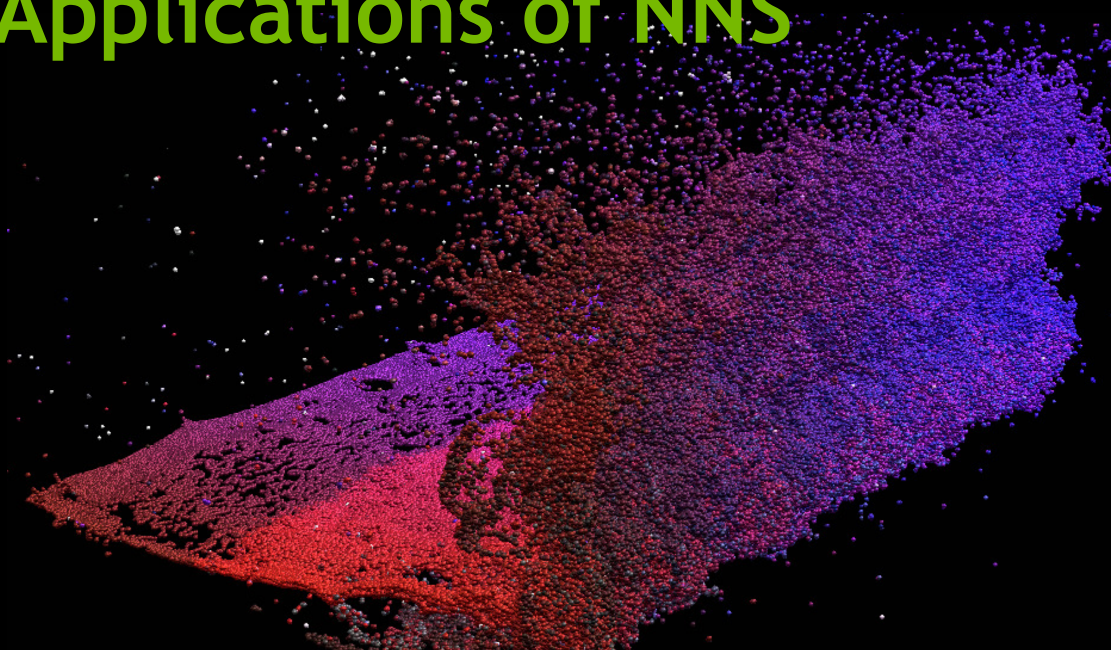
GPU TECHNOLOGY CONFERENCE

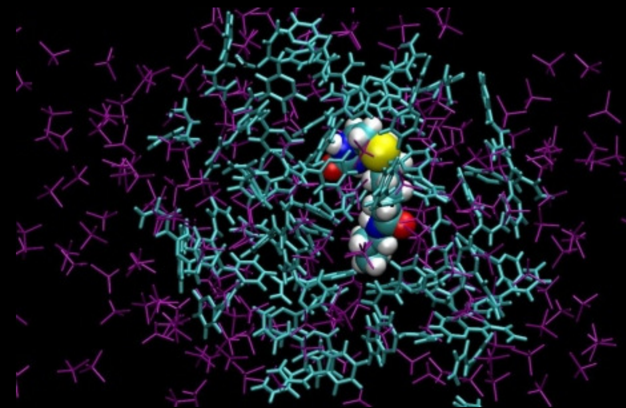
FAST FIXED-RADIUS NEAREST NEIGHBORS: INTERACTIVE MILLION-PARTICLE FLUIDS

Rama C. Hoetzlein, Graphics Devtech, NVIDIA

Applications of NNS



Fluid Simulation

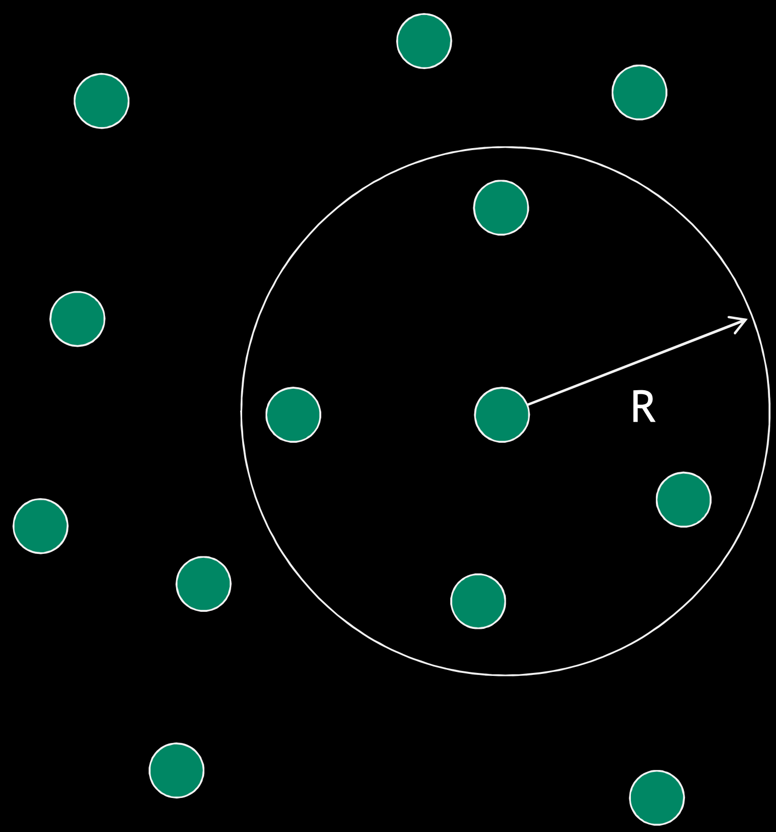


Molecular Modeling

Cognitive Science – Behavioral simulation

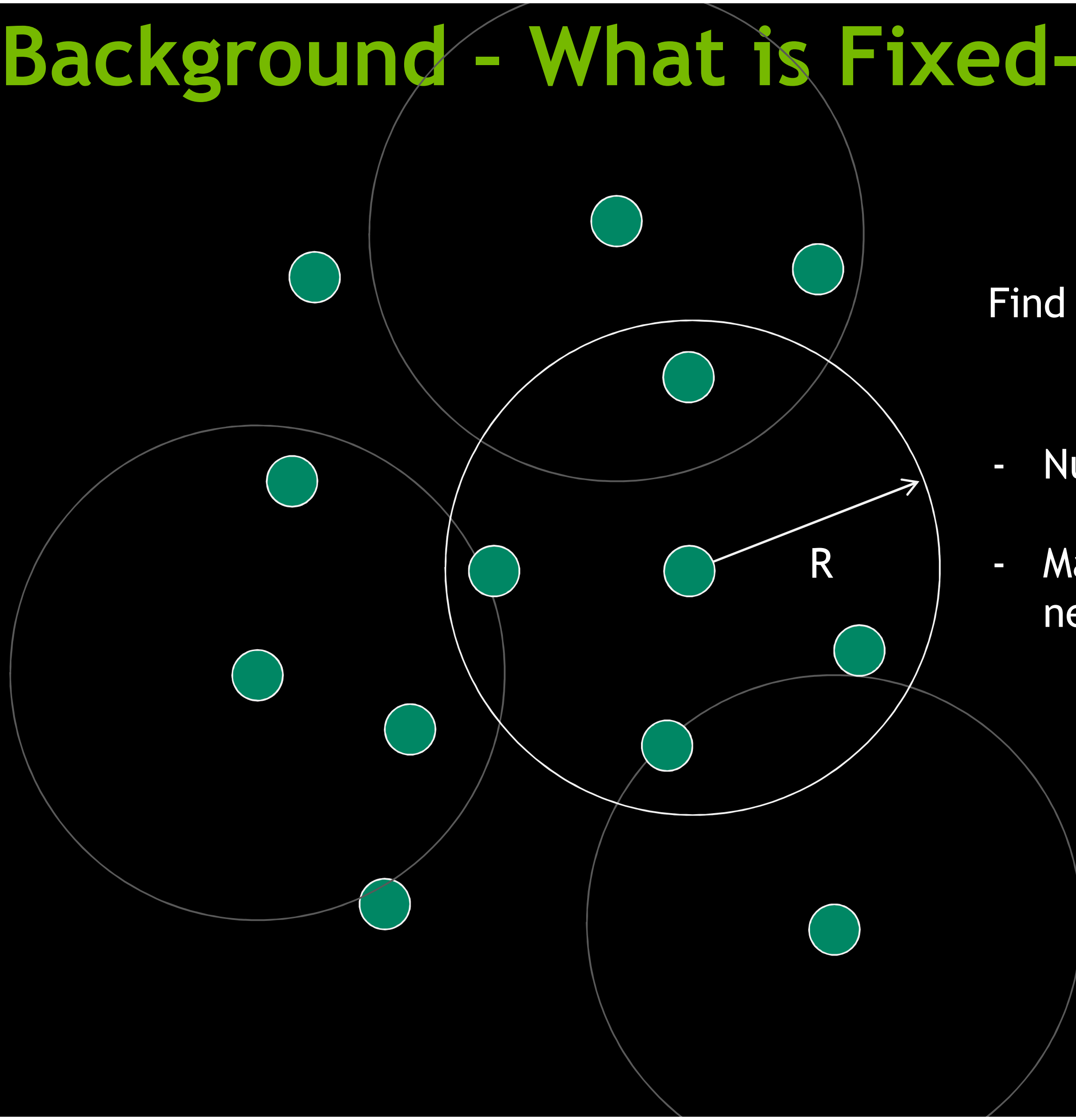
3D Scanning - Point Cloud Reconstruction

