

GPU acceleration in Lagrangian Particle Tracking using CUDA for Python and Fortran

2025-01-23

Thisal Mandula Sugathapala

Background

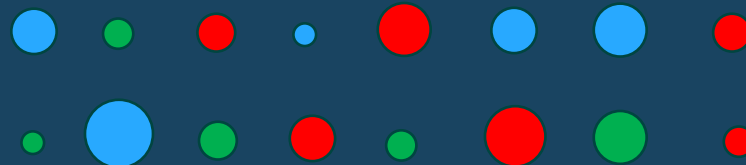
Project Overview

- Lagrangian biophysical particle tracking (LPT) code for microplastics in the ocean.
- Port the LPT code to GPU architecture using CUDA Python and CUDA Fortran.
- Perform a comprehensive comparison of the LPT model's performance across each programming language (Python and Fortran) on CPU and GPU platforms.
 - Performance evaluations on workstation with Nvidia Geforce RTX 3070.
 - Performance evaluations on Vera cluster GPUs (A100, V100, A40, T4).
 - Impact of thread block structure on model performance.
 - MPI vs CUDA.
 - Limitations of running CFD codes on GPU.

Background

Particle distribution on GPU

 Polypropene  Low Density Polyethylene  High Density Polyethylene



Background

Particle distribution on GPU



Polypropene

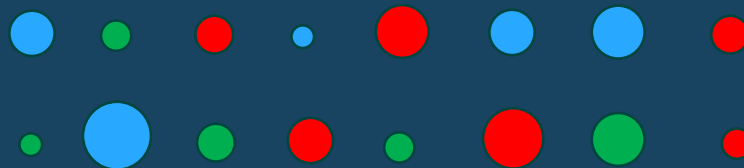
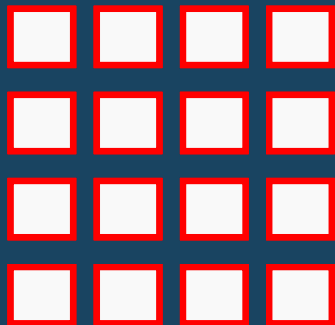


