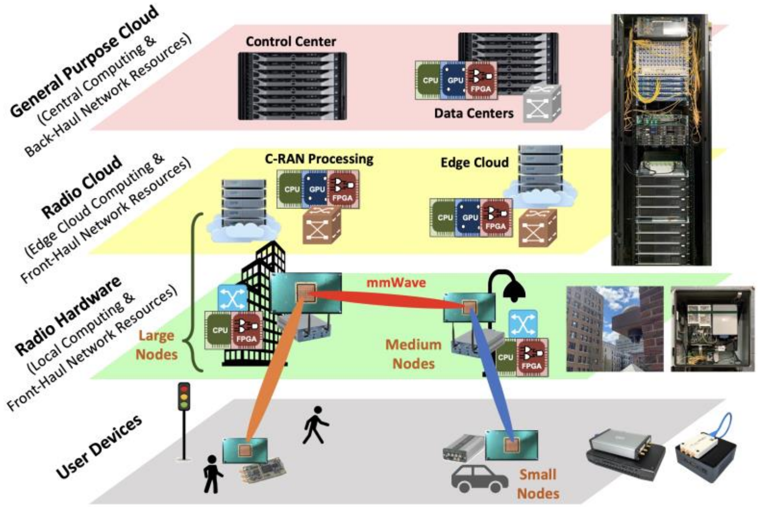


COSMOS

- **Open-access** city-scale wireless testbed
- **Advanced mmWave** SDRs at 28 and 60 GHz
- Fully **open-source** and **remotely programmable**
- Useful for practical experiments



(a) SB1 indoor environment



(b) COSMOS' multi-layered computing architecture [1], [2]

[1] D. Raychaudhuri, et al. "Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless." in Proc. ACM MobiCom'20.
[2] T. Chen, et al. "Programmable and open-access millimeter-wave radios in the PAWR COSMOS testbed." in Proc. ACM WiNTECH@MobiCom'21.

Measurement Setup

- Xilinx RFSoC ZCU111
- Sivers IMA 60 GHz transceiver
- Mounted on X-Y tables
  - Area: 1.3×1.3 m^2
  - Rotation: ±45°
  - Distance: ~20m
- JavaScript-based web interface
- Live camera streaming

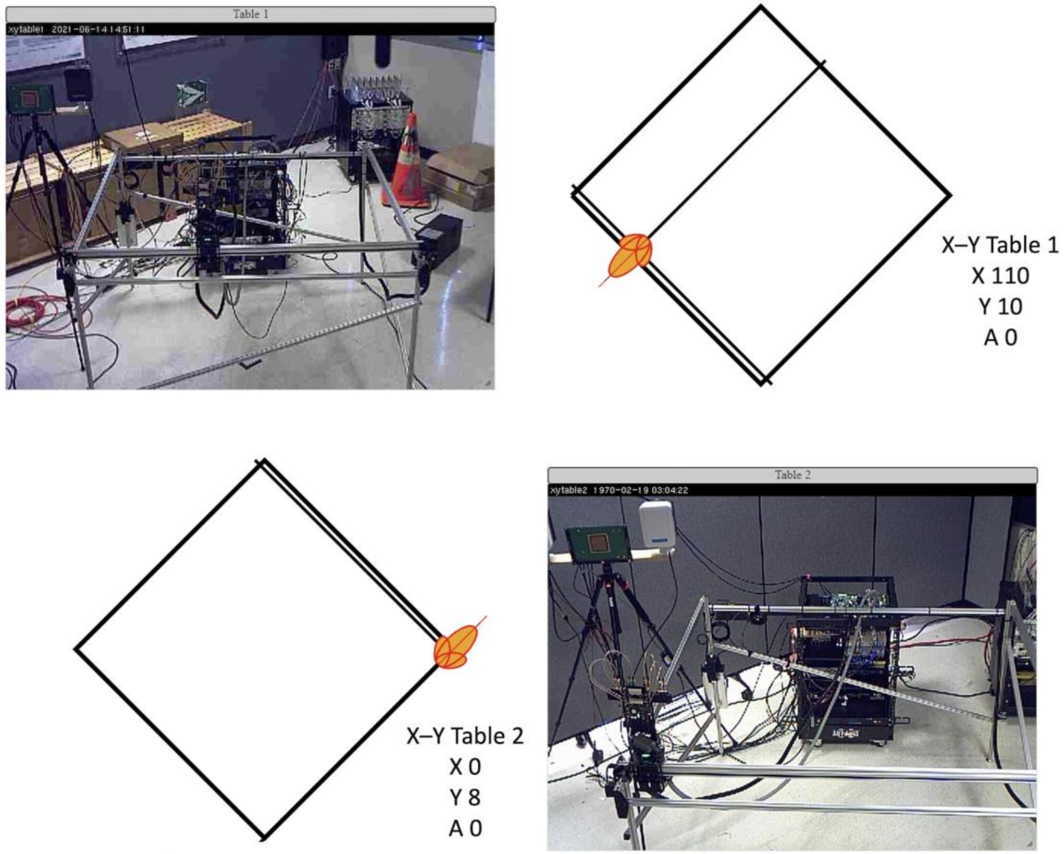
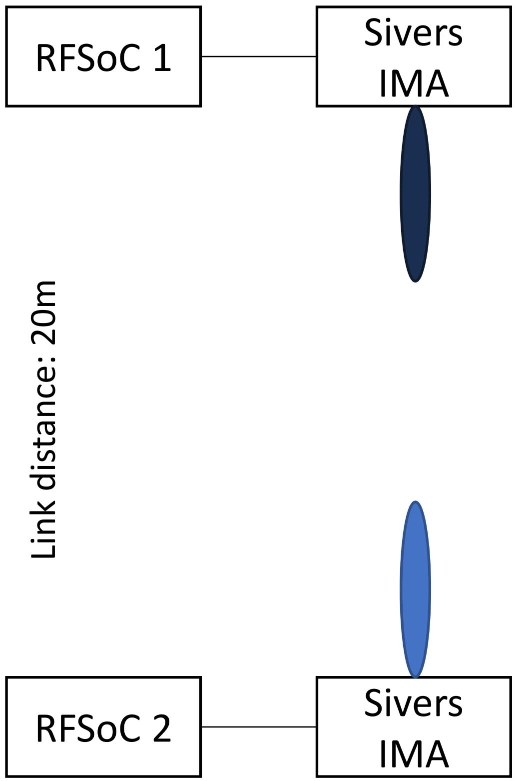