Background - What is Fixed-Radius NNS?



Find all neighbors in a fixed radius R

Background - What is Fixed-Radius NNS?

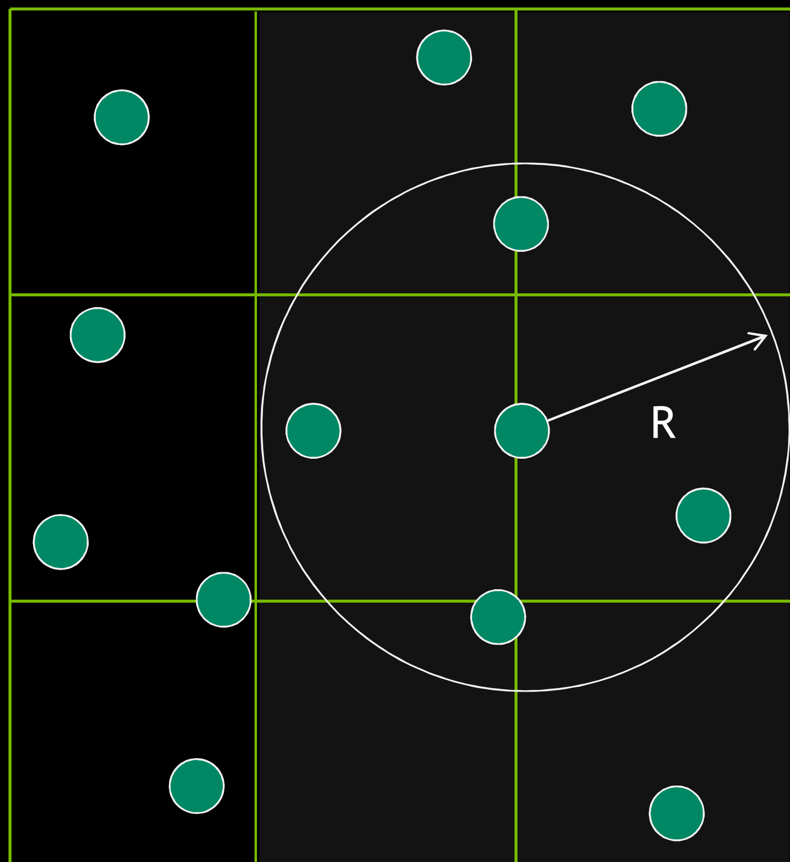


Find all neighbors in a fixed radius R

- Number of neighbors may vary
- May need to find all fixed-radius neighbors of all particles

O (n^2)   brute force

Overall Strategy



Spatial Partitioning:

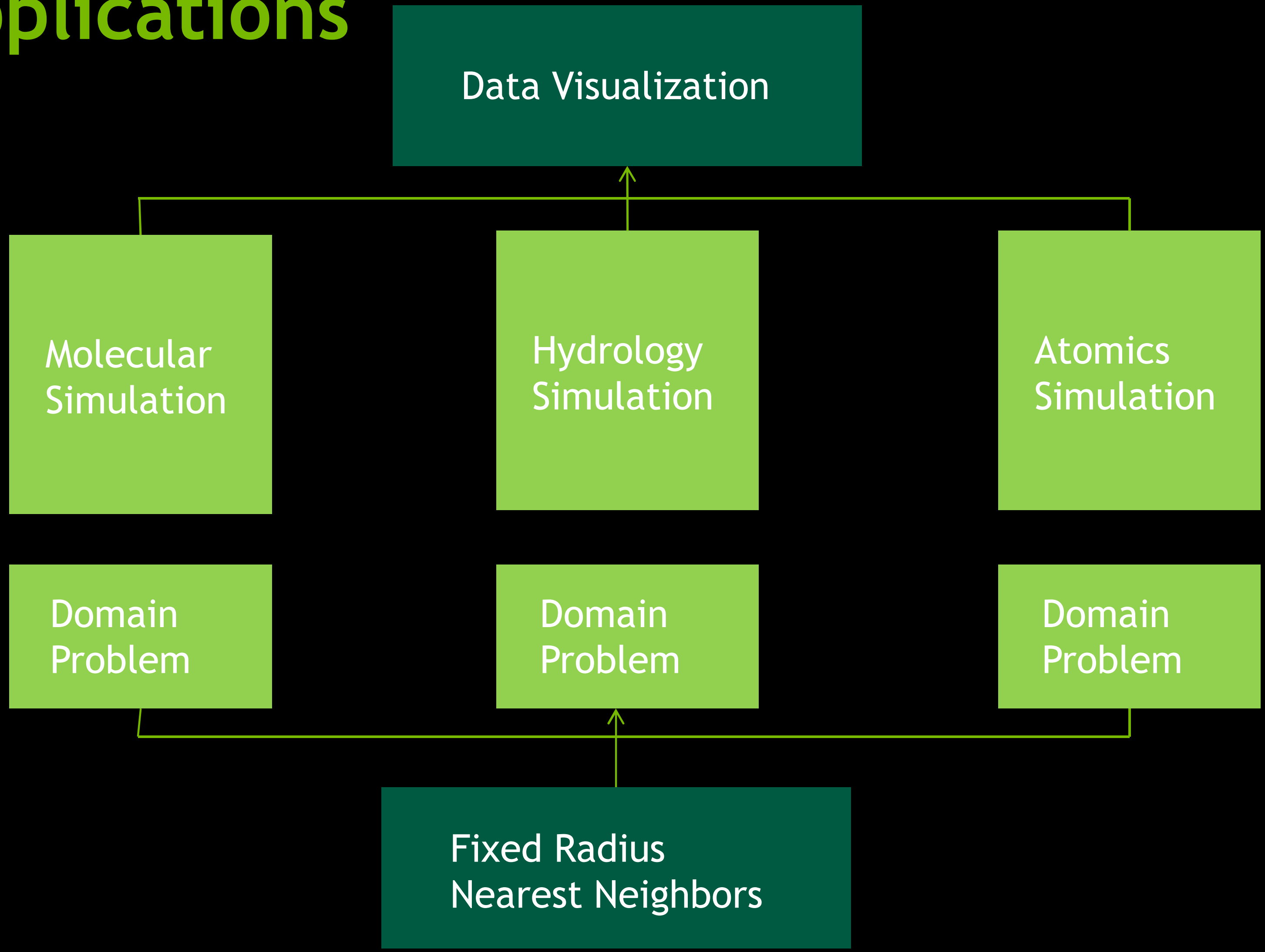
1. Partition space equally into bins

2. Insert each particle into bins

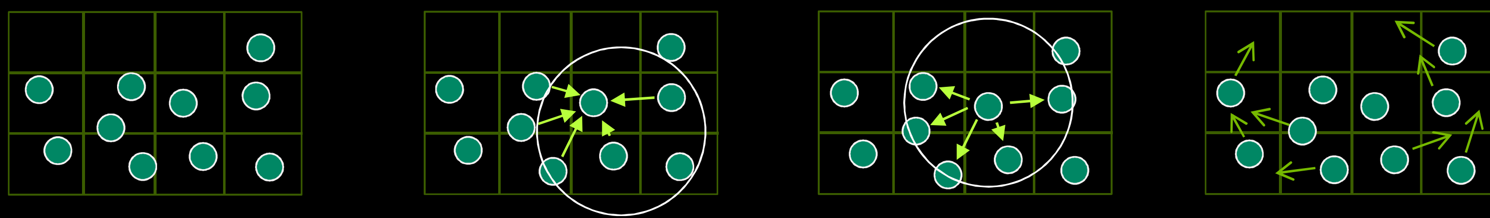
3. Only need to search particles found in neighboring bins

O (N k)

Applications



Applications - Example



Insert Particles	Compute Pressures	Compute Forces	Integration (Advance)
Add to bins	Find neighbors / Compute P	Find neighbors / Compute F	Move particles
$O(N)$	$O(\ N(k+D_1)\ )$	$O(\ N(k+D_2)\ )$	$O(N * D_3)$

Nearest Neighbor Search

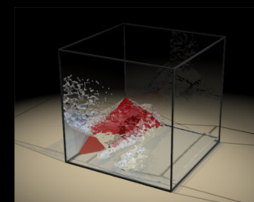
Domain Specific Cost

GPU TECHNOLOGY CONFERENCE

Research Background

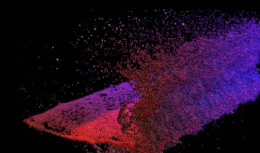
Fluid Simulation

2003 Muller
Particle-Based Fluid Simulation for Interactive Apps (SPH)

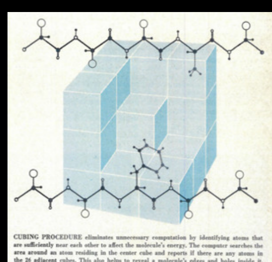


2009 Solenthaler
Predictor-Corrector (PCISPH)

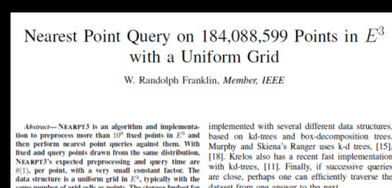
2013 Macklin & Muller
Position Based Fluids



Fluids v.3 (SPH)



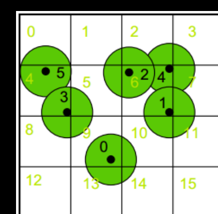
1966 Levinthal
Molecular-Model Building by Computer "Cubing" method



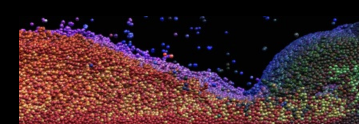
Nearest Point Query on 184,088,599 Points in E^3 with a Uniform Grid

W. Randolph Franklin, *Member, IEEE*

2006 Randolph Franklin
Nearest Point Query on 184,088,599 Points in E^3 with a Uniform Grid.
CPU Grid Search



2010 Simon Green
Particle Simulation using CUDA
Parallel Radix Sort

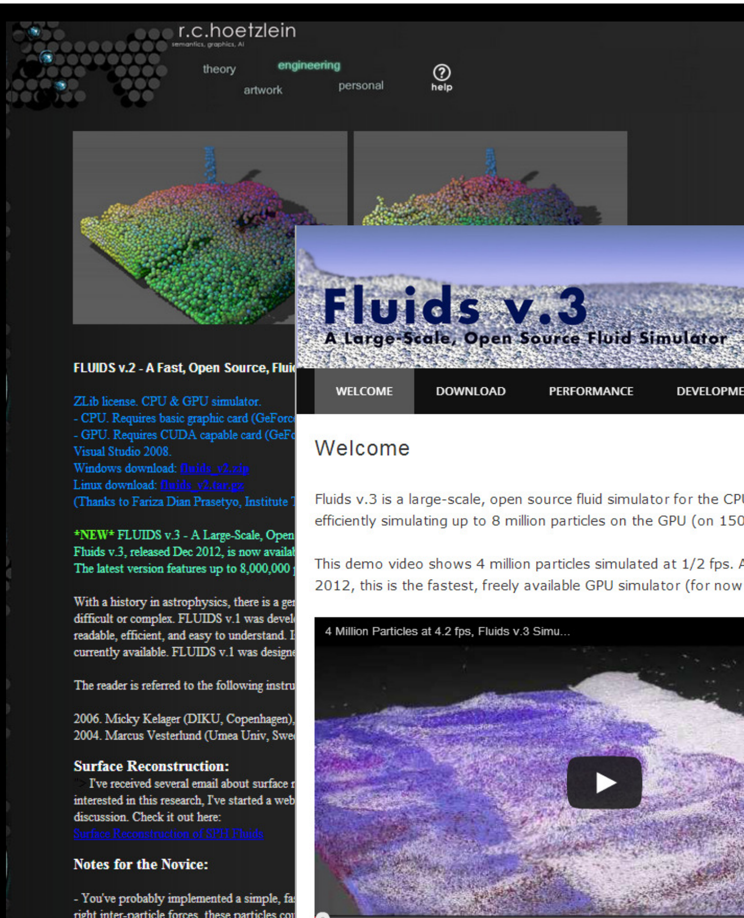


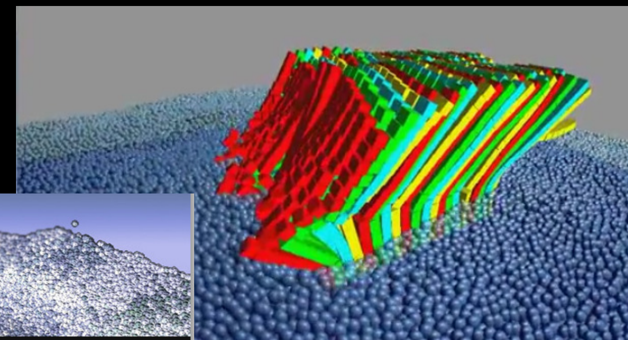
2013 Hoetzlein
Parallel Counting Sort NNS

Nearest Neighbor Search

GPU-based NNS

GPU TECHNOLOGY CONFERENCE



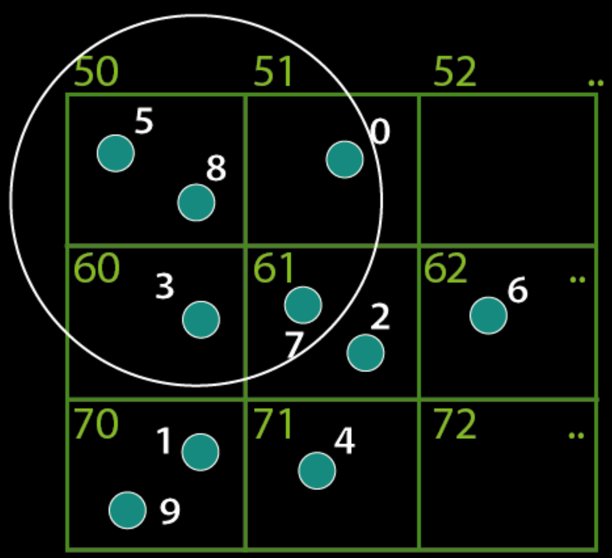


BULLET PHYSICS LIBRARY

Fluids v.3 http://fluids3.com

High Performance computing for scientific applications.
“Just the basics”, Zlib license, Research & Education, Solves NNS