Low Density Polyethylene



High Density Polyethylene

Thread Block 1



Background

Particle distribution on GPU



Polypropene

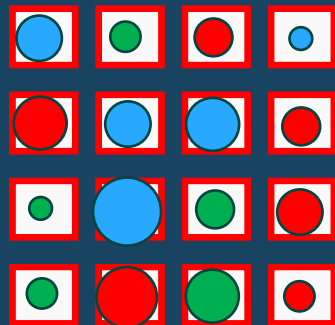


Low Density Polyethylene



High Density Polyethylene

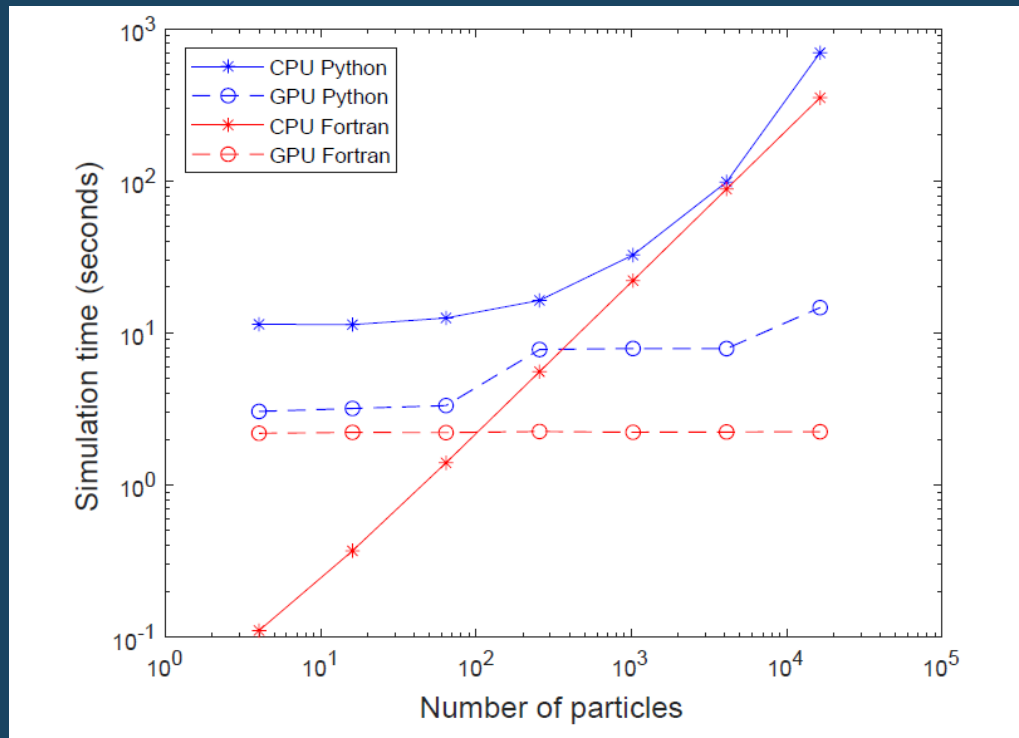
Thread Block 1



- For 128×128 particles tracked for 100 days at a time step of 100 seconds.
- Python
 - CPU – 696 seconds
 - RTX 3070 GPU – 14 seconds
 - A100 GPU – 1.8 seconds

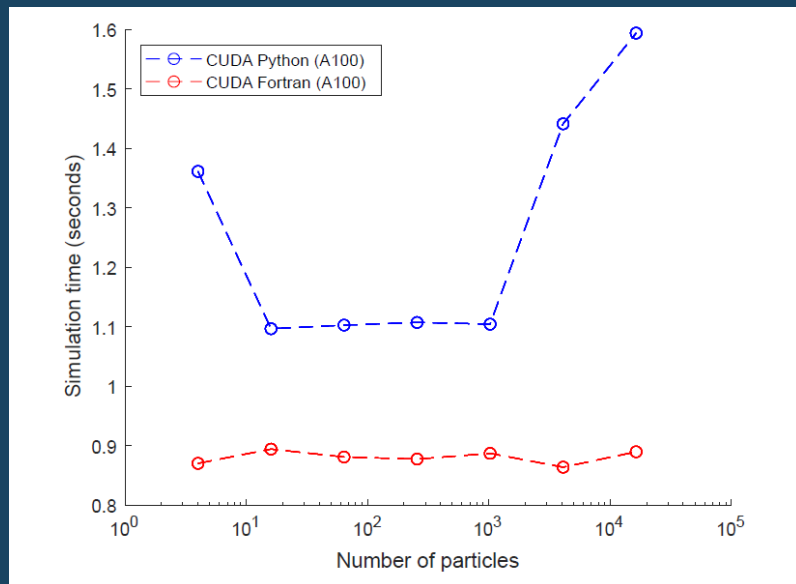
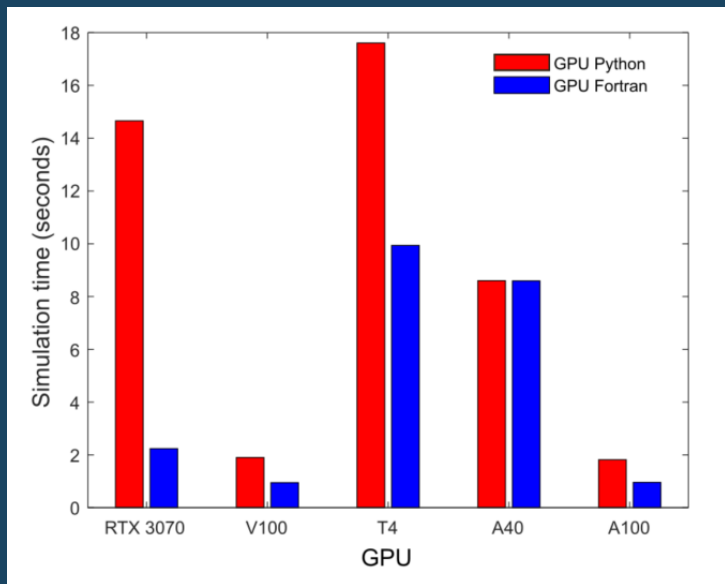
Results

