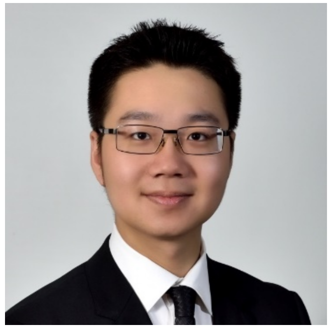
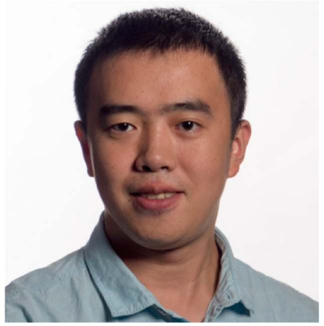
High-throughput Computation of Shannon Mutual Information on Chip

Peter Zhi Xuan Li*, Zhengdong Zhang*, Sertac Karaman, Vivienne Sze

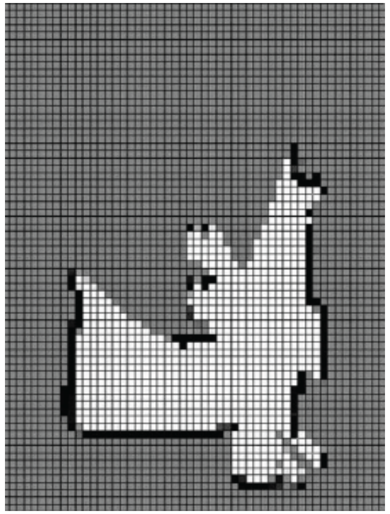
Massachusetts Institute of Technology



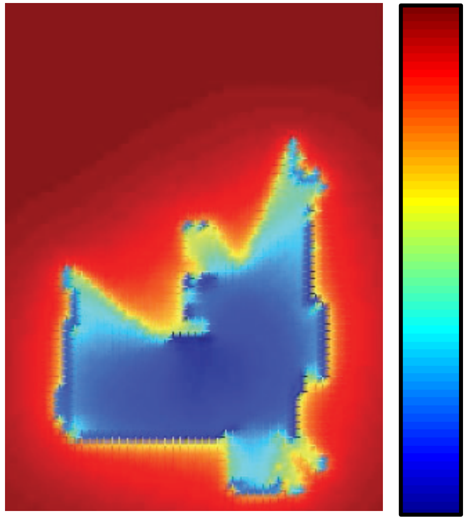


Where to Go Next: Planning and Mapping

- **Exploration:** decide where to go by computing Shannon Mutual Information (**MI**)