Fig: Web interface for controlling the X-Y table [2]

[2] T. Chen, et al. "Programmable and open-access millimeter-wave radios in the PAWR COSMOS testbed." in Proc. ACM WiNTECH@MobiCom'21.

Channel Sounder



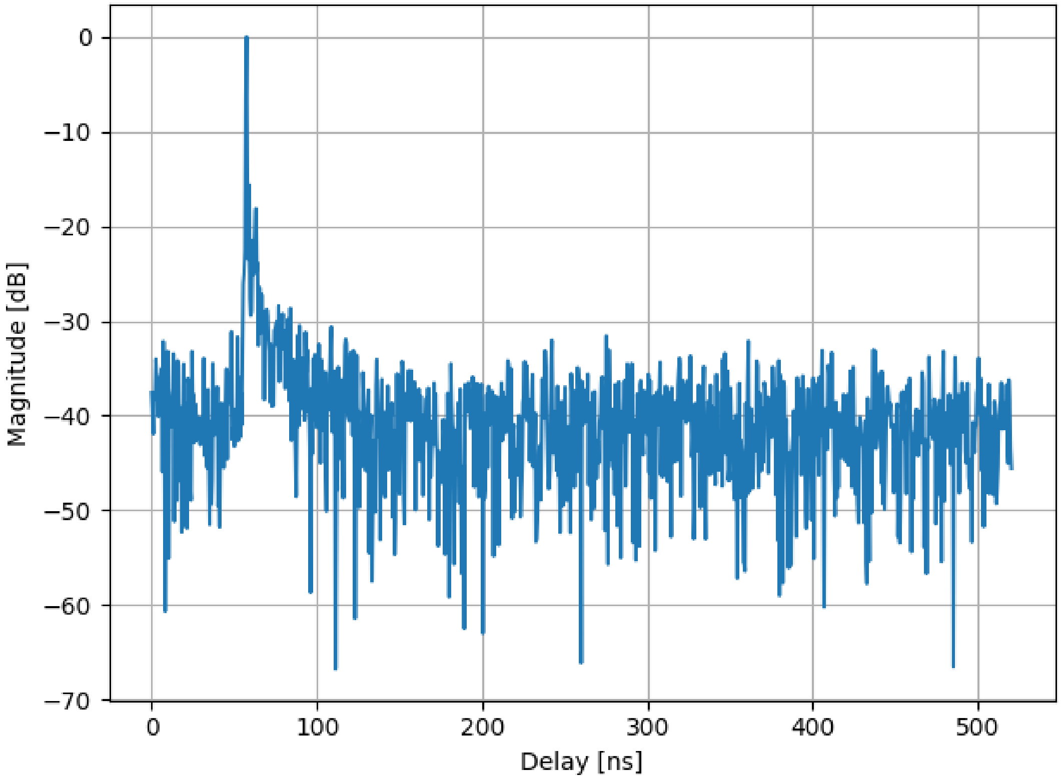


Fig: Example of a channel impulse response [3]

[10] P. Skrimponis, “Experimentation with Advanced SDR at NYU WIRELESS,“. https://github.com/nyu-wireless/mmwsdr, https://github.com/nyu-wireless/mmwsdr