Simulation Time on Nvidia RTX 3070 General Purpose GPU



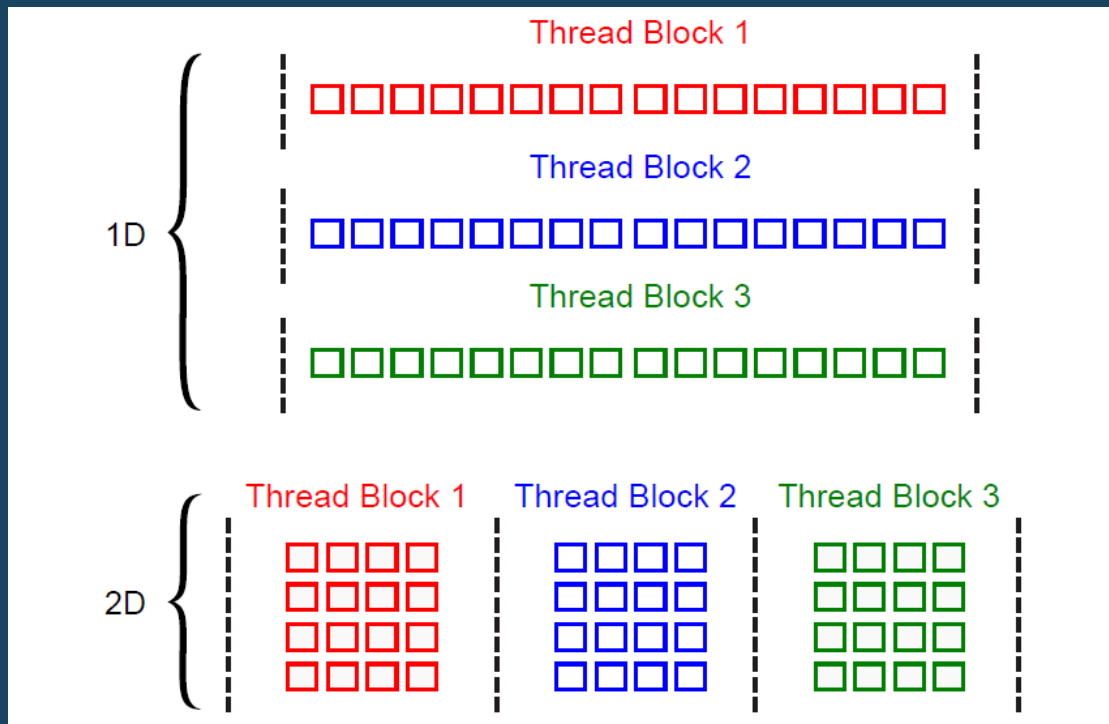
Results

CLUSTER GPU PERFORMANCE COMPARISON (16,384 particles for 100 days)



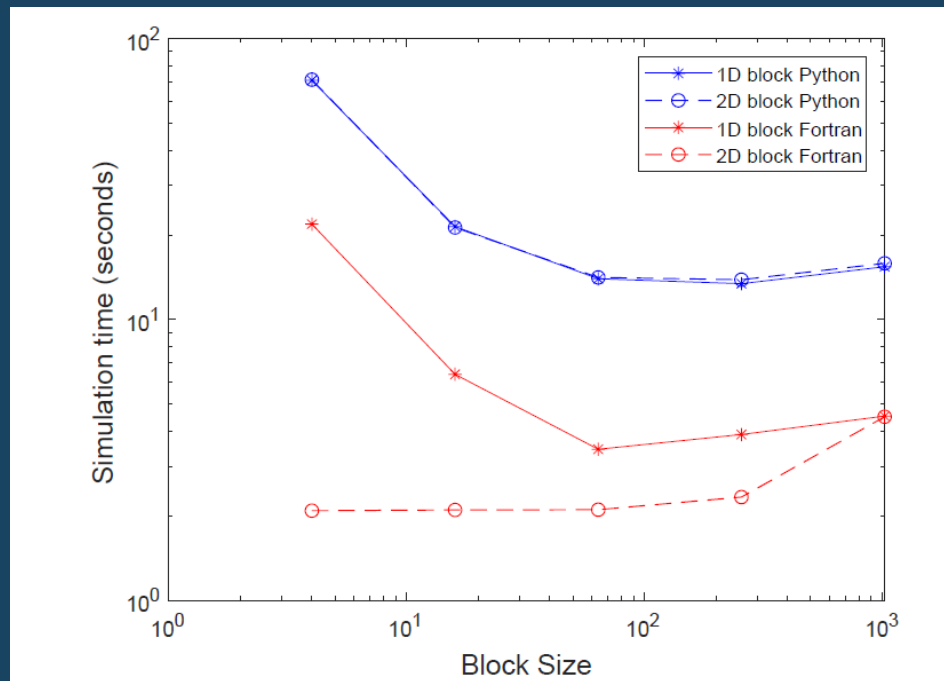
Results

Block size (1D vs 2D)