Occupancy grid map, M

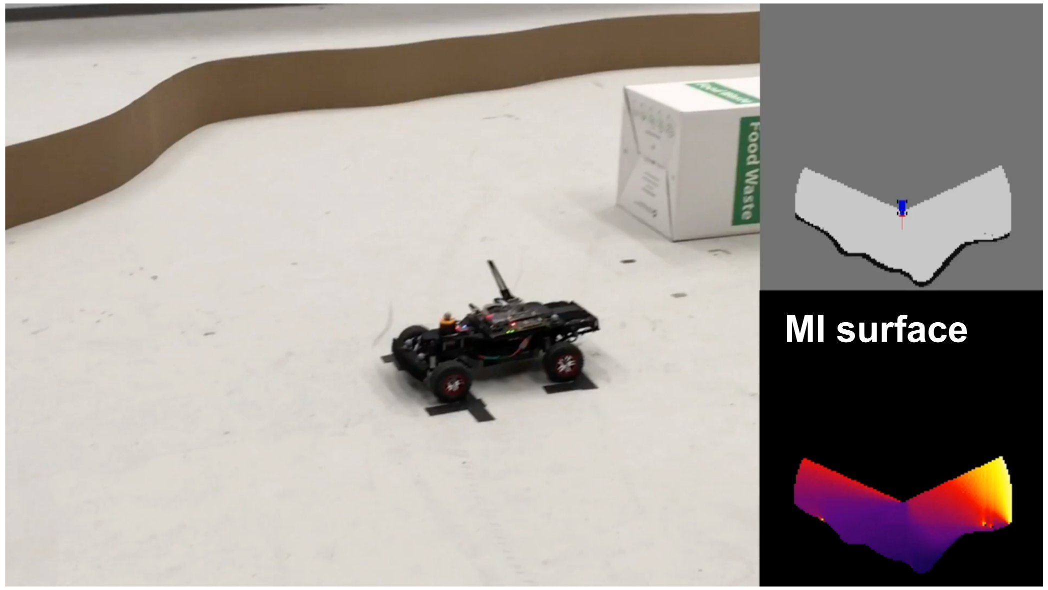


Mutual information map, $I(M;Z)$

$$\underbrace{H(M|Z)}_{\text{Perspective updated map entropy}} = \underbrace{H(M)}_{\text{Current map entropy}} - \underbrace{\color{red}{I(M;Z)}}_{\color{red}{\text{Mutual information}}}$$

Autonomous exploration with a mini racecar using motion capture for localization

Occupancy map with planned path

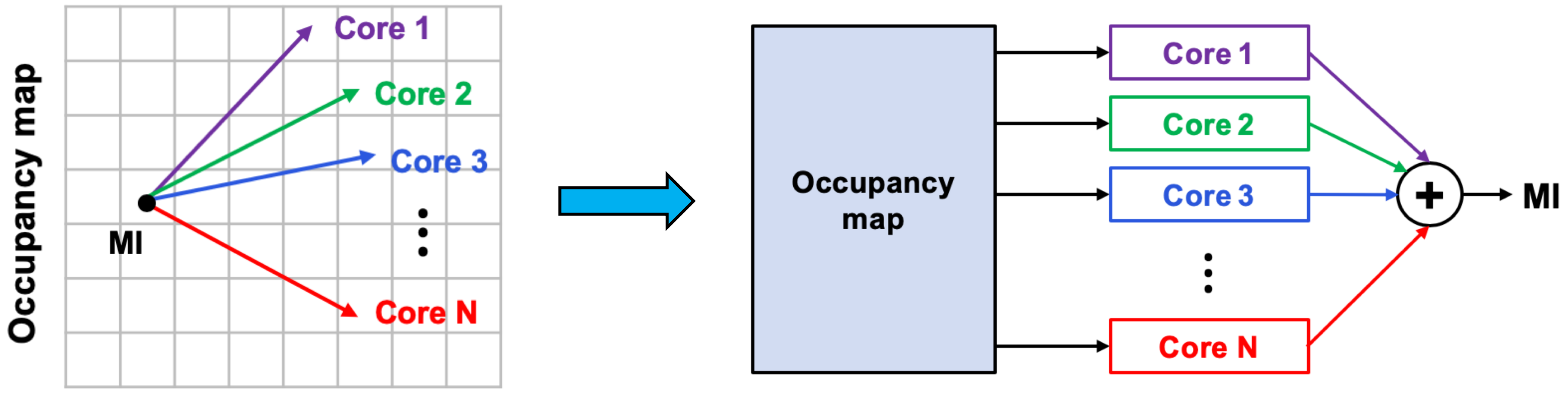


Computing Shannon MI at **more locations** allows for more optimal selection of the next scan location for mapping

Hardware Design Challenge: Data Delivery to MI Cores

- Computing Shannon MI for multiple sensor beams is **extremely parallel,** which requires a compute hardware with a **multi-port memory architecture**

Process beams in parallel with multiple cores using a multi-port memory to store the occupancy map



