ROG: A High Performance and Robust Distributed Training System for Robotic IoT

Xiuxian Guan, Zekai Sun, Shengliang Deng, Xusheng Chen, Shixiong Zhao, Zongyuan Zhang, Tianyang Duan, Yuexuan Wang, Chenshu Wu, Yong Cui, Libo Zhang, Yanjun Wu, Rui Wang, Heming Cui

The University of Hong Kong

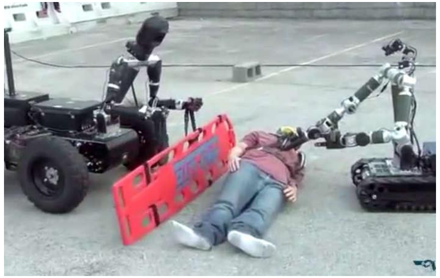
Tsinghua University

Institute of Software, Chinese Academy of Science

Southern University of Science and Technology

Motivation

- ➢ The limited computation resource on robots motivates more cooperative robotic tasks deployed over a robot team.
  - ○ Machine learning (ML) models in complex environments.
    - ■ e.g., objective recognition models, action control models



rescue

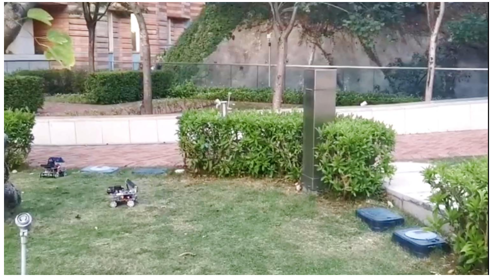
disaster response

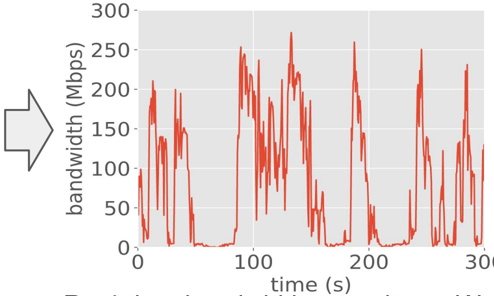
field exploration

- ➢ ML models typically require **real-time training** to adapt to **changing environments.**
  - ○ e.g., from sunny to foggy
- ➢ Lack stable Internet access to a cloud data center.
  - ○ The robots connect via the Internet of Things of robots (robotic IoT networks), which are **wireless**.
- ➢ Cooperative robotic tasks require **distributed training** among the robot team in the **robotic IoT networks** to accelerate the real-time training.

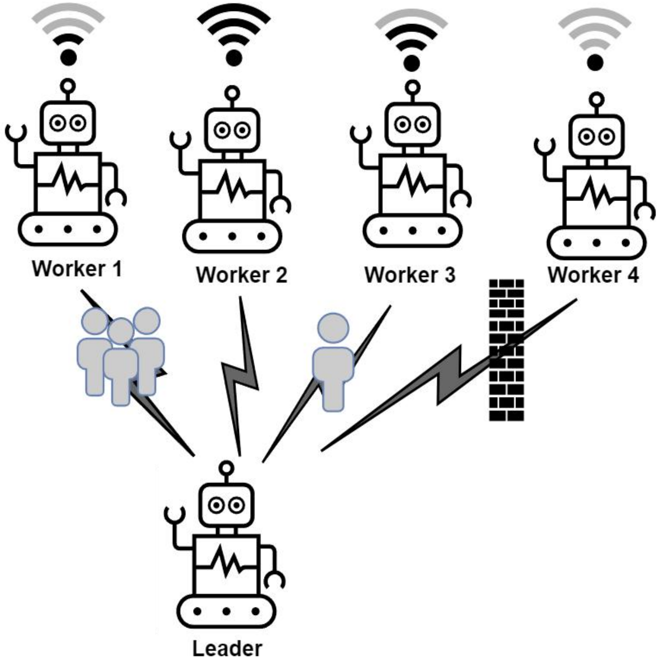
Challenges of Distributed Training for Robotic IoT