Results

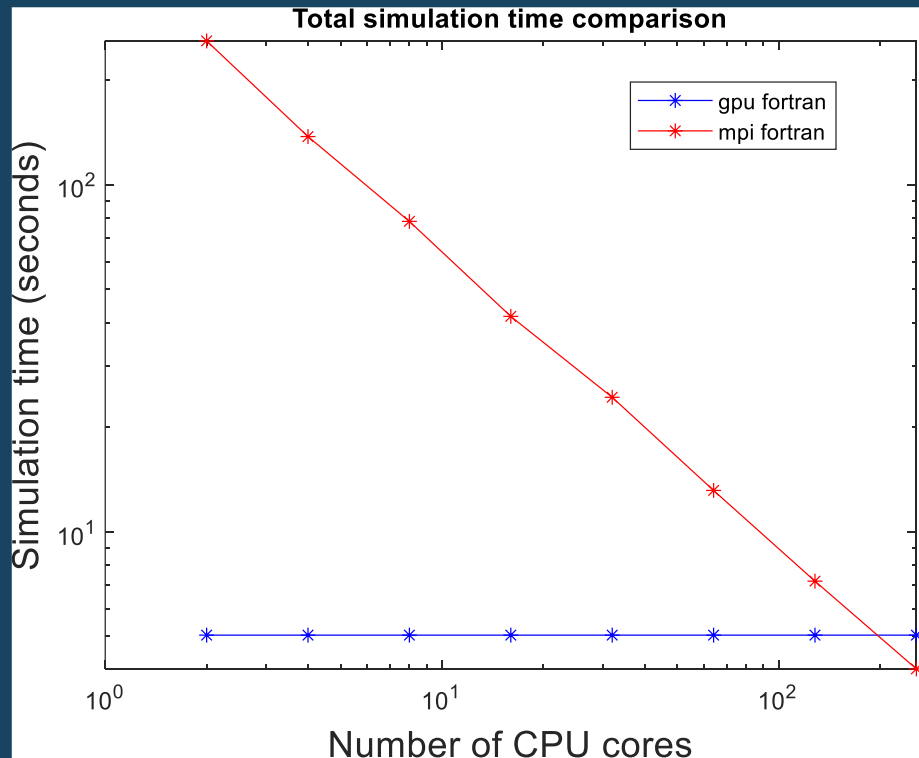
IMPACT OF BLOCK SIZE (256*256 particles for 10000 days without saving results)



A100

Results

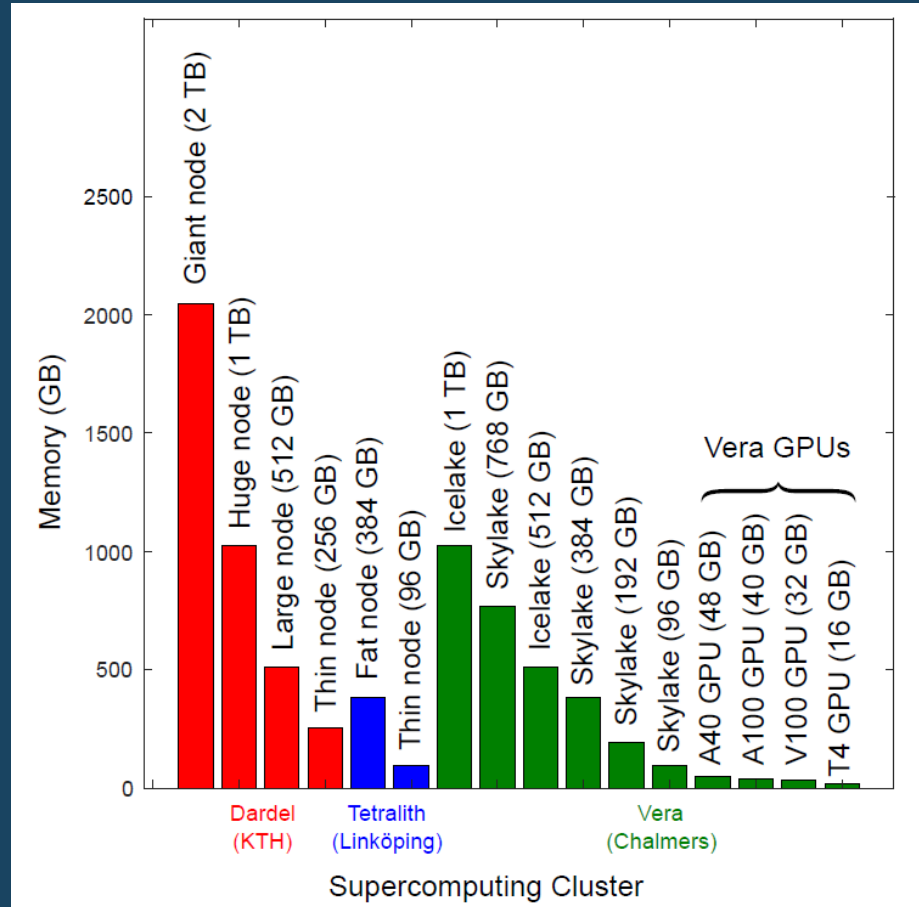
MPI vs CUDA (128*128 particles for 100 days)