Grid Search



Original Input

0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70

Neighboring Bins

50 51 60 61

0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70

SLOW - SCATTERED READS

Ideal Layout - Sorted by bins

5	8	0	3	7	2	6	1	9	4
50	50	51	60	61	61	62	70	70	71

Neighboring Bins

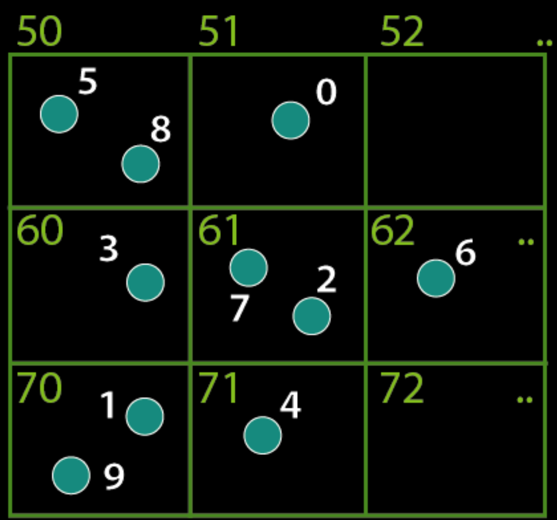
50 51 60 61

5	8	0	3	7	2	6	1	9	4
50	50	51	60	61	61	62	70	70	71

COHERENT READS !

Radix Sort

Previous method (Green 2008)



Input

0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70

Radix Sort

Digit 0

0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70

0 = 5   1 = 4   2 = 1

70	60	50	50	70	51	61	71	61	62

Digit 1

0	1	2	3	4	5	6	7	8	9
70	60	50	50	70	51	61	71	61	62

5 = 3   6 = 4   7 = 3

50	50	51	60	61	61	62	70	70	71

Repeat step for each digit in key

Output

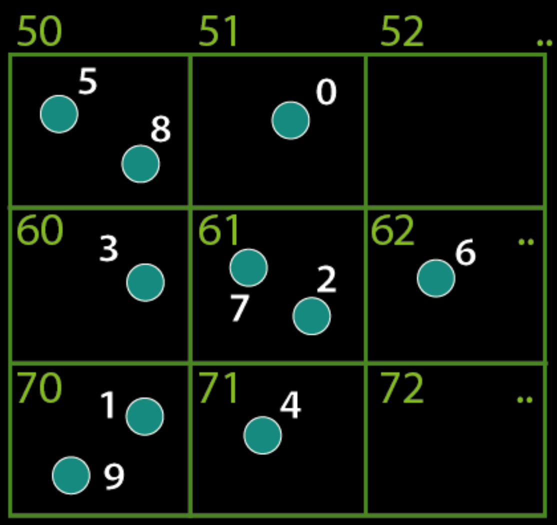
5	8	0	3	7	2	6	1	9	4
50	50	51	60	61	61	62	70	70	71

GPU TECHNOLOGY CONFERENCE

Counting Sort

Input

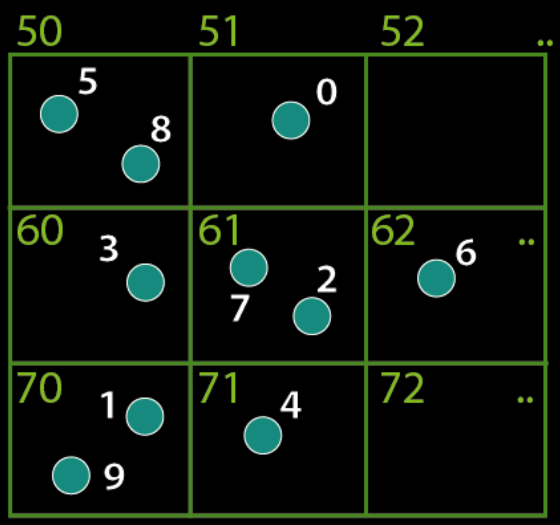
0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70



Observations:

1. Goal is to sort by bin
2. Position inside bin in irrelevant
3. Therefore, many duplicate keys
4. Better to perform 1-radix on exact bins, rather than on digits.

Counting Sort



Input

0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70

Insert + Counts

0	1	2	3	4	5	6	7	8	9
51	70	61	60	71	50	62	61	50	70
1	1	1	1	1	1	1	2	2	2

50 = 2	60 = 1	70 = 2
51 = 1	61 = 2	71 = 1
52 = 0	62 = 1	72 = 0

Prefix Sum

50	50	51	60	61	61	62	70	70	71
0		2	3	4		6	7		9

Counting Sort

0	1	2	3	4	5	6	7	8	9
		1					2		
5	8	0	3	2	7	6	1	9	4
50	50	51	60	61	61	62	70	70	71

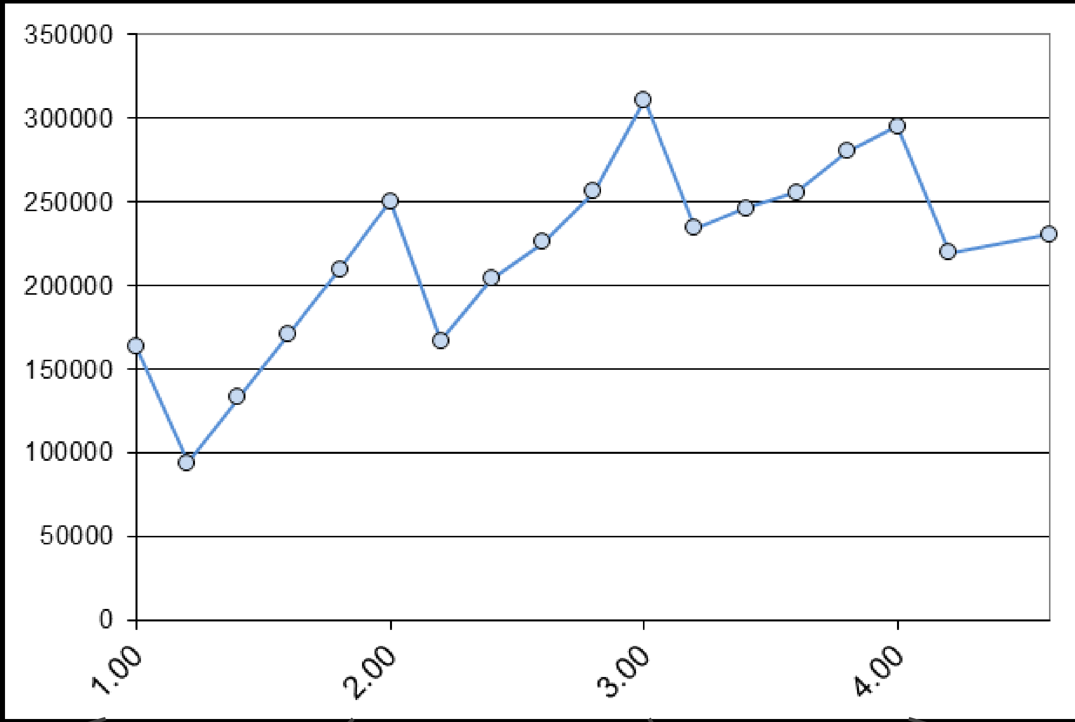
Output

5	8	0	3	2	7	6	1	9	4
50	50	51	60	61	61	62	70	70	71

What is a good Grid Cell Size?