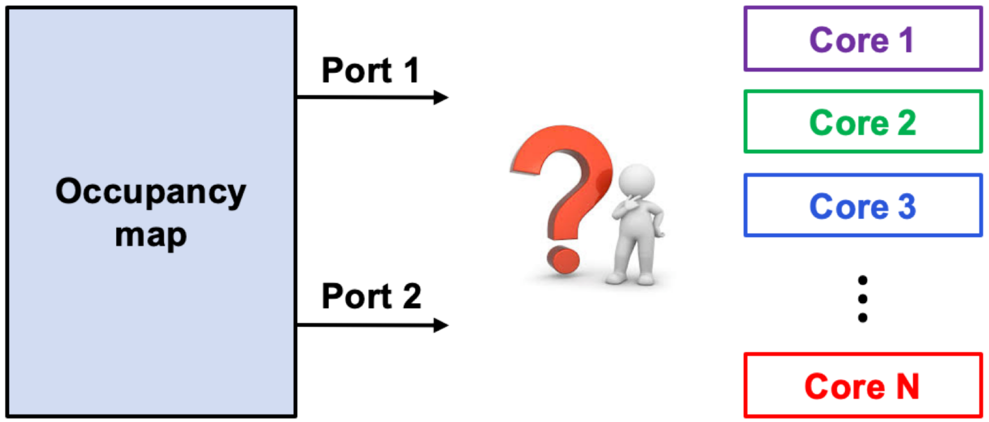
Parallelism and high system throughput require each beam to use **its own memory read port** to **independently access the map**

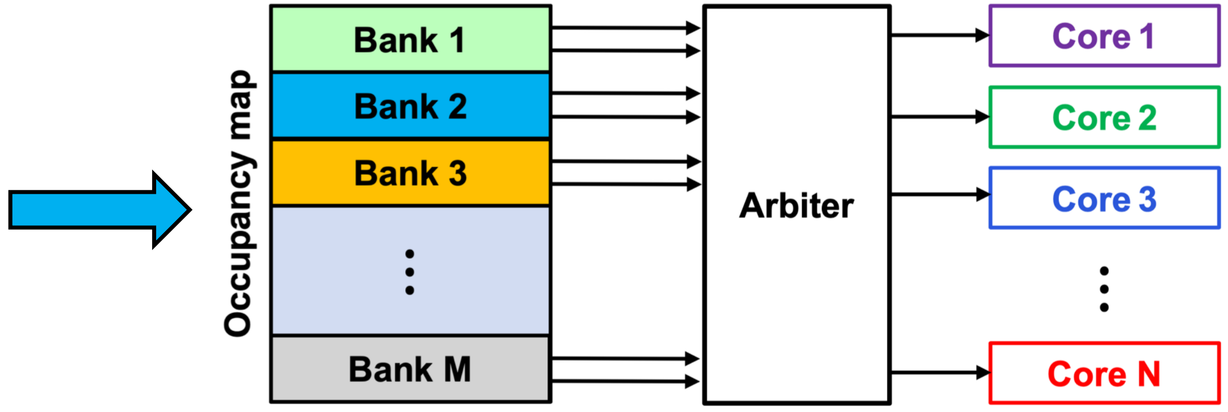
Hardware Design Challenge: Data Delivery to MI Cores

- Data delivery, specifically **memory bandwidth** (not compute), limits the throughput

A standard, low power SRAM is limited to **two ports!**

Partition the occupancy map into **several memory banks** to increase memory bandwidth