- Cross-over point is around 200 CPU cores to match GPU performance.
- On Vera, a V100 GPU costs 80 core hours, and A100 GPU cost 52 core-hours.
- That means there is, at most, potential savings of 120 core hours per hour on V100 and 148 core hours per hour on A100 when running the code on GPU compared to CPU.

Limitations

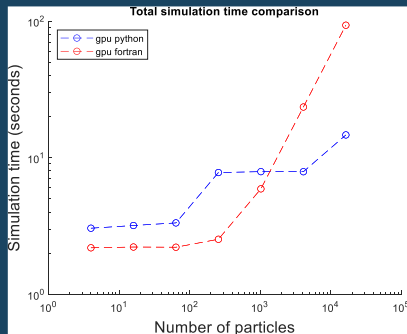
Limited memory capacity



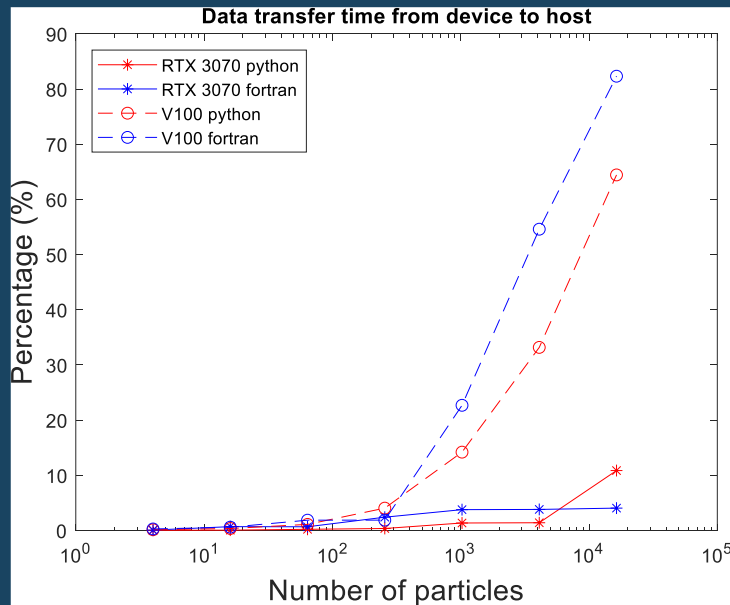
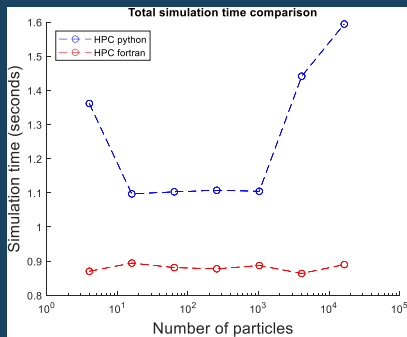
Limitations

Data Transfer Overheads

NVIDIA RTX 3070
(Math Compute Time)



NVIDIA A100
(Math Compute Time)



Memory transfer speed affects overall performance?

Limitations

Data Transfer Overheads

GPU acceleration of Eulerian-Lagrangian particle laden flows (Sweet et al.)