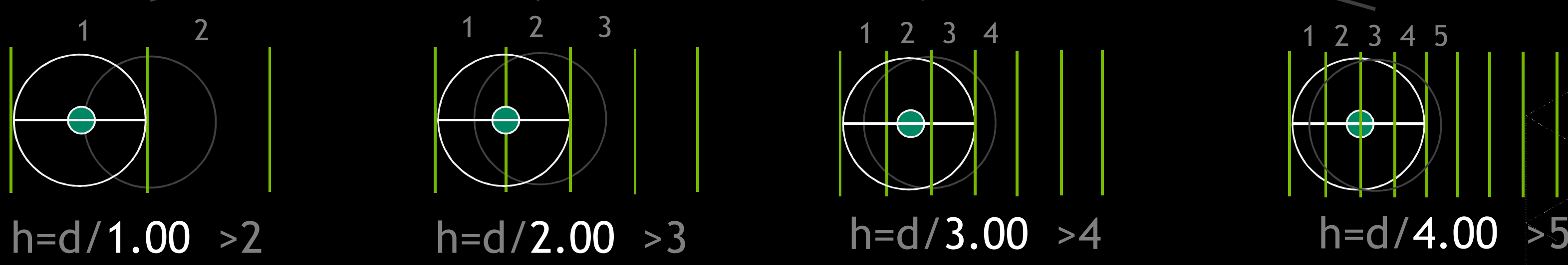
Too many particles per cell

e.g.
p = 200+ / cell



Too many grid cells to check

e.g.
5x5x5 = 125 cells



Algorithm Comparison

CUDA Particles (Radix Sort)

Insert Particles
- assign particle to cell

Sort Particles
- thrust:sort_by_key
- example: CUDA RadixSort
- for 1 to 4 (each byte in key)
  - Bin Counts
  - Bin Prefix Sum
  - RadixAddOffsetAndShuffle

Reindex (copy particles in order)

Time integration

Fluids v.3 (Counting Sort)

Insert Particles
- assign particle to cell

Sort Particles
  - Bin Sums
  - Bin Prefix Sum
  - ~~Radix Offset and Shuffle~~

ReIndex (copy particles in order)

Time integration

GPU TECHNOLOGY CONFERENCE

Algorithm Comparison

CUDA Particles (Radix Sort)

1 Insert Particles
    assign particle to cell

Sort Particles
    thrust:sort_by_key
    example: CUDA RadixSort
    for 1 to 4 (each byte in key)
4   Bin Counts
4   Bin Prefix Sum
4   RadixAddOffsetAndShuffle

1 Reindex (copy particles in order)

1 Time integration

15 Kernel calls / Frame

Fluids v.3 (Counting Sort)

1 Insert Particles
    assign particle to cell
    AtomicAdd for Bin Counts & Indices

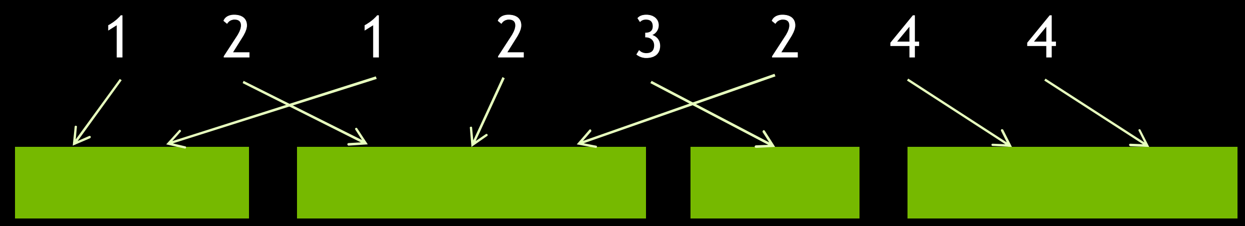
Sort Particles
        ~~Bin Sums~~
1   Bin Prefix Sum
        ~~Radix Offset and Shuffle~~

1 ReIndex (copy particles in order)

1 Time integration

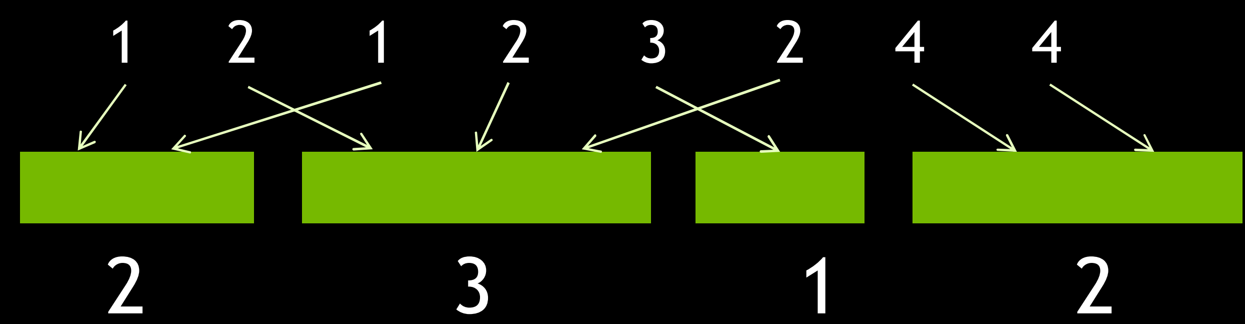
4 Kernel calls / Frame

Bin Counting



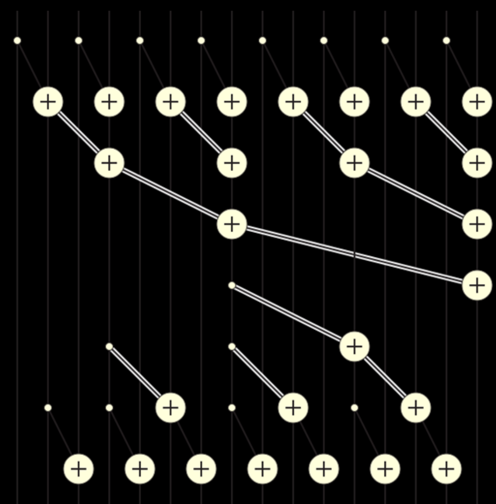
How many? 2 3 1 2

Bin Counting



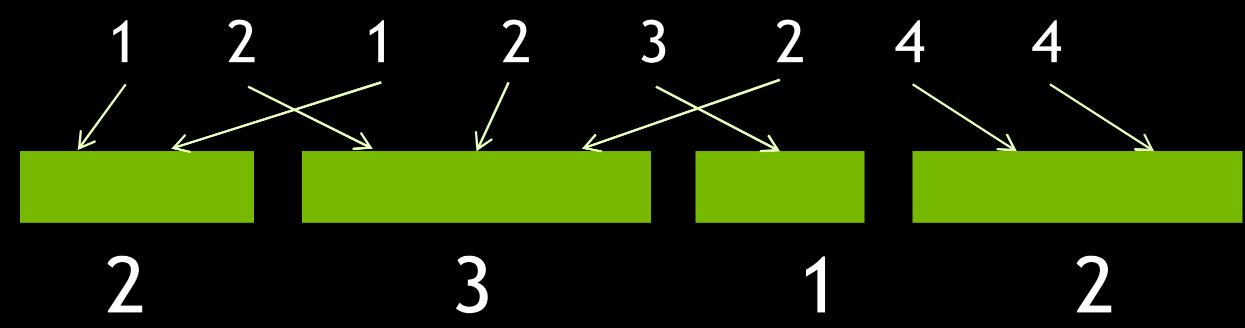
Old Method

Parallel Sums



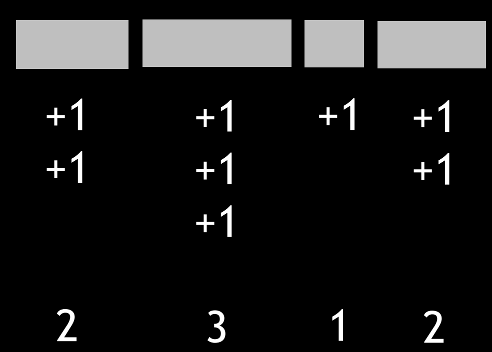
Each GPU thread computes binary sets of sums.

