A CUDA Monte Carlo simulator for radiation therapy dosimetry based on Geant4

SNA+MC 2013

Nick Henderson, ICME

Koichi Murakami, KEK

Collaboration

Geant4 @ SLAC

Geant4 @ KEK

ICME
INSTITUTE for COMPUTATIONAL & MATHEMATICAL ENGINEERING at STANFORD UNIVERSITY

NVIDIA CUDA

Special thanks to the CUDA Center of Excellence Program

Makoto Asai, SLAC

Joseph Perl, SLAC

Koichi Murakami, KEK

Takashi Sasaki, KEK

Margot Gerritsen, ICME

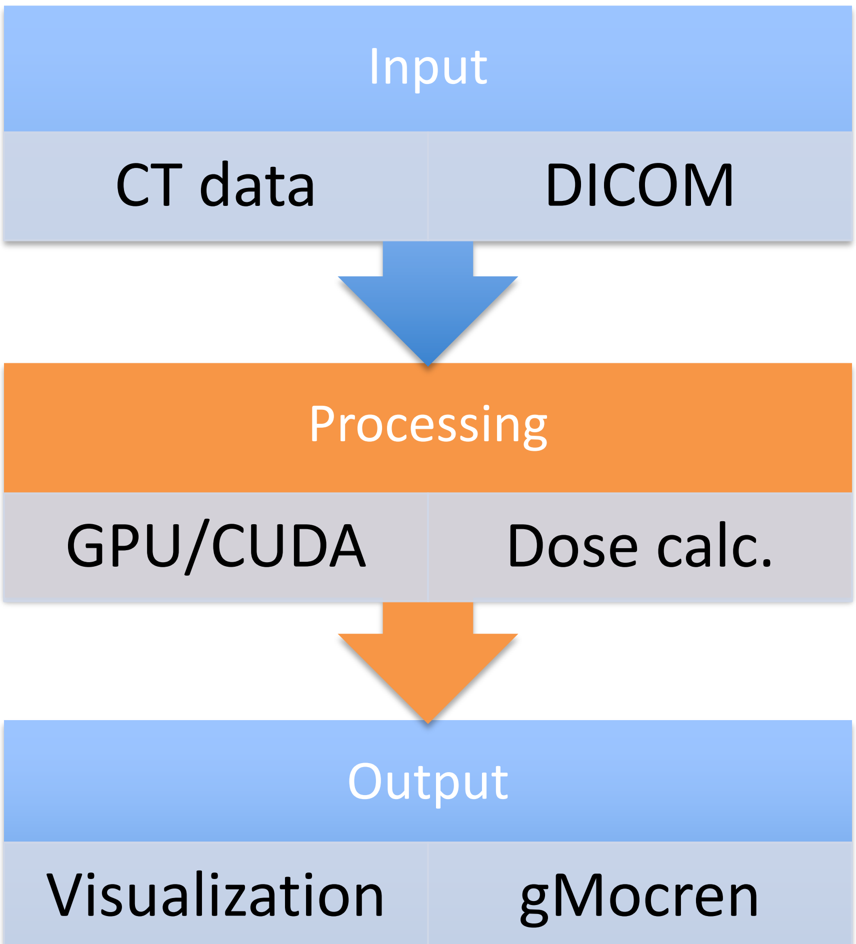
Nick Henderson, ICME

Outline

- Project overview
- Method / algorithm
- GPU implementation
- Performance and validation

Project overview

GPU based dose calculation for radiation therapy



Features

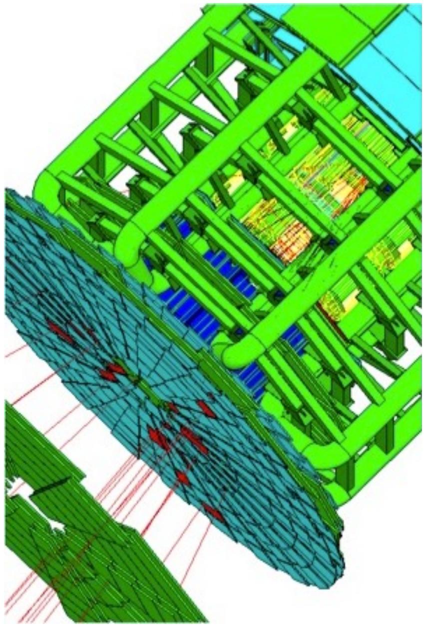
- GPU code based on Geant4
- Voxelized geometry
- DICOM interface
- Material is water with variable density
- Limited Geant4 EM physics: electron/positron/gamma
- Scoring of dose in each voxel
- Process secondary particles

Geant4 Toolkit

- Enables Monte Carlo simulation of particles travelling through and interacting with matter
- Allows modeling of complex geometries
- Covers all elementary particles and nuclei for a wide energy range