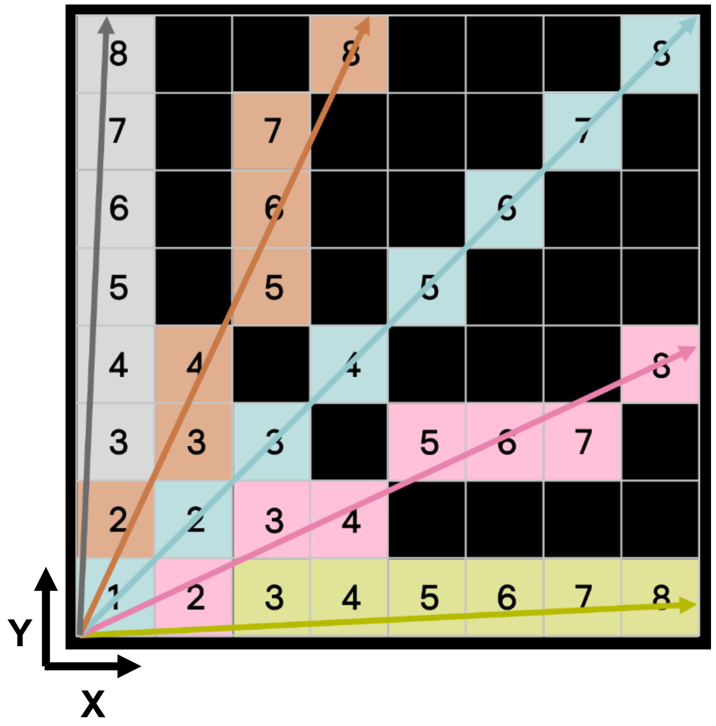


Specialized **banking pattern** required to optimize data delivery to the cores

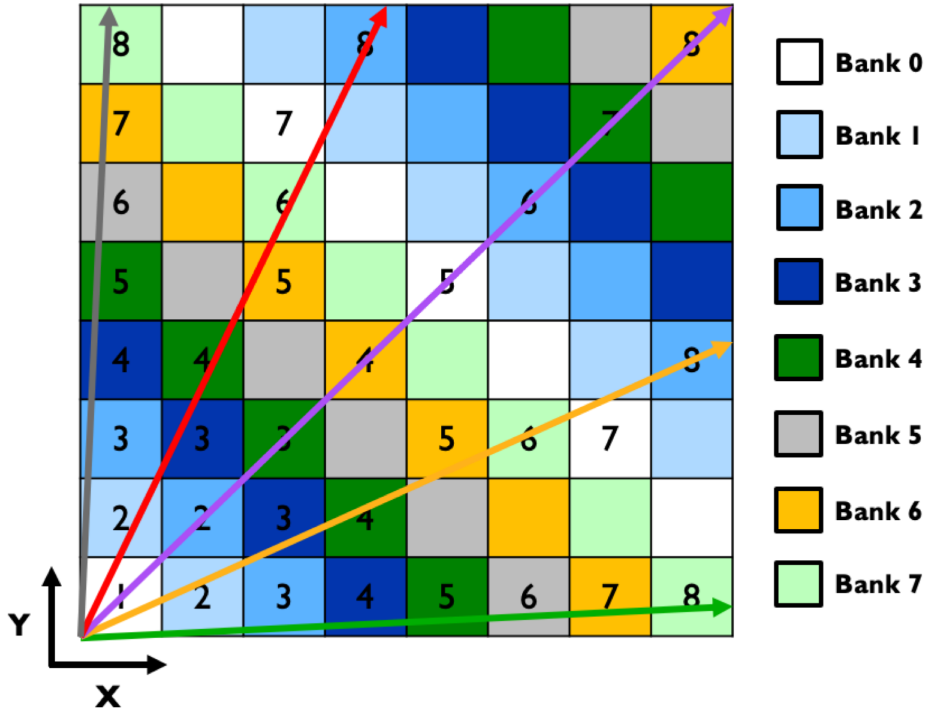
Specialized Memory Architecture

- **Goal:** Maximize data delivery bandwidth to computation cores

Memory access pattern at every cycle

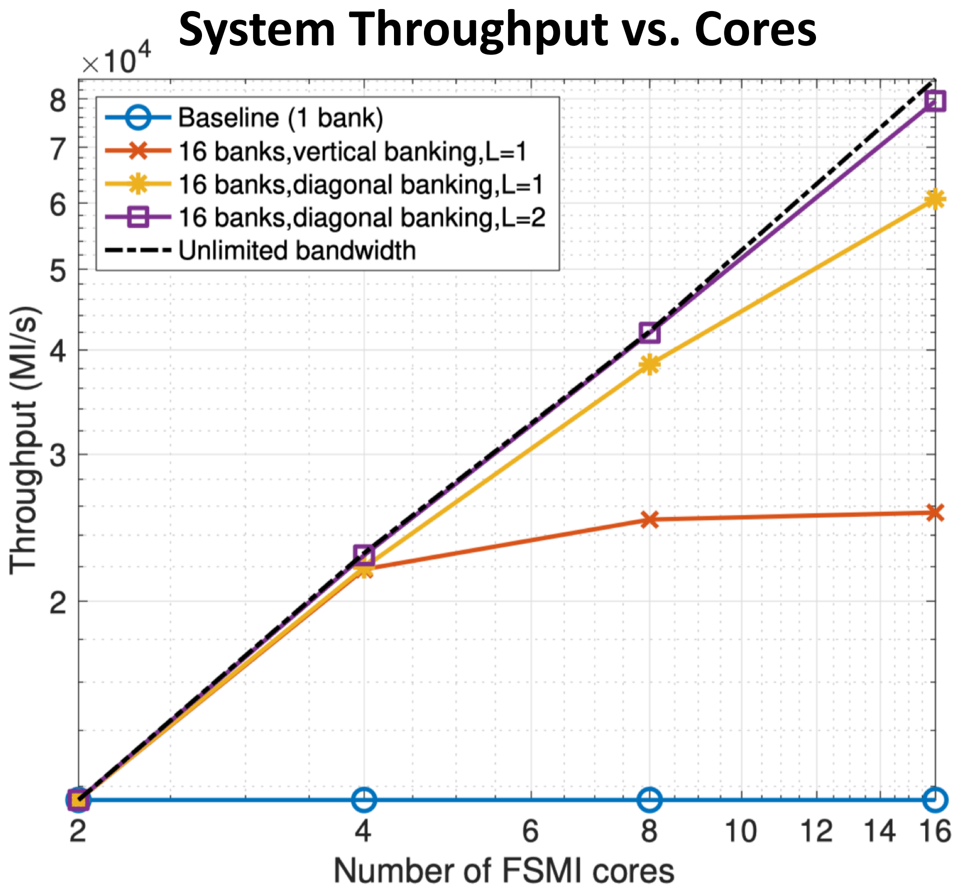


Diagonally partition occupancy map



- **Final design:** diagonal partitioned occupancy grid map of size 512x512 into **16 dual-port memory banks** provides enough bandwidth for **16 cores.**

Result



Theoretical limit (dotted black line):
throughput **increases linearly** with cores

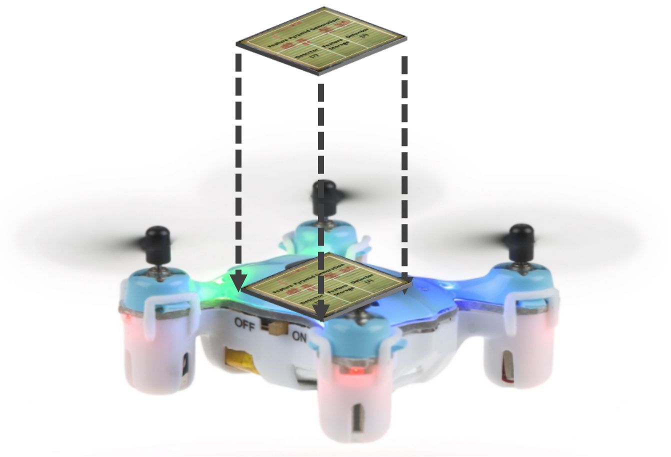
Baseline (blue line):
throughput is memory **bandwidth limited**

Proposed system (purple line):
throughput **within 94%** of theoretical limit

- **System throughput:** MI computation for the **entire map** with **200x200 grids at 2Hz** (**>100x faster** than single-threaded Intel Xeon CPU)
- **System power:** **2W** on Xilinx FPGA, which is **10x less** than single-threaded Intel Xeon CPU

Summary

- Having a parallelizable algorithm is **not** a sufficient condition for high-throughput computation on hardware
- Throughput of the multicore hardware is also dictated by its **memory architecture** and **data delivery** method to the cores
- **Near real-time computation** of Shannon MI for an **entire map** with **low power consumption**



Journal paper with proofs for the optimality of the memory architecture and an ASIC implementation coming soon!