ECE 455 - GPU Algorithm and System Design

Performance Analysis of Generic Approaches for Large-Scale Small-Size Matrix Multiplication

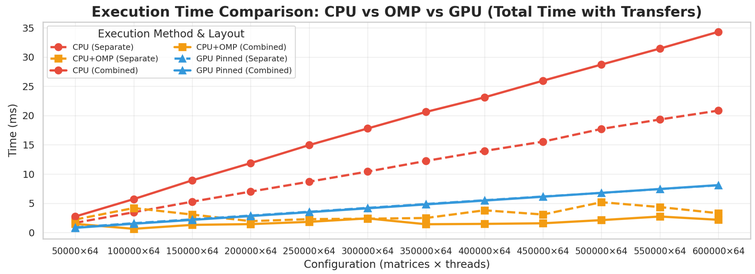
Warp Kinematics: Yucheng Huang, Wuzhen Li, Steven Li, Lynnix Zou

Advisor: Tsung-Wei Huang

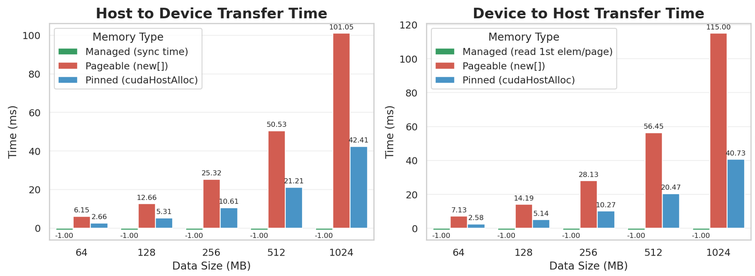
GitHub Repo QR code

Workstation[1] - Xeon Gold 6330 + 3090

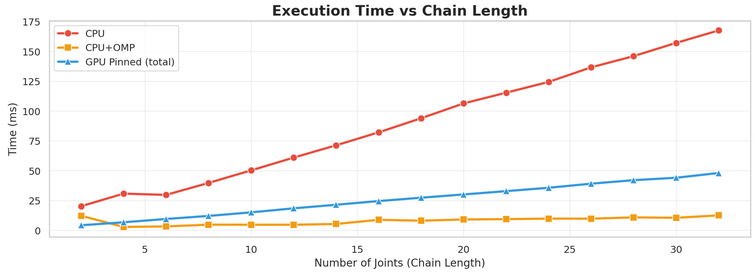
Data Structure Improvement



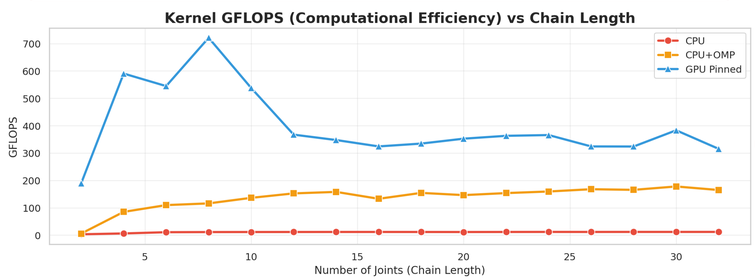
Data Transfer Improvement[4]



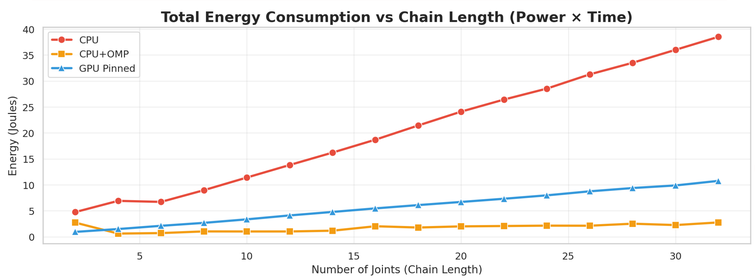
Variable Workload Thread Performance



Throughput Performance



CPU + GPU Energy Consumption



Motivations and Goal

Modern robots relies on thousands of **Forward Kinematics** per move instructions using 4x4 **matrix multiplications**

On **MCUs with limited cores,** this yields poor throughput but on **edge devices with CUDA** we get to utilize **more threads and unified memory**

Out goal is to investigate that under what certain setups and tasks, **does it worth it to compute with CUDA cores instead of CPU cores to solve Large-Scale Small-Size Matrix Multiplication** problem

Methodology and Experiments[3]

Data Layout Strategy: Pinned Memory Comparison:

Method A: Separate Memory Chunks (Per-step separation)

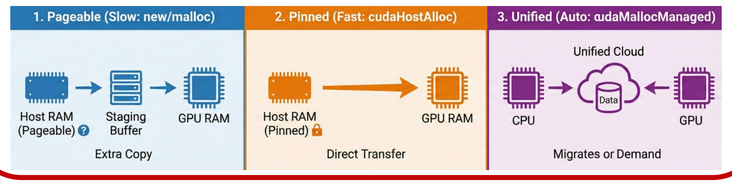
Method B: Continuous Interleaved Chunk

Key Finding:

Discrete Memory: Continuous chunks cause **poor prefetching behavior for Single Core CPU version**

Unified Memory: Separate and continuous memory chunks perform roughly the same due to **conservative prefetching**

Platform	Host to Device	Device to Host
Edge Device	3.51x	3.58x
Workstation	2.37x	2.77x