Channel Measurements

- Receiver at fixed location
  - $(x_{rx} = 650, y_{rx} = 650, \theta_{rx} = 0)$
- Transmitter moves on a grid:
  - $x_{tx}, y_{tx} \in \{0, 100, 200, \dots, 1300\}$
  - $\theta_{tx} \in \{-45, -30, \dots, 45\}$
- Measure the signal-to-noise ratio (SNR)
- Receiver beamforms with $\phi_{rx} = 0^{\circ}$
- Transmitter performs beam scan
  - 17 beams uniformly distributed in $[-45, 45]$
  - $\phi_{tx} \in \{-45, -39.2, -33.4, \cdots, 39.2, 45\}$
- SNR range is from about 10 to 45 dB

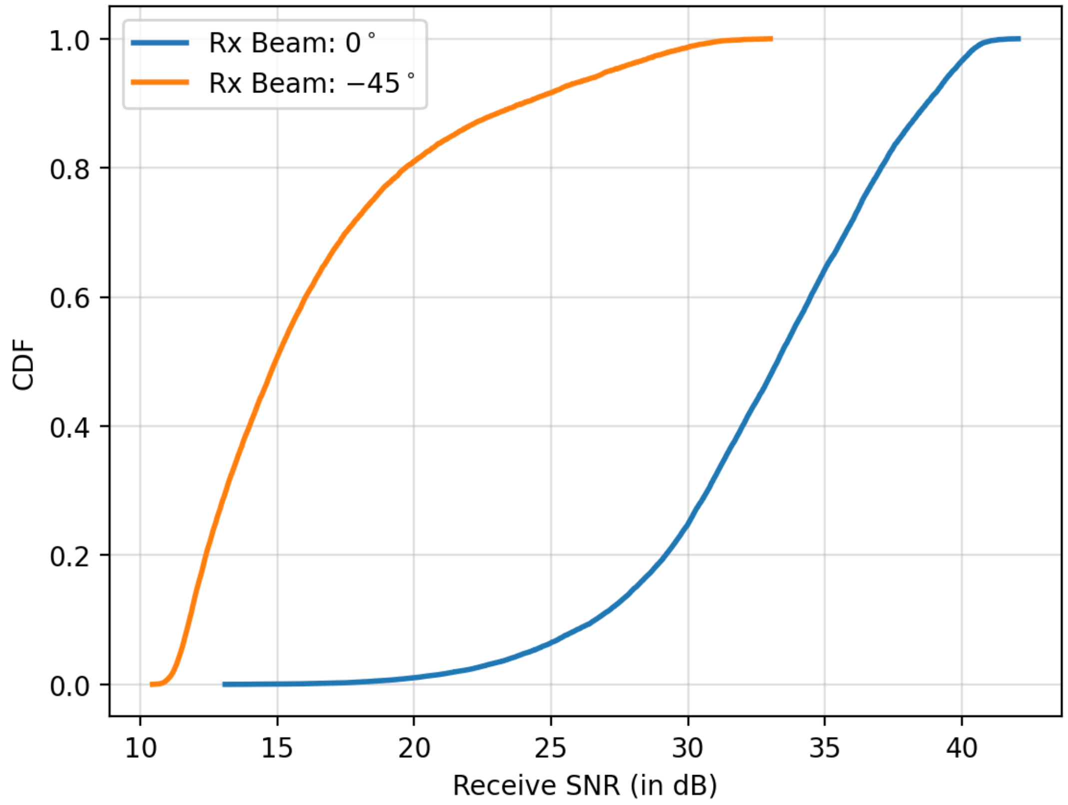


Fig: Empirical CDF of the received signal strength when receiver beamforms at 0 and -45 degrees

Mobility Model (1/2)

- Mobility of user in virtual reality (VR)
- Generate trajectories from [4], [5]
  - Redirected walking algorithm for avoiding physical collisions
- Fit trajectories on COSMOS measurements

