



The
University
Of
Sheffield.

Fixed Radius Near Neighbours on the GPU

Robert Chisholm - Visual Computing Research Group

Supervised by: Dr Paul Richmond, Dr Steve Maddock

Motivation & Aims

Complex simulations provide a convenient means for researching fields which are impractical to carry out real-world experiments.

These systems are often decomposed into collections of many independent actors (e.g. people, particles), which require live information about their neighbouring agents.

Central to this lies the operation of fixed radius near neighbours (FRNNs) which allows agents to iterate their neighbours. In a dynamic environment however, this process is expensive, far outweighing the cost of other agent behaviours.

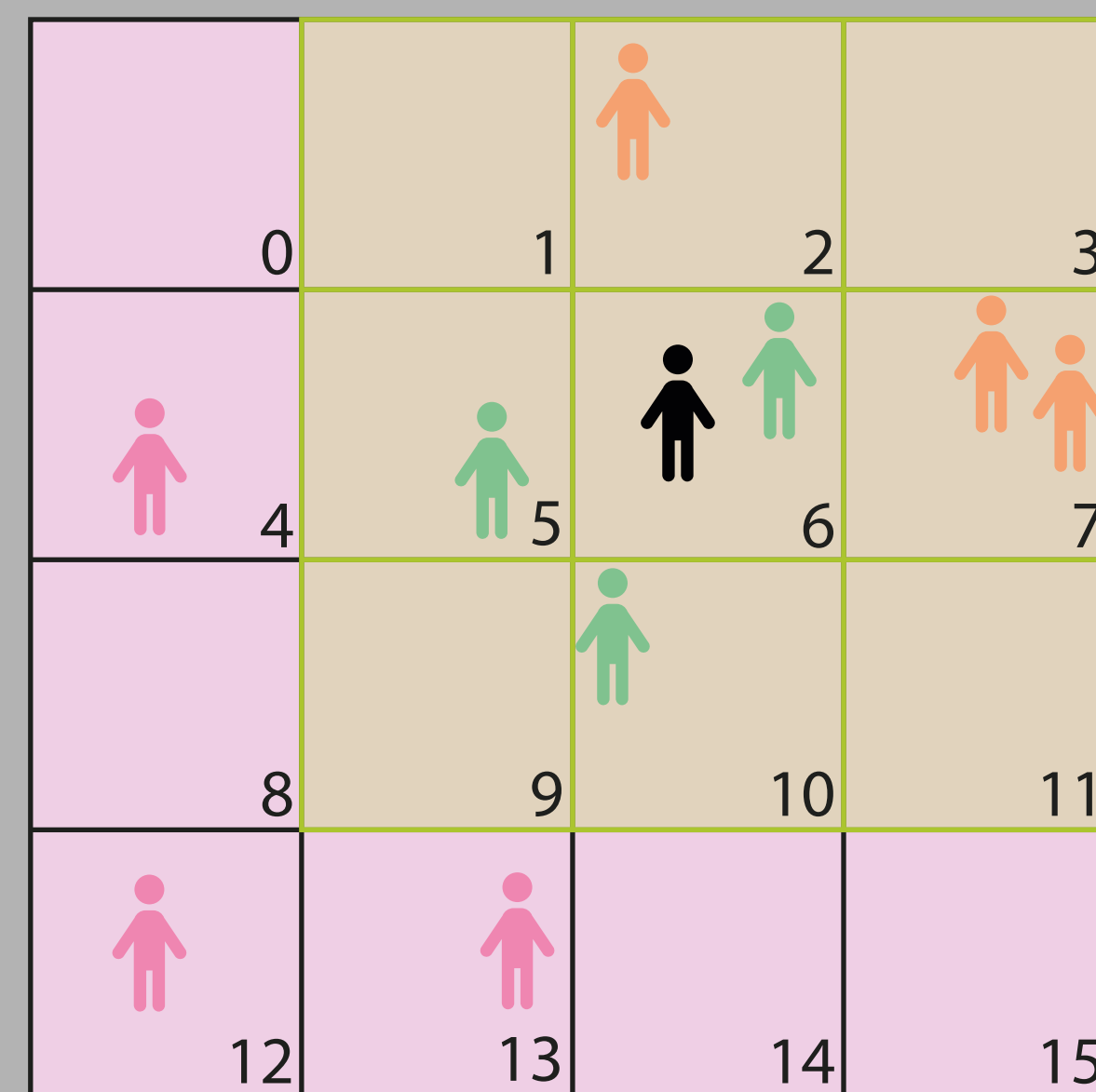
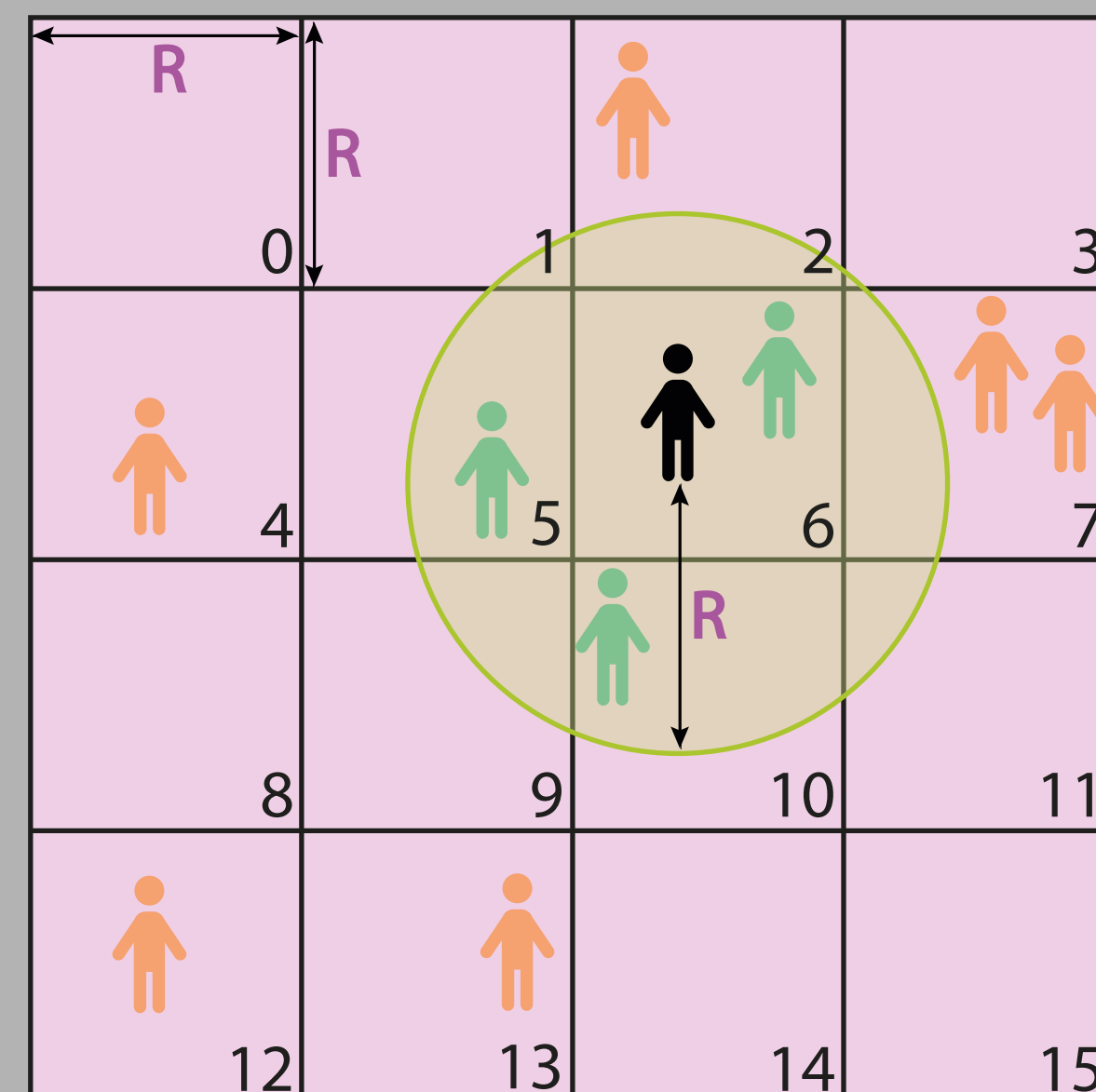
By improving the performance of FRNNs we are able to benefit a wide range of fields and improve awareness of advanced GPU optimisation techniques.