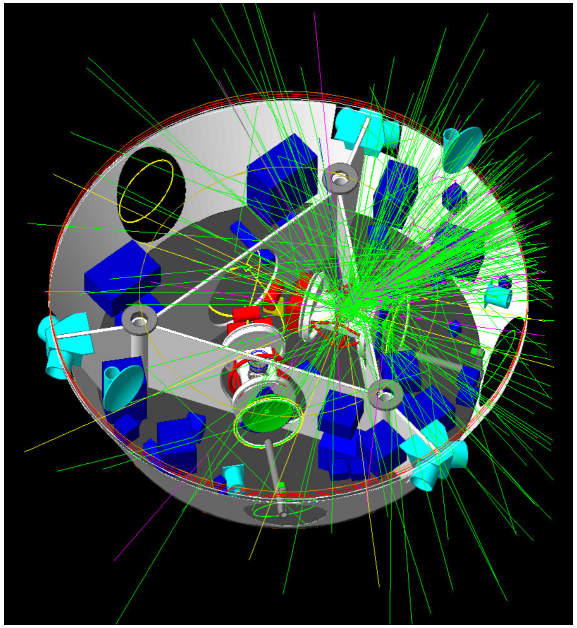
Geant4 Applications

High Energy Physics



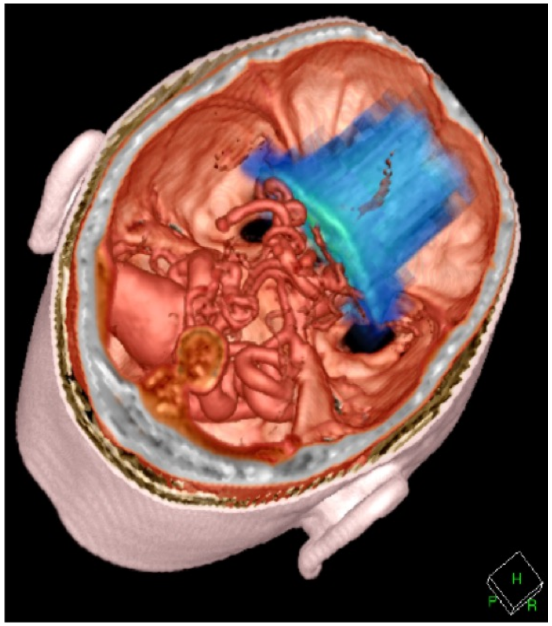
ATLAS

Space & Radiation



LISA

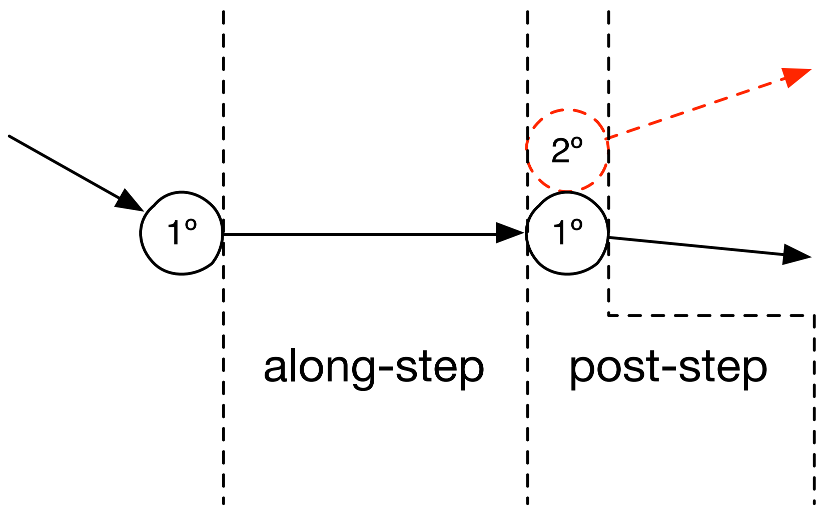
Medical Physics



gMocren

Images from: [Geant4 gallery](#)and [gMocren](#)

Geant4 algorithm summary



Each step can have:

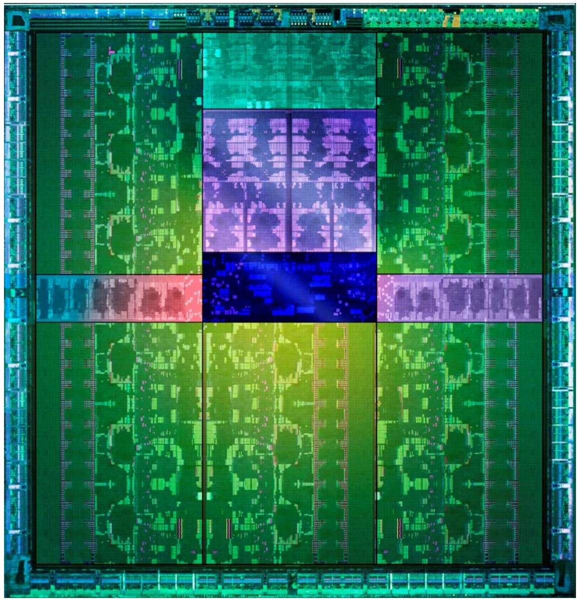
- along-step process
  - energy loss
  - multiple scattering
- post-step process
  - Compton scattering
  - Photoelectric effect
  - (several others)
  - May generate secondary particles

Parallelization challenges in Geant4

- Large and complex code base
  - object-oriented design
  - many inter-dependencies
- Sophisticated geometry and tracking management
- Elaborate physics models
- Branching, look-up tables, single-thread optimizations

GPU implementation strategy