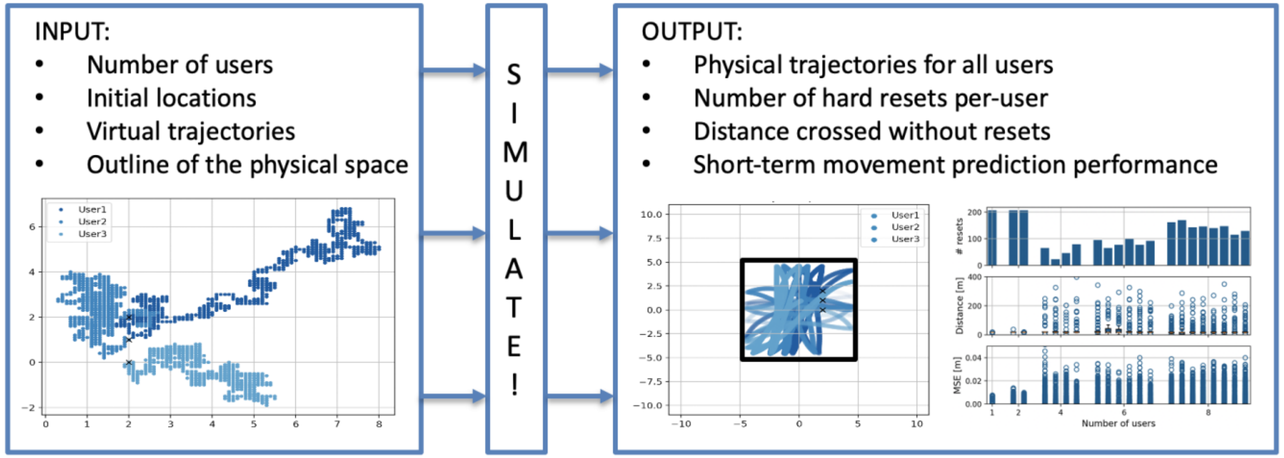


Fig. Structure of the open-source software in [6]

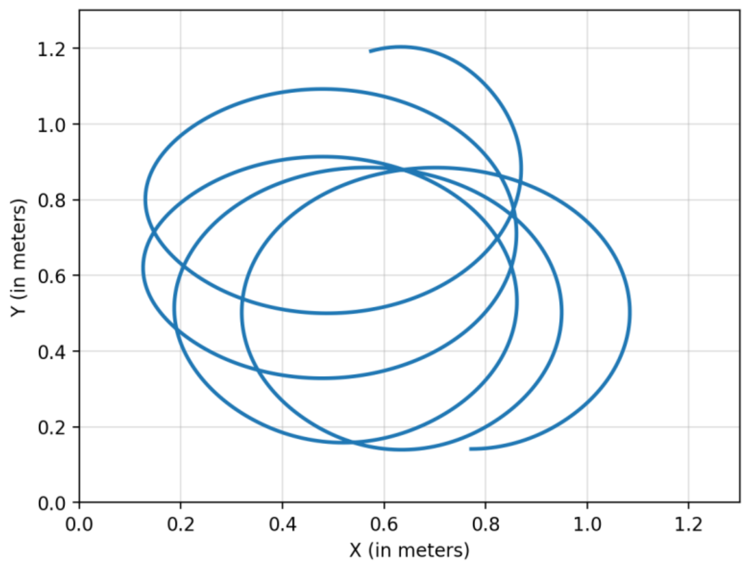
[4] E. R. Bachmann, E. Hodgson, C. Hoffbauer and J. Messinger, "Multi-User Redirected Walking and Resetting Using Artificial Potential Fields," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 5, pp. 2022-2031, May 2019, doi: 10.1109/TVCG.2019.2898764.

[5] F. Lemic, J. Struye, and J. Famaey, “User Mobility Simulator for Full-Immersive Multiuser Virtual Reality with Redirected Walking,” in Prof. of MMSys’21

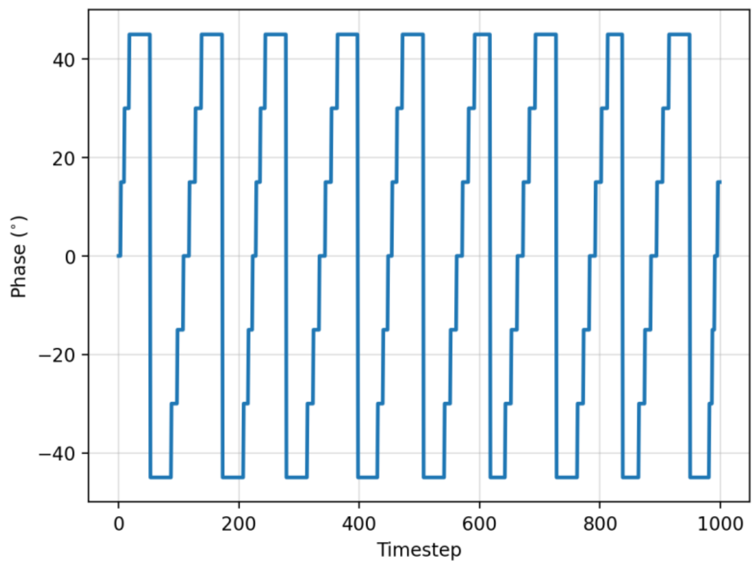
[6] IMEC IDLab, “User mobility simulator for full-immersive multiuser virtual reality with redirected walking,” https://github.com/imec-idlab/pm4vr

Mobility Model (2/2)