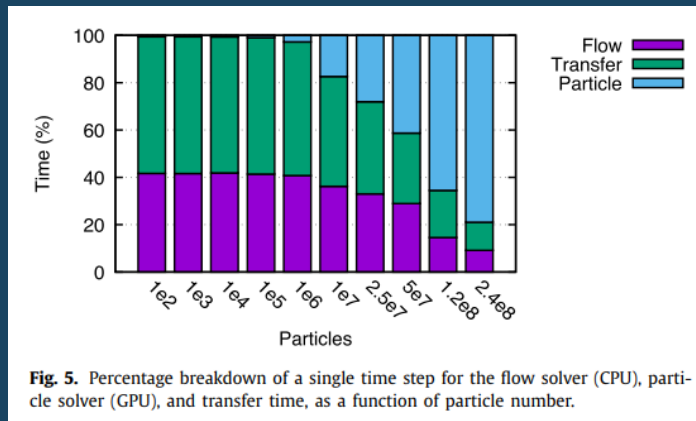
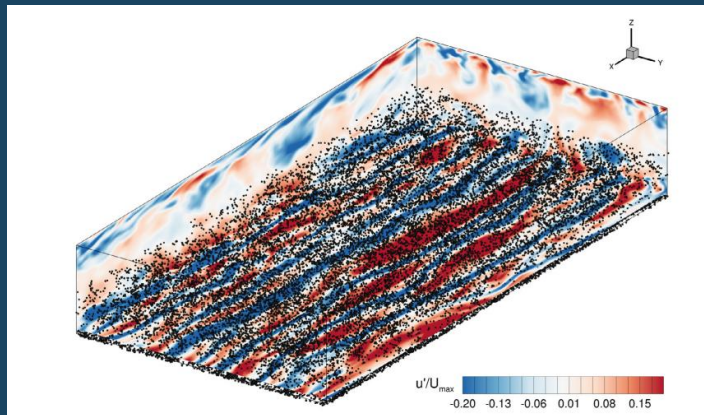


Fig. 5. Percentage breakdown of a single time step for the flow solver (CPU), particle solver (GPU), and transfer time, as a function of particle number.

- Large Scale GPU codes are parallelized across many GPUs.
- Computations are carried out on both CPU and GPU.
- GPU codes developed with NVIDIA's CUDA framework is incompatible with AMD GPUs, which restricts their portability and requires separate implementations for different hardware platforms.

Conclusion

Final Remarks

- Achieved Significant Performance Enhancements by Porting LPT Code to GPU.
 - **CUDA Python:** Attained speedups of up to **350 times**.
 - **CUDA Fortran:** Realized speedups of up to **391 times**.
- Best threads per block structure is 16x16 (256 threads).
- CUDA Fortran is very sensitive to threading structure.
- At the moment, several challenges still hinder GPUs from completely overhauling the CFD Sector.
- Nonetheless, porting CFD codes to GPUs remains a well-funded and actively pursued area of research.

Developed Codes Open Sourced on Gitub

Final Remarks

GPU_accelerated_LPT_CFD_code Public

1 Branch 0 Tags

Go to file

Add file

Code

About

This repository contains GPU accelerated version of the particle tracking model developed by Merel Kooi for biofouled microplastic particles (available at: <https://pubs.acs.org/doi/10.1021/acs.est.6b04702>) written in CUDA Fortran and CUDA Python. This repository is intended as a learning tool for GPU programming.

Readme

MIT license

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

Python 100.0%

thaisalmandula Update README.md

13 hours ago

5 Commits

Fortran_GPU Version 1.0

Python_GPU Version 1.0

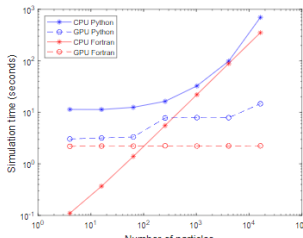
LICENSE Initial commit

README.md Update README.md

GPU_accelerated_LPT_CFD_code

This repository contains GPU accelerated version of the particle tracking model developed by Merel Kooi for biofouled microplastic particles (available at: <https://pubs.acs.org/doi/10.1021/acs.est.6b04702>) written in CUDA Fortran and CUDA Python. This repository is intended as a learning tool for GPU programming and developed during the TRA220 GPU-accelerated Computational Methods using Python and CUDA offered at Chalmers University of Technology (<https://www.tfd.chalmers.se/~lada/CUDA-tracks.html>).

The performance of an LPT code developed in Python and Fortran was evaluated on both CPU and GPU architectures. This study aimed to quantify the performance enhancements achieved by running LPT simulations on GPUs and to compare the efficiencies of CUDA implementations in Python versus Fortran. The number of particles was varied from 4×2 to 16,384 (128×128) to assess scalability and computational efficiency. The following simulations were conducted on an NVIDIA GeForce RTX 3070 GPU. The results are illustrated in the figure below.





CHALMERS
UNIVERSITY OF TECHNOLOGY