- Bandwidth capacity of Robotic IoT networks (e.g., Wi-Fi, 5G, WiMAX) is typically **unstable**.
  - Random, frequent and sharp degradation
  - Movement of the devices; occlusion from obstacles (e.g., crowds, walls)
- **Straggler effect** in distributed training in robotic IoT
  - Distributed Training requires periodical synchronizations.
  - Such bandwidth capacity degradation in wireless networks causes frequent stall (e.g., idle for 50% time of the training time).
  - **Stall significantly wastes energy** (e.g., 35%).





Real-time bandwidth capacity on Worker 4



	computation	communication	stall
Power (W)	13.35	4.25	4.04

Table III: Power (Watt) in different states.

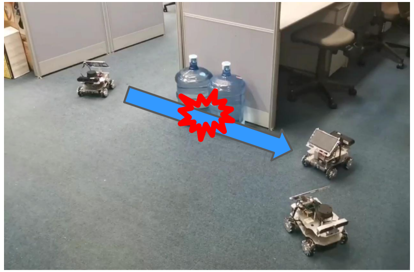
Rethinking Distributed Training for Robotic IoT

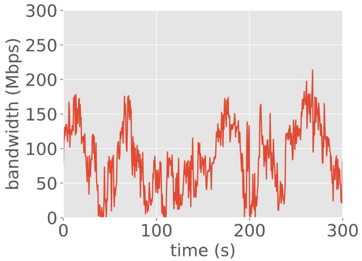
- Existing distributed training paradigms can **not** handle straggler effect.
  - **Stale Synchronous Parallel (SSP)**:
    - Only **high** staleness threshold can handle random, frequent and sharp bandwidth fluctuations.
    - Downgrade the statistical efficiency.
      - i.e., the training accuracy gain per training iteration
  - **Scheduling-based SSP** (e.g., FLOWN [TWC '21]):
    - Bandwidth fluctuations make the pre-scheduled transmission mismatch with the actual bandwidth.
    - Downgrade the training throughput.
      - i.e., the training iterations in unit time
- An **ideal** distributed training system for robotic IoT should optimize towards the following 2 trade-off goals (**2Gs**):
  - **High training throughput (G1)**: The training process should avoid stall caused by the straggler effect.
  - **High statistical efficiency (G2)**: The training system should not sacrifice the statistical efficiency during handling the straggler effect.
- Better trade off between 2Gs can help distributed training to achieve high training accuracy and avoid wasting energy.

Why Existing Work Can Not Achieve the Best Trade-off between 2Gs

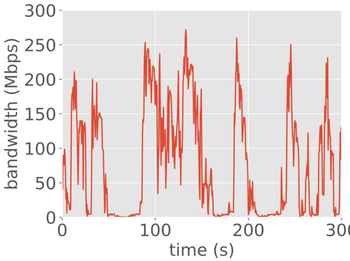
- **Key reason: Existing coarse-grained training synchronization in the granularity of a whole model (model granularity) cannot adapt to random, frequent and sharp bandwidth fluctuation in robotic IoT.**
  - The recorded bandwidth fluctuation in wireless networks caused by movement and obstacles
  - Model granularity causes imbalanced transmission time under such bandwidth fluctuation.