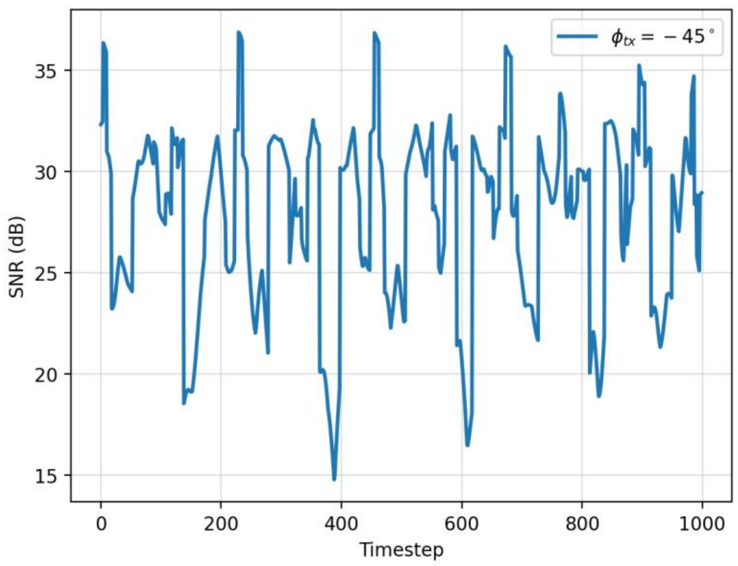
- Generate 1000 trajectories
- Capture user mobility for a total of 1000 steps
- Calculate phase based on the angle between continuous points



(a) User trajectory



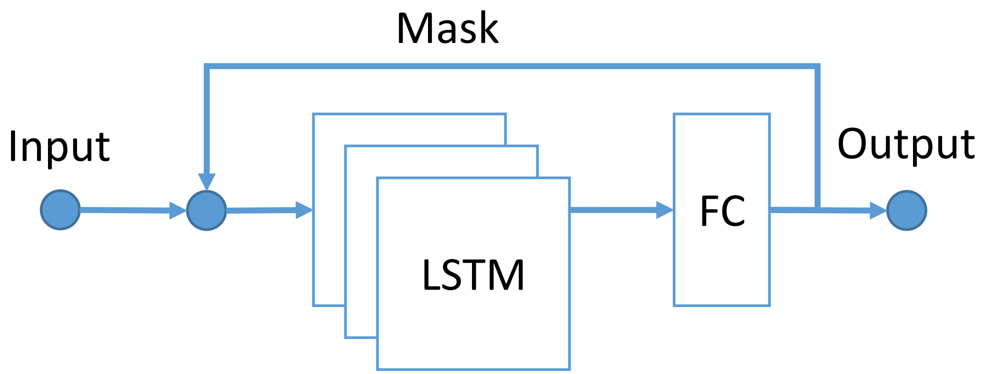
(b) User phase



(c) Signal strength for a Tx beam

AI Beam Tracking

- Develop a model using PyTorch
  - Data have spatial correlation → RNN
- Our model is based on LSTM network
  - Regression problem
  - 3 LSTM cells (128→256→128)
  - 1 Fully-connected layer (128→17)
  - Output activation: ReLU
  - Learning scheduler: Cosine Annealing
  - Optimizer: Adam
  - Learning rate: 1e-3
  - Loss: MSE
- Feedback to predict optimal mask
  - 1 for the best k beams
  - 0 for the rest



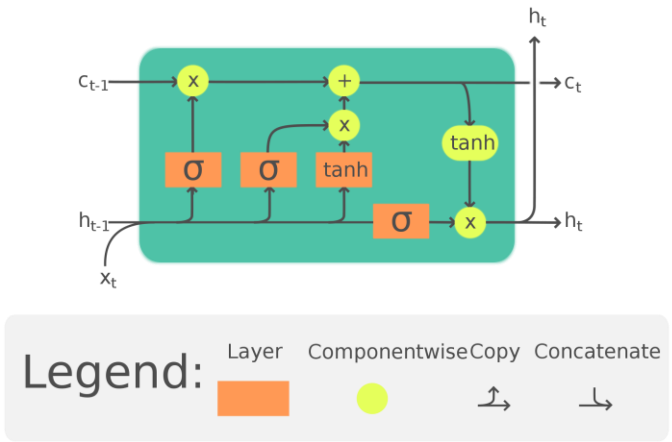


Fig: The long short-term memory (LSTM) cell [Wikipedia]

IBM Analog Hardware Acceleration Kit

- Open-source Python toolkit
- In-memory computing for AI applications
- Resistive processing units (RPUs)
  - Every element on the crossbar
  - Described by RPU configuration
- Examples
  - Single resistive
  - Unit cells
  - Compound devices