Bin Counting



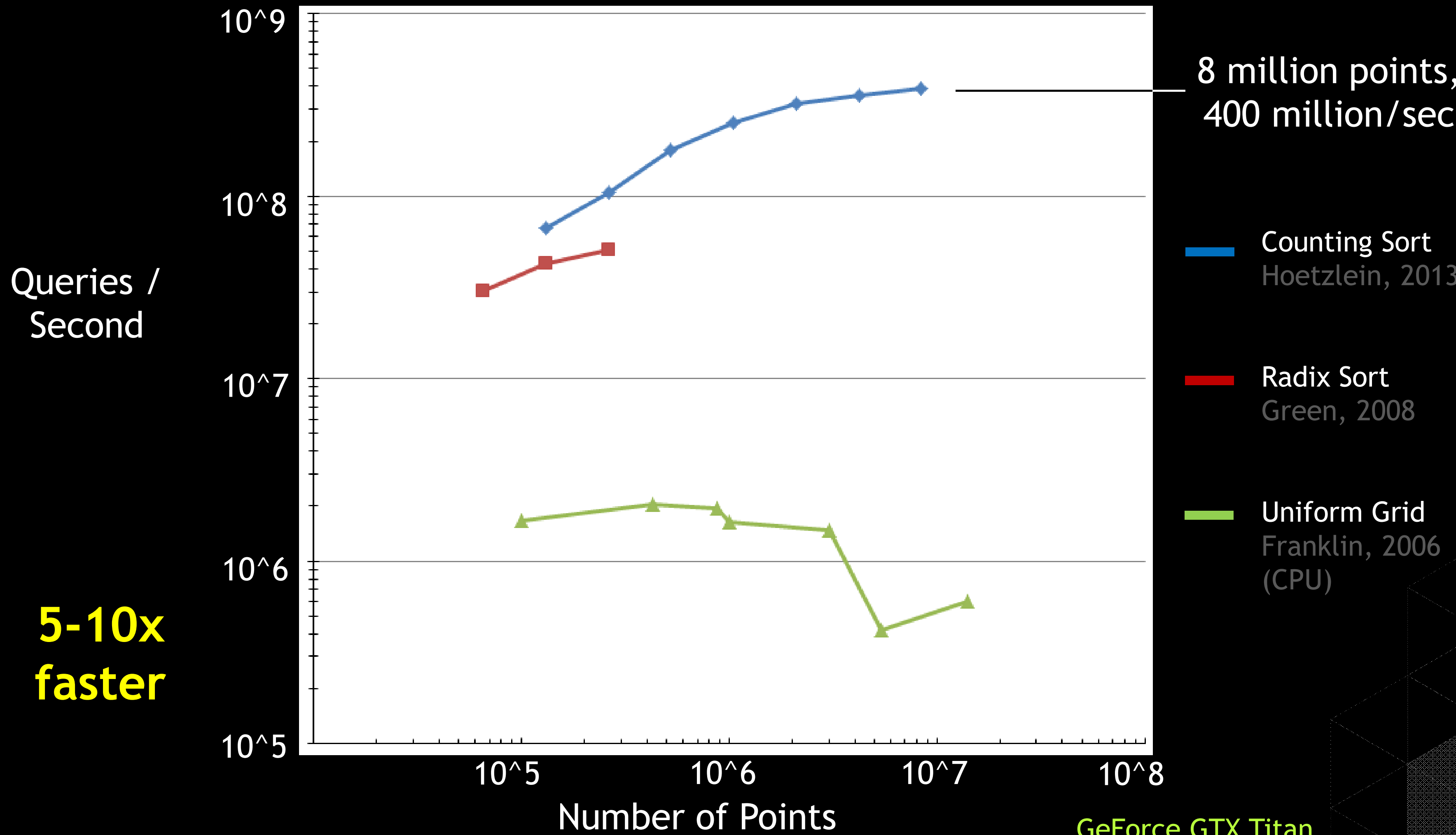
New Method

Atomic
Adds



Each particle
atomic-adds into bin
at the same time
bin is determined.