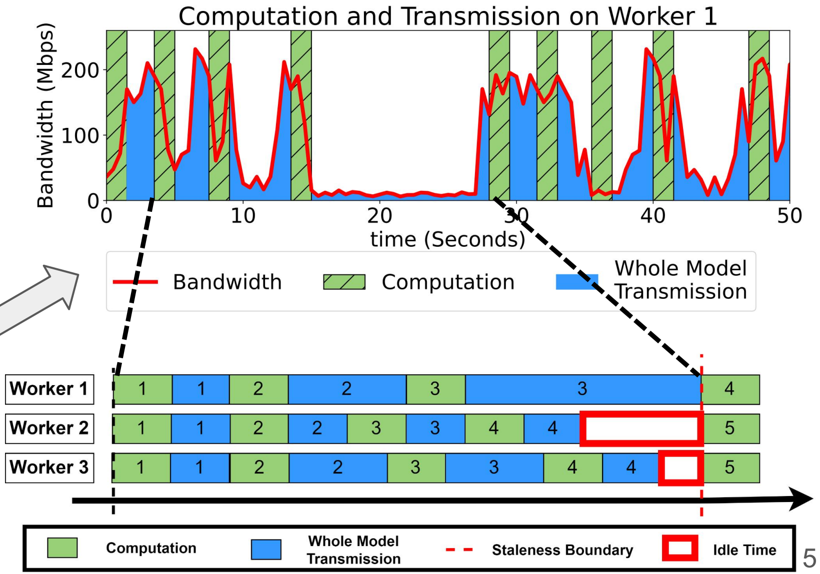
Indoors （office）





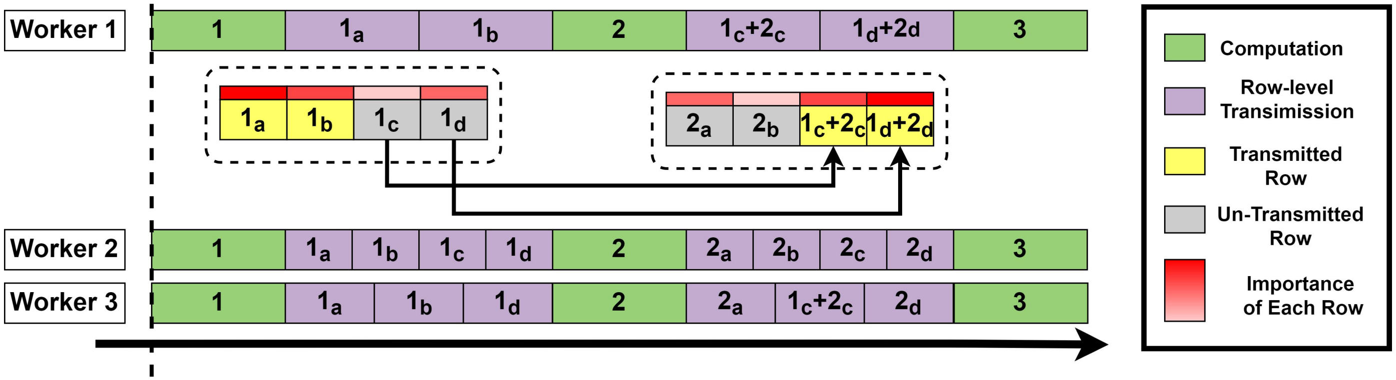
Outdoors (garden)





From Model Granularity to Row Granularity

- We propose a new **row granularity** synchronization method: break up the synchronization granularity of gradients into matrix rows.
  - Element granularity incurs **high management cost;** layer granularity (matrixes) still suffers from **low transmission flexibility.**
  - Row granularity enables more flexibility in scheduling:
    - Adaptively adjust the transmission volume to balance transmission time
    - Transmit more important gradient rows first to accelerate model convergence



ROG: ROw-Granulated, High Performance and Robust

- In this paper, we present **ROG**, a **RO**w-**G**ranulated, high performance and robust wireless distributed training system optimized for real-world robotic IoT networks.
  - How to guarantee convergence when synchronizing gradients on row granularity?
    - Row Synchronous Parallel (RSP)
      - Same convergence guarantee as SSP
  - How to schedule the transmission of rows to achieve **2Gs**？
    - Schedule the transmission of each row adaptively to the fluctuating bandwidth (**G1**)
      - Let the straggler transmit fewer rows than others to avoid straggler effect
    - Transmit gradient rows with more contribution to training accuracy first (**G2**)

