UNIVERSITY OF CAMBRIDGE

StationSim code acceleration

Project RADDISH

Eleftherios Avramidis
Research Software Engineering
Research Computing Services
University of Cambridge

StationSim

- Simulates crowds in real time with agent-based modelling and a particle filter (Nicolas Malleson, et al. 2019)
- Written in Python
- Good code quality
- No tests
- Jupyter notebooks examples
- Used with typical size of 40 agents and 10,000 particles

Project aim and organisation

- Multi-core/node and GPU implementation of the StationSim model and particle filter
- Stage 1 (done):
  - Familiarisation with model and Python code
  - Establishment of implementation strategy
- Stage 2 (in progress): Implementation using C++ with OpenMP and MPI
- Stage 3: GPU implementation

Extra aims

- Maintainable code
- Addition of tests and CI
- Write instructions for
  - Building
  - Running
  - Testing

StationSimCpp

- Private GitHub repository
- C++ shared library
- C++17 with OpenMP and MPI
- CMake to build, test and package the library
- CI using GitLab
- Needs more test (unit, integration)
- HDF5 for file output

Early runtime/speedup results – one CPU core

StationSim single model run with default parameters

Population	Python (s)	C++ (s)	Speedup (Python/C++)
100	4.58	0.034	134.7059
200	13.03	0.14	93.07143
400	40.51	0.74	54.74324
600	86.19	2.1	41.04286
800	153.54	4.55	33.74505
1000	265.71	7.67	34.64276

Preliminary results

Early runtime results – 6 CPU cores using OpenMP

StationSim multiple model runs with default parameters and population size of 400

Model instances (i.e. particles)	Runtime
100	11.1435
200	22.1494
400	40.3555
600	61.8257

Preliminary results

Next steps

- Improve MPI multi-node implementation
- Implement simple particle filter
- Implement HDF5 outputs code for particle filter
- Performance profiling
- GPU code
- Improve repository and documentation
- Finalise code API