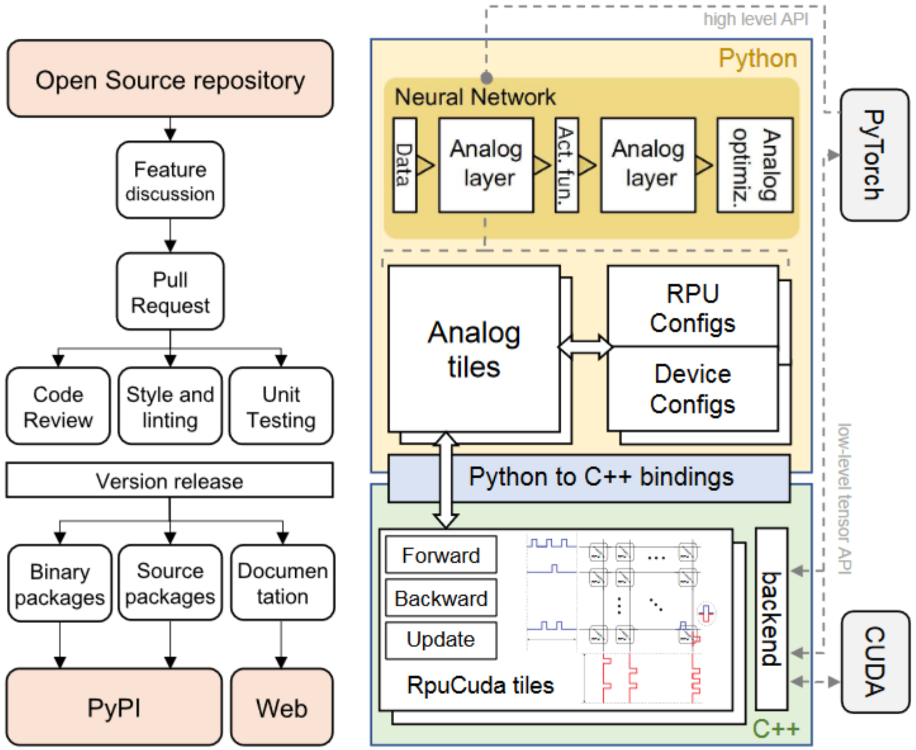


Fig: Repository and code structure of the IBM Analog Hardware Acceleration Kit [7], [8]

[7] M. J. Rasch, et. al., "A flexible and fast PyTorch toolkit for simulating training and inference on analog crossbar arrays," in Proc. IEEE AICAS'21
[8] IBM, "IBM Analog Hardware Acceleration Kit," https://github.com/IBM/aihwkit/

Hardware-Aware Training

- Our model is based on LSTM network
  - 3 AnalogLSTM cells (128→256→128)
  - 1 AnalogLinear layer (128→17)
  - Output activation: ReLU
  - Learning scheduler: Cosine Annealing
  - Optimizer: AnalogSGD
- Use feedback to predict optimal mask
  - 1 for the best k beams
  - 0 for the rest
- Train using noise sources
  - Phase change memory (PCM)
  - Conductance of PCM drifts over time
- Mitigate the effect of drift
  - Global scaling calibration procedure