Implementation and Evaluation

- **Implementation Details:**
  - Implemented as a PyTorch optimizer, easy to use and extensible to any imperative Deep Learning frameworks.
- **Baseline Distributed Training System**: BSP, SSP with high staleness threshold (SSP 20), SSP with low staleness threshold (SSP 4), FLOWN.
- **Evaluation settings**
  - **Real-world robots:** four-wheel robots with NVIDIA Jetson Xavier NX.
  - **Cooperative online learning robotic applications**: domain adaptation and implicit mapping
  - **Real-world wireless networks and environments:** **indoors** (laboratory with desks and separators); **outdoors** (campus garden with trees and bushes).
  - Minimize the communication volume with **gradient compression** [NIPS '19]

Evaluation Questions

- ➢ How does ROG benefit real-world robotic applications in terms of training accuracy and power consumption?
- ➢ How does ROG handle the instability of wireless networks?
- ➢ How sensitive is ROG to different batch sizes, different numbers of devices, and different thresholds?
- ➢ What are the limitations and potentials of ROG?

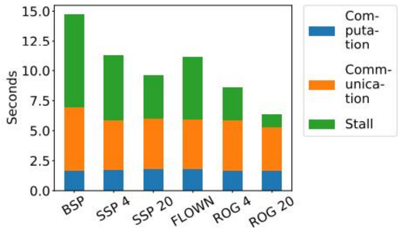
End-to-End Performance

- ➢ **ROG achieves high training accuracy.** ROG achieved a **4.9%~6.5%** accuracy gain over the baselines after training for 60 minutes.
- ➢ **ROG is energy-efficient.** When the training model reached a same high accuracy, compared with the baselines, ROG reduced battery energy consumption by **20.4%~50.7%.**
- ➢ After training for 60 minutes, compared with baselines, ROG achieved:
  - ○ **High training throughput** (25.2%~80.4% higher than baselines, **G1**)
  - ○ **Non-degraded** statistical efficiency (**G2**)

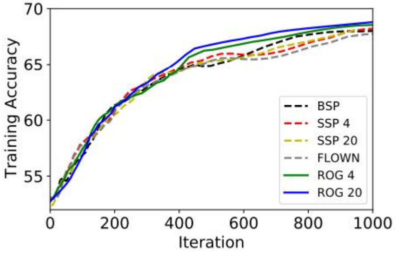
Source Image

Domain Adaptation

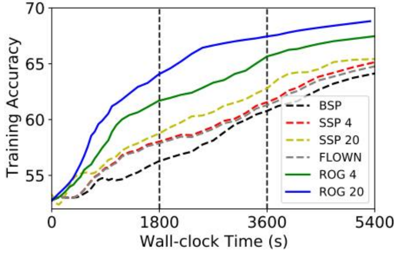
Target Image



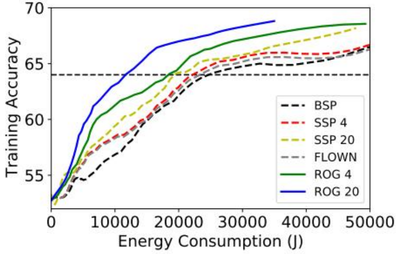
(a) Average time composition of a training iteration.



(b) Statistical efficiency.



(c) Training accuracy against wall-clock time.



(d) Energy consumption against training accuracy.