Algorithm

Every time step:

Every agent:











Considers neighbouring agents and decides their next state



Agent Storage

Array ID

Agent Data
(Cell ID included for clarity)

0	1	2	3	4	5	6	7	8	9
									

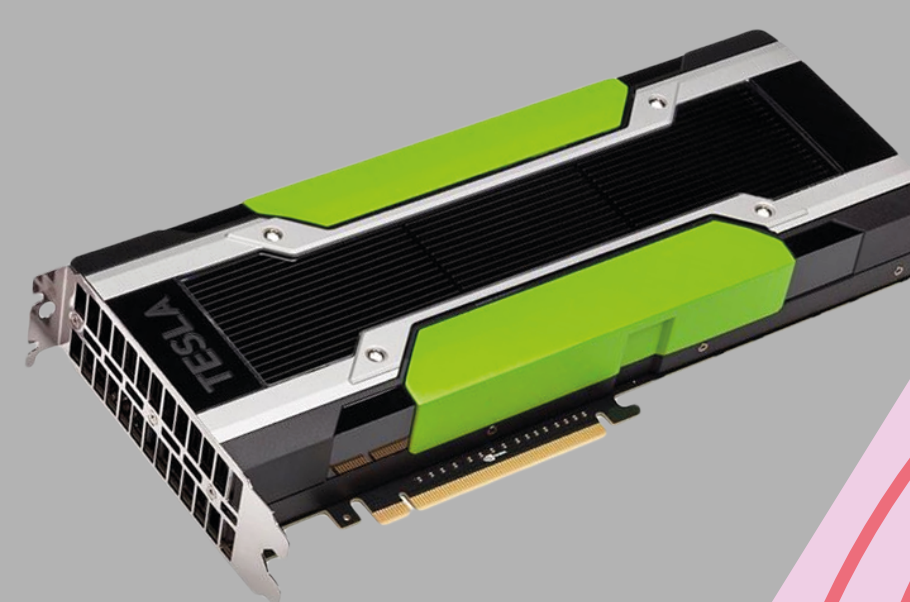
Boundary Index

Cell ID

Cell Data Start ID

Cell Data End ID
(Redundant, Use Next Start ID)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	1	1	2	3	5	7	7	7	8	8	9	10	10
0	0	1	1	2	3	5	7	7	7	8	8	9	10	10	10



Moore Neighbourhood

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Agent Storage

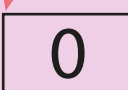

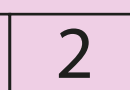
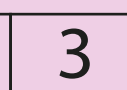


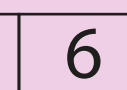

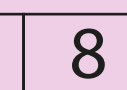
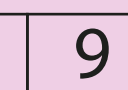






Cell ID

Cell Data Start ID

Boundary Index

Array ID

Agent Data
(Cell ID included for clarity)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	1	1	2	3	5	7	7	7	8	8	9	10	10
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
															

Uniform Spatial Partitioning

1. Decide the interaction radius: **R**
2. Partition the environment:
 - Uniform cells of height/width **R**
 - Clamp agents to environmental bounds
3. Search the Moore neighbourhood:
 - Ignoring agents outside of the Euclidean distance **R**

Figures: Top-Left (1-2), Bottom-Left (3)

Implementation: Construction

1. Sort the agent data according to their cell index's into an array
2. Build a boundary index of where each cell's agent storage begins

Figure: Top-Right (1-2)

Implementation: Search

1. For each cell in the Moore neighbourhood
2. Locate the first index of the cell's data
3. Locate the first index of the next cell's data
4. For each agent within this range:
 - Calculate the Euclidean distance
 - If distance $\leq R$:

The agent is a valid neighbour

Figure: Bottom-Right (1-3)

Advances for Investigation: How Effective Are They?, How Do They Interact?

Increasing Memory Locality

Space filling curves reduce multi-dimensional problems in to single dimension. They can increase memory locality, improving cache usage.

Awareness of Pre-Sorted Data During Construction

Some serial sorting algorithms excel with pre-sorted data. Can similar attributes be achieved with parallel sorts?

Reducing Unnecessary Memory Accesses

Reducing the dimensions of partitioning cells allows a greater fidelity of agents targeted. However this also decreases locality, by splitting agent into more rows/columns.

Reducing Memory Read Contention

Each agent wishes to search the same number of cells. Currently they all start from the lowest ID, leading to potentially huge scattered accesses. The effective memory bandwidth can be improved by ensuring agents search the same cell simultaneously.

Load Balancing

Agents with the least neighbours must wait for agents with the most neighbours in their thread group (warp) to finish. Non-uniform agent distributions can make this effect significant. Load balancing via work queues would improve device utilisation.

Summary

This research seeks to advance FRNNs, a core process of complex simulations, improving access to high performance modelling benefiting a wide range of fields. GPU accelerated FRNNs is a memory access bounded problem, compounded by the requirement of regular live information. Approaches for improving the performance therefore focus on techniques for reducing scattered and unnecessary memory accesses in order to better utilise the available hardware.