Discrete Memory:

CPU+OMP is the best solution because it has moderate level of compute power while free from PCIe data movement constraint

Unified Memory:

GPU with managed memory allocation is the best solution because it enables **zero-copy** compute between GPU and CPU

Device Energy Consumption (CPU OMP vs. GPU managed)**:**

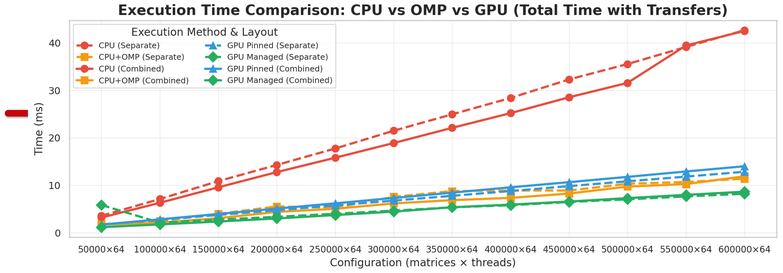
With **2.3% (0.04/1.75) the energy consumption** of workstation, Jetson can **process same amount of workload as the workstation** at a cost of **2.96x (23.25/7.85) more time**

Conclusion and Limitations

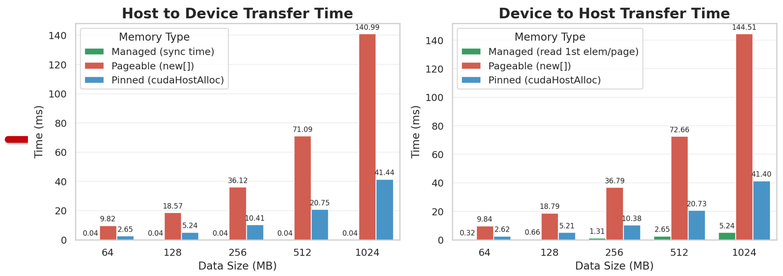
1. Large-scale small-size matrix multiplication is best using
   - **OMP on a Discrete Memory Device**
   - **CUDA with pinned memory on a Discrete Memory Device with fast data transfer**
   - **CUDA with managed memory on a Unified Memory Device**
2. This optimized kernel does not benefit from using **CUDA Stream** because invoking a stream has additional overhead but our execution time for the kernel is too short to benefit from overlapping compute and transfer

Edge Device[2] - Jetson Nano Super

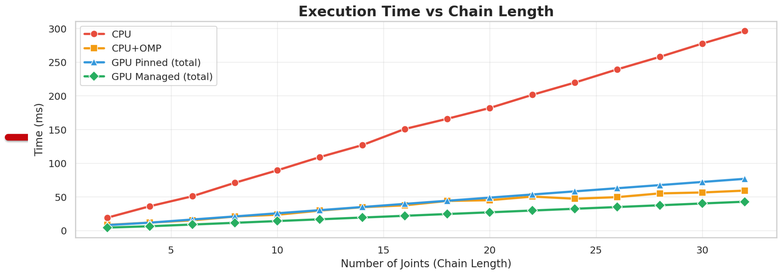
Data Structure Improvement



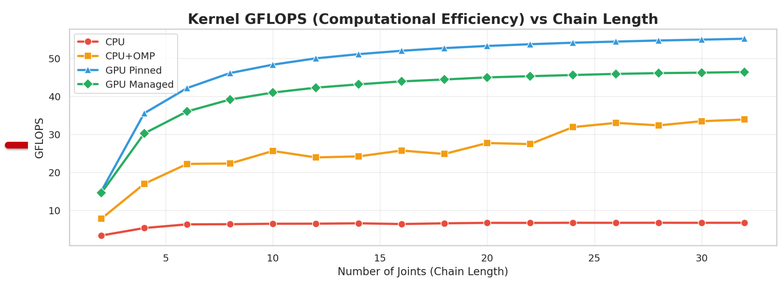
Data Transfer Improvement



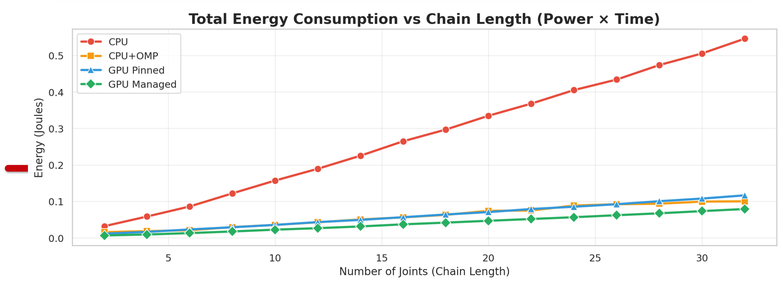
Variable Workload Thread Performance



Throughput Performance



CPU + GPU Energy Consumption



1. Workstation has a variable clock frequency for both CPU (Xeon Gold 6330, 56 cores) and GPU (RTX 3090, Ampere architecture), so it appears to have more fluctuation. 2. Jetson CPU (Arm, 6 cores) is locked at Super Mode (25W) with fixed 1.344GHz and GPU (Ampere architecture) at 918MHz frequency. 3. Experiment was conducted 10 times on each devices and taken the average of results for stability. 4. The workstation is running with PCIe 4.0 x16, so theoretical bandwidth is 32GB/s