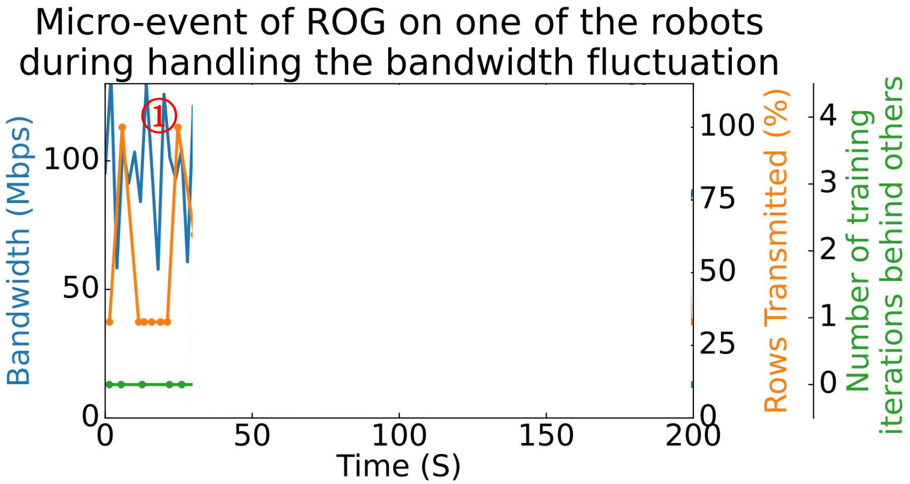
Domain Adaptation in Outdoors Scenario

How ROG Handles the straggler effect

① **Bandwidth fluctuating**: ROG adaptively adjusts the number of rows to be transmitted to avoid this robot from falling behind.

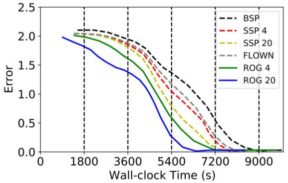
② **Bandwidth degrades**: this robot becomes a straggler and is required to transmit the minimal number of rows to lower the speed of falling behind.

③ **Bandwidth rises**: this robot quickly catches up with others because it is allowed to transmit fewer number of rows than other robots.

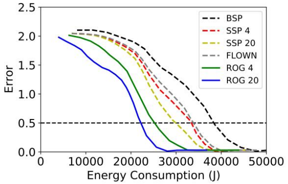


ROG’s Performance on Various Applications and Environments

Implicit Mapping

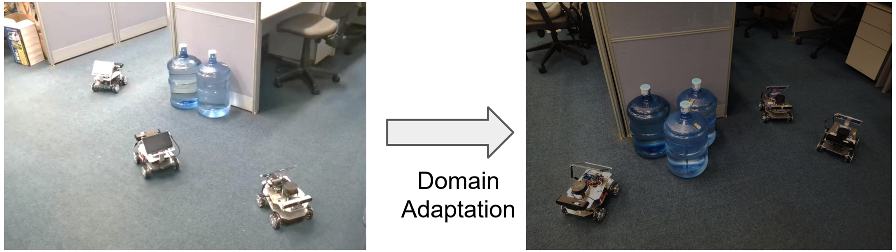


(c) Training accuracy against wall-clock time.



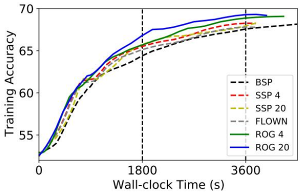
(d) Energy consumption against training accuracy.

Domain Adaptation in Indoors Scenario

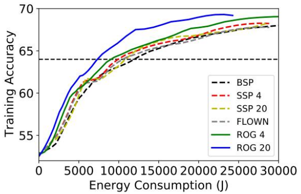


Source Image

Target Image



(c) Training accuracy against wall-clock time.



(d) Energy consumption against training accuracy.

For various real-world online learning robotic applications and real-world environments, ROG’s advantages in high training accuracy and high energy efficiency remain.

Conclusion

- In this paper, we present ROG, a ROw-Granulated distributed training system optimized for robotic IoT networks.
  - ROG breaks up the granularity of model synchronization into rows and applies adaptive scheduling to the transmission of each row, fulfilling 3Rs.
  - ROG is easy to use. It took only tens of lines of code to apply ROG to existing ML applications.
- By better trade off 2 goals for bandwidth fluctuation in wireless networks, ROG will nurture diverse ML applications deployed on mobile robots in the field.
- ROG's future work:
  - Integrate ROG with existing methods for accelerating distributed training such as pipelining communication and computation (Pipe-SGD [NIPS '18]).
- ROG's artifact is available at https://github.com/hku-systems/ROG

Thank you for listening. Questions are welcome!