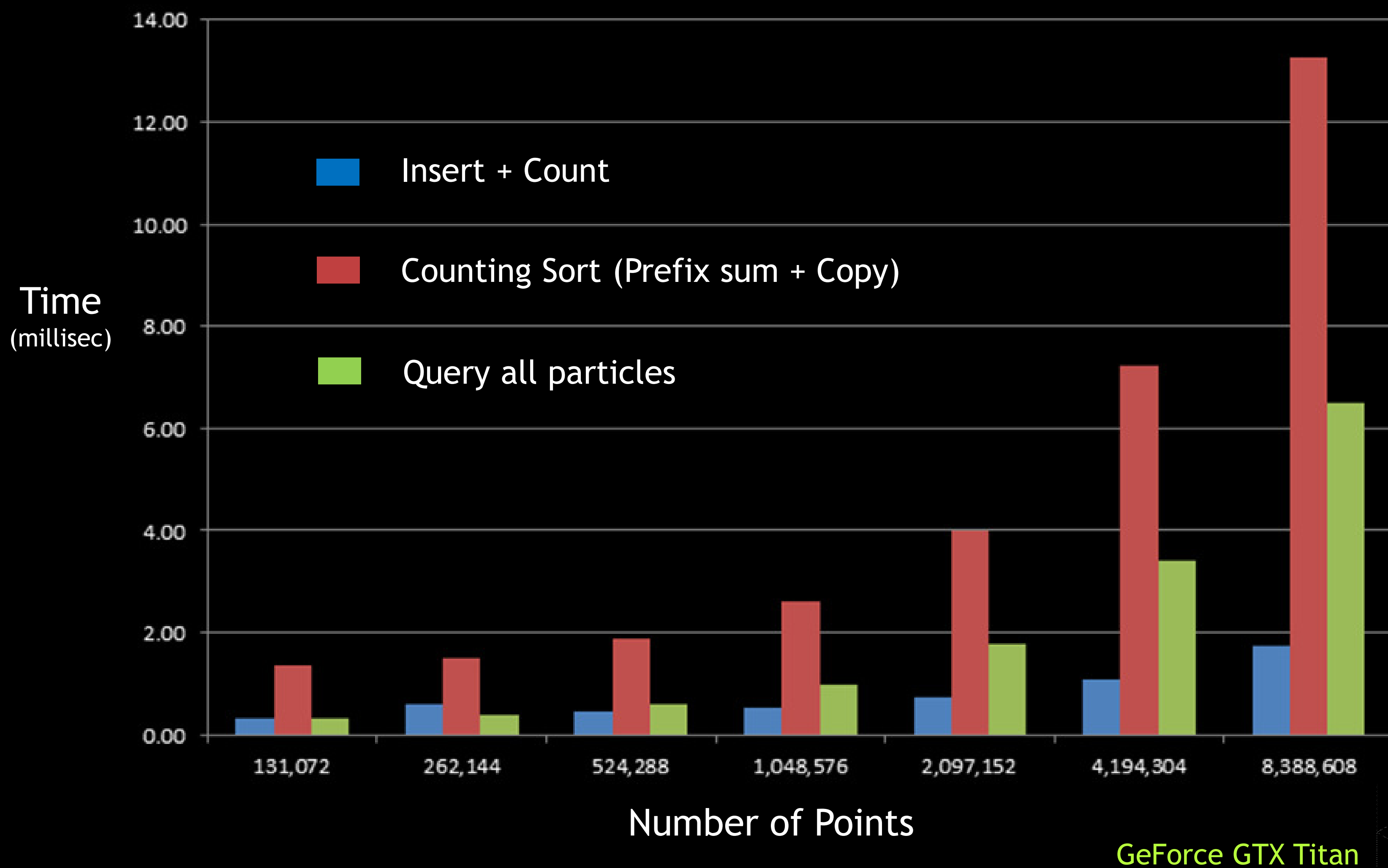
Results - Queries per Second



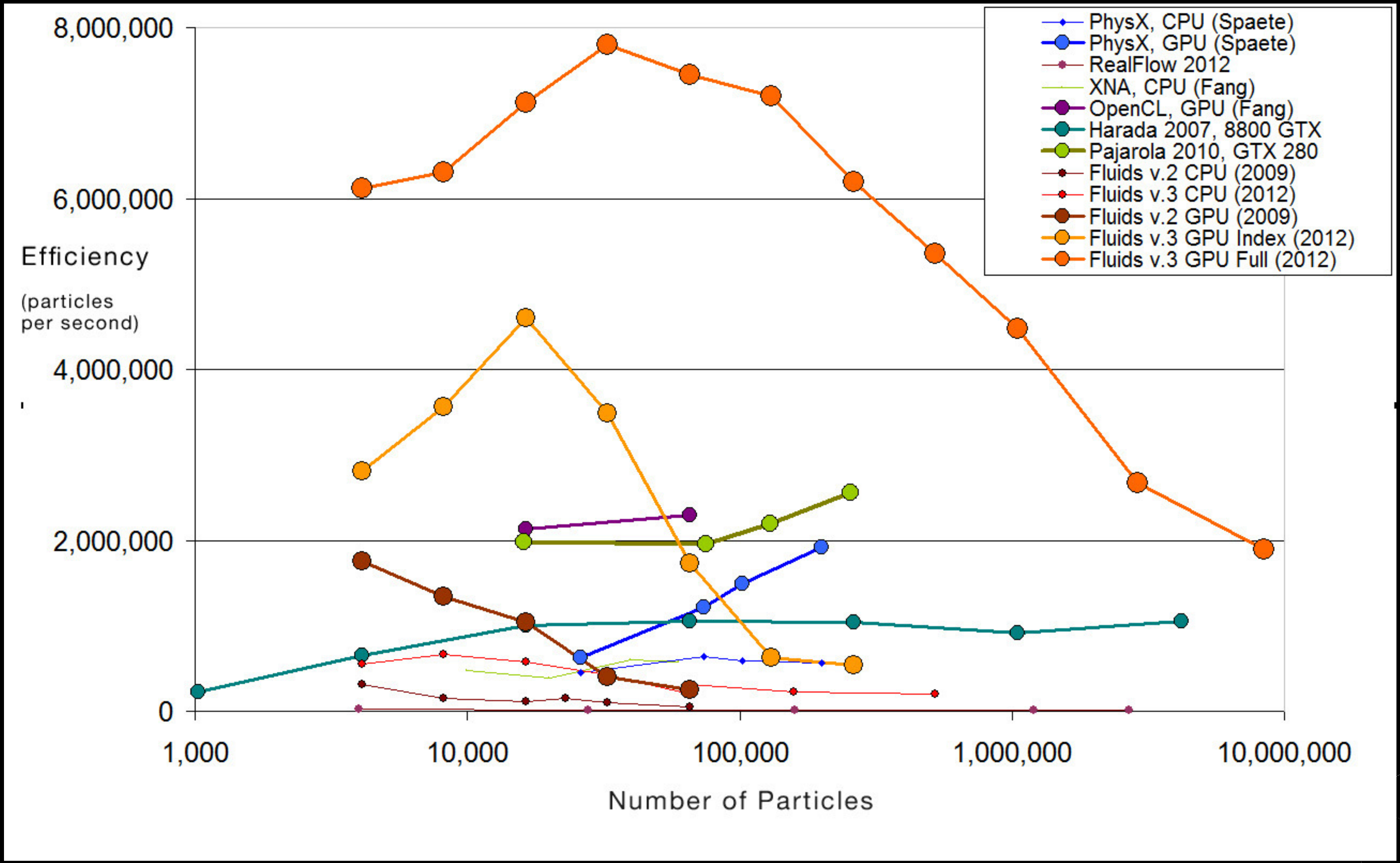
5-10x faster

GeForce GTX Titan

Results - Time for Each Step

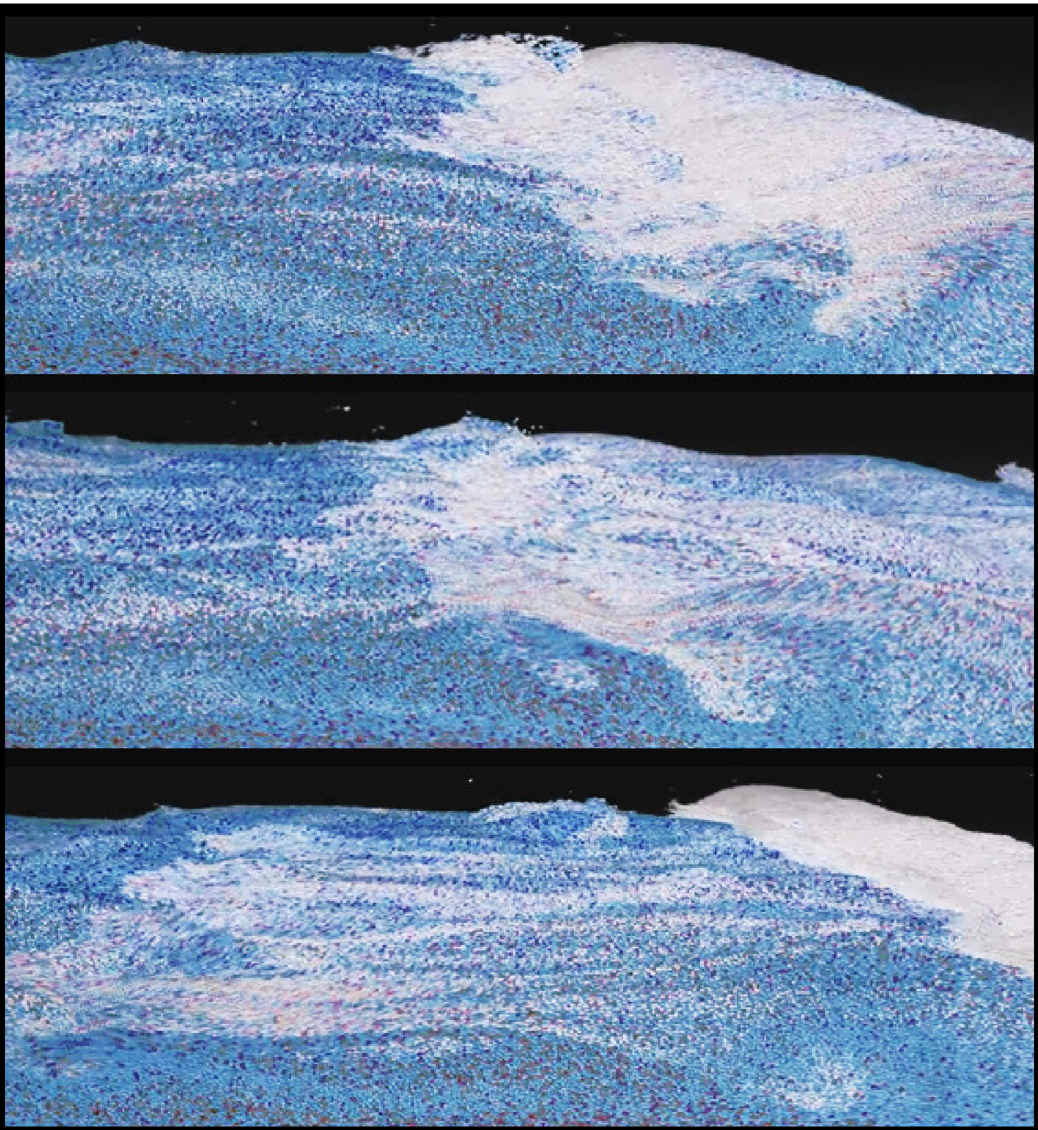


Results - Fluids Example

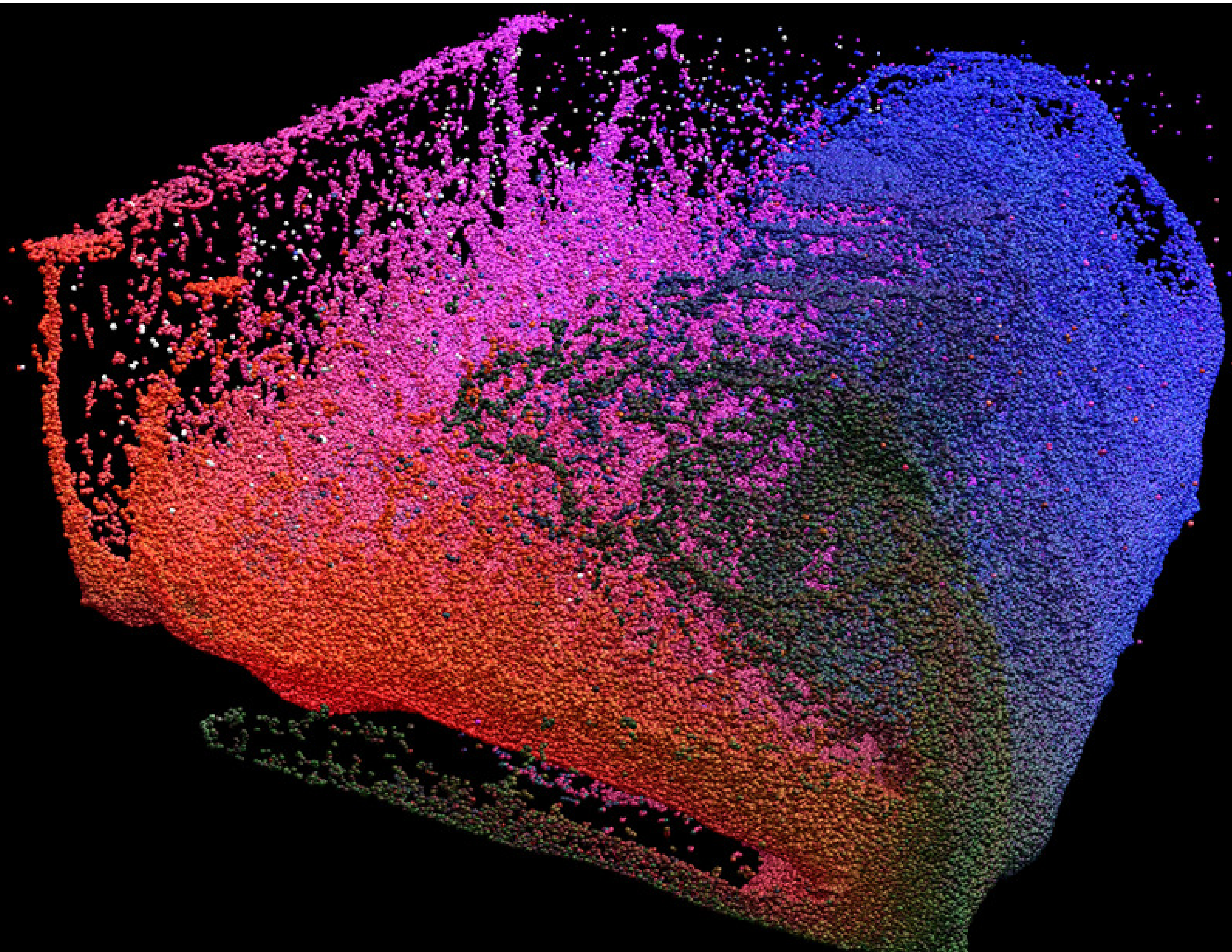


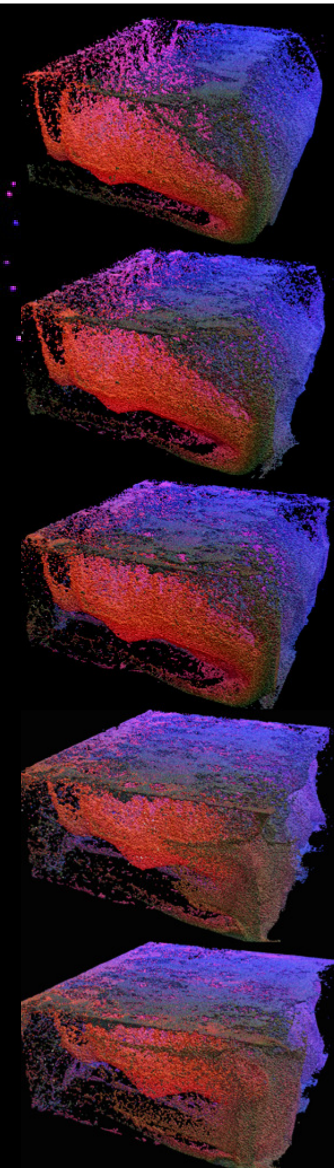
** Results do not show fluid accuracy or time step.*

GPU TECHNOLOGY CONFERENCE