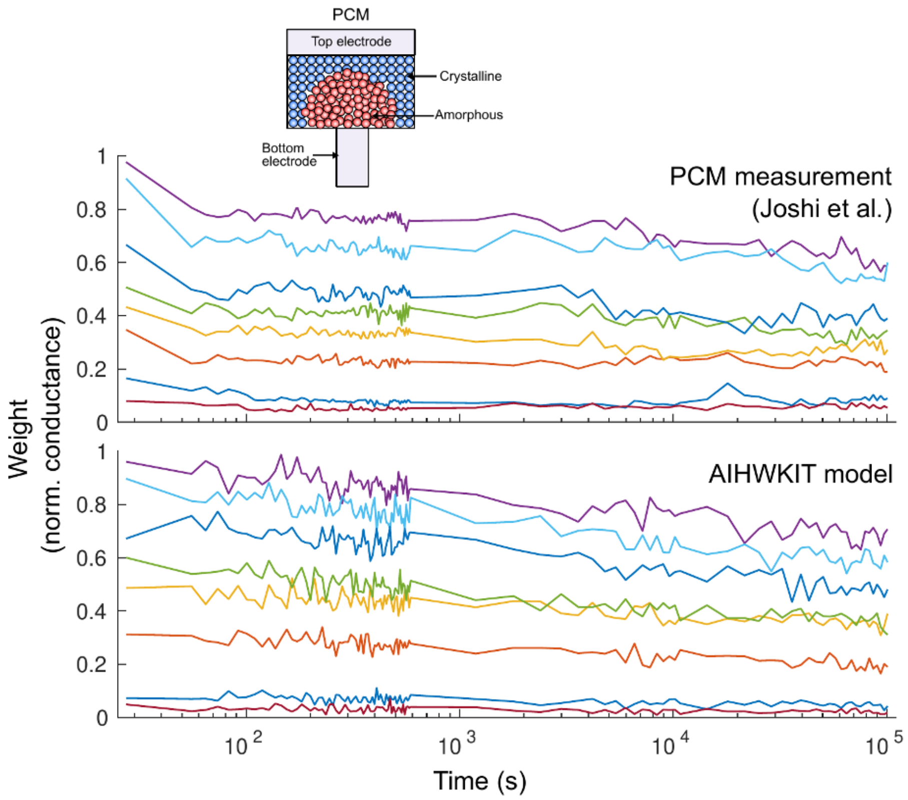
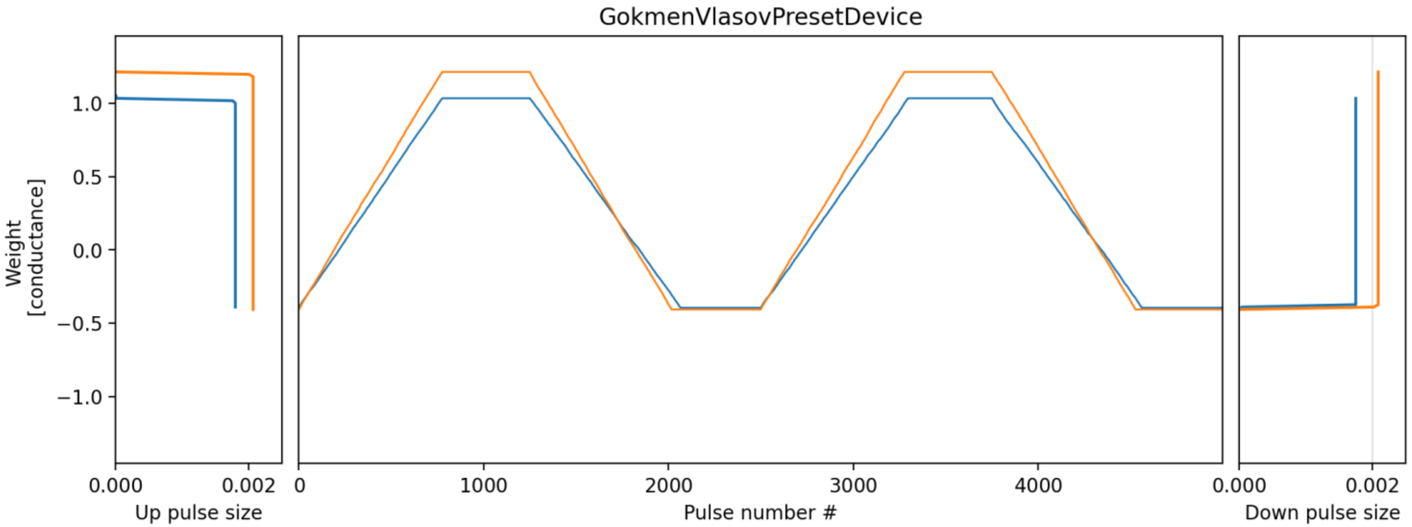


Fig: PCM model in IBM aihwkit [8] [source]

[8] IBM, "IBM Analog Hardware Acceleration Kit," https://github.com/IBM/aihwkit/

Analog Training

- Use the same model and change the RPU configuration
- Train and inference on analog device
- Use a preset device [9]
  - Calibrated on measured characteristics of real hardware devices
  - Non-ideal characteristics, noise and variability



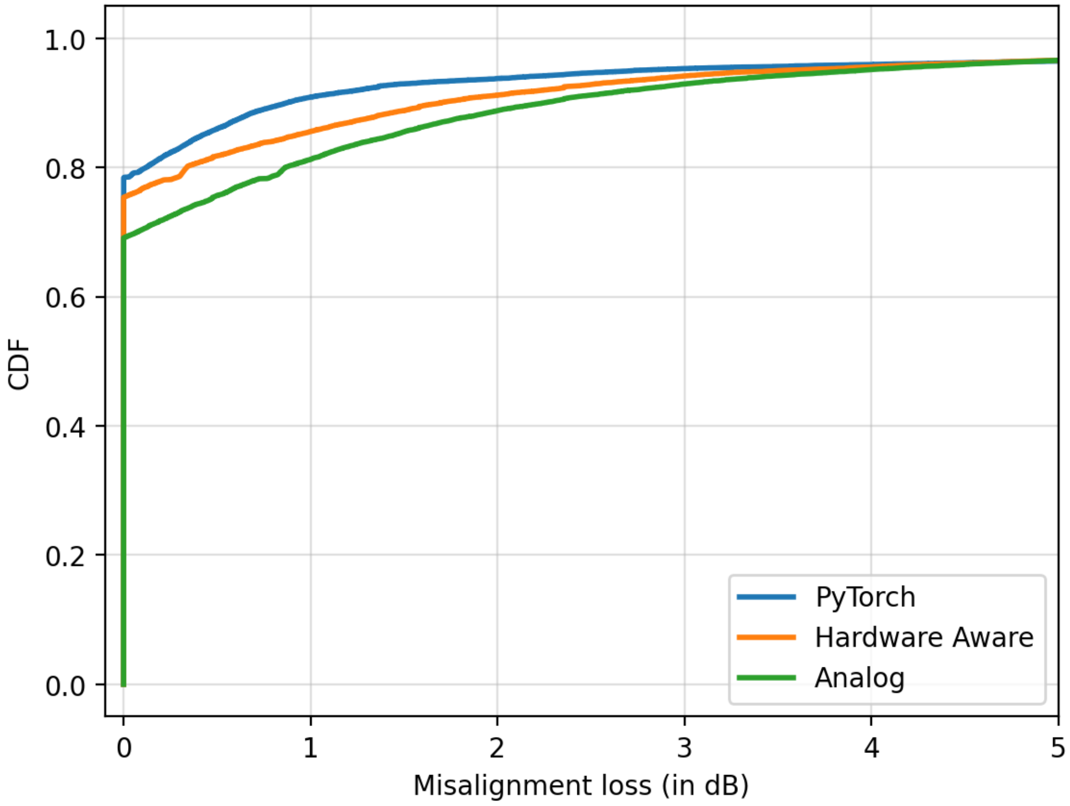
[9] T. Gokmen and V. Yurii, "Acceleration of Deep Neural Network Training with Resistive Cross-Point Devices: Design Considerations," Frontiers in Neuroscience 2016