Continuous Ocean Simulation, 4.2 fps, 4,194,304 particles

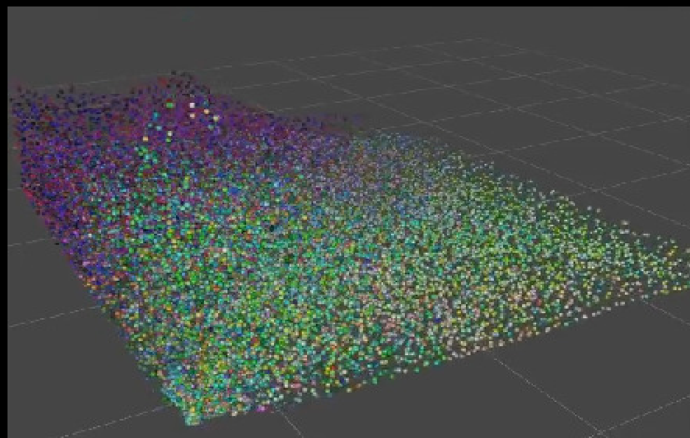




Interactive Microgravity Simulation. 12 fps, 2,097,152 particles

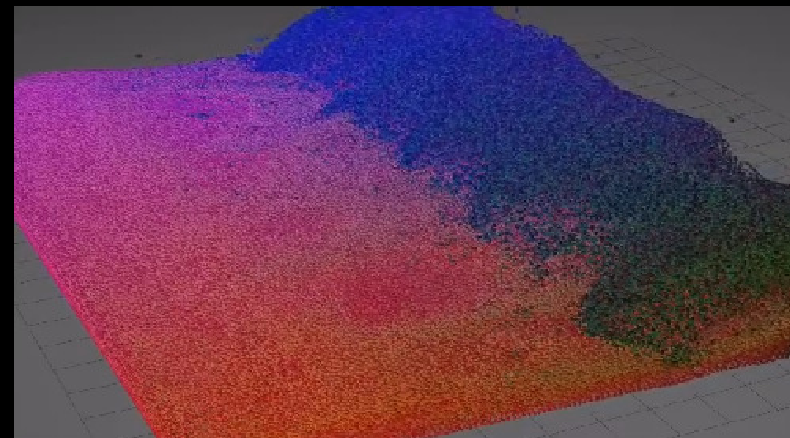
GPU TECHNOLOGY CONFERENCE

GeForce GTX 460M



Fluids v.2, 2009
16,384 at 32 fps

same hardware
11x faster



Fluids v.3, 2013
193,487 at 32 fps

Thank you!

- Rama C. Hoetzlein, rama@rchoetzlein.com

http://fluids3.com