Experimental results

- Compare results from 3 models
  - Reference
  - Hardware-aware training
  - Analog training
- Track a fraction of the beams (23.5%)
- Performance metric
  - Best possible SNR for a location: γ_{best}
  - Achievable SNR from beam prediction: γ_{pred}

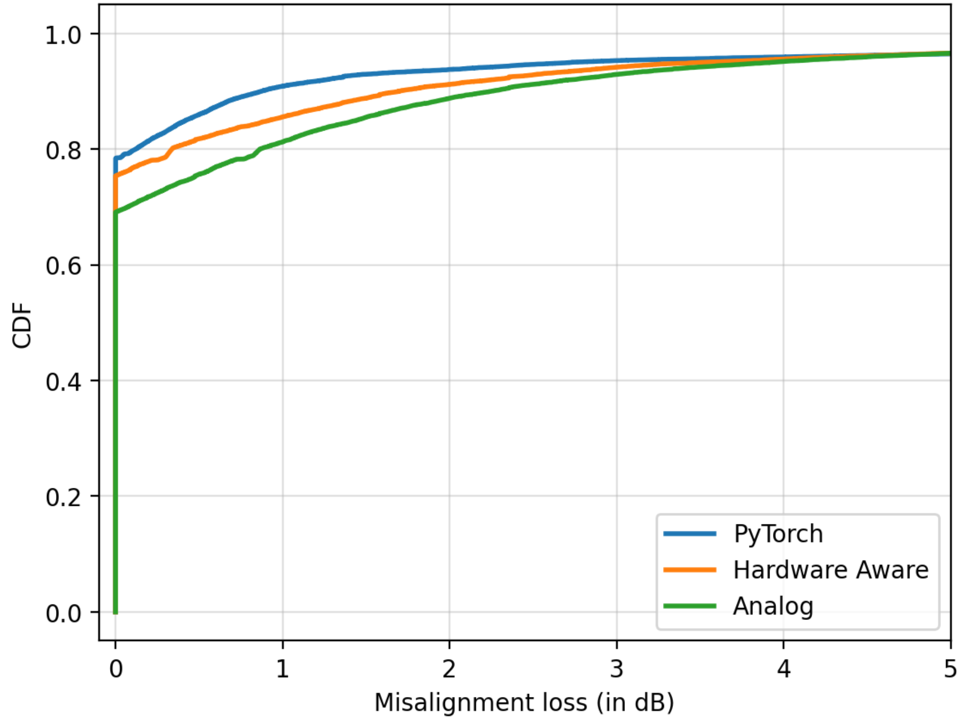
$$Loss = \gamma_{best} - \gamma_{pred}$$

- Misalignment loss is <2dB 90% of the time



Conclusions

- Technical challenges
  - Use shared cloud resources to generate dataset
  - Utilize open-source software to generate user trajectories
  - Use a framework for analog AI training and inference
  - Find the right number of beams to track
- Develop
  - Reference solution using PyTorch
  - Analog AI based solutions using aihwkit
- Track only a **small fraction** of total available beams
  - **Reduce power and time overhead ☺**
- Misalignment loss <2 dB 90% of the time



https://github.com/skrimpon/mmw-beamtrack

https://github.com/skrimpon/mmw-beamtrack

Thank you!

Any questions?

Panagiotis Skrimponis
Ph.D. Candidate at the Electrical and Computer Engineering, New York University
ps3857@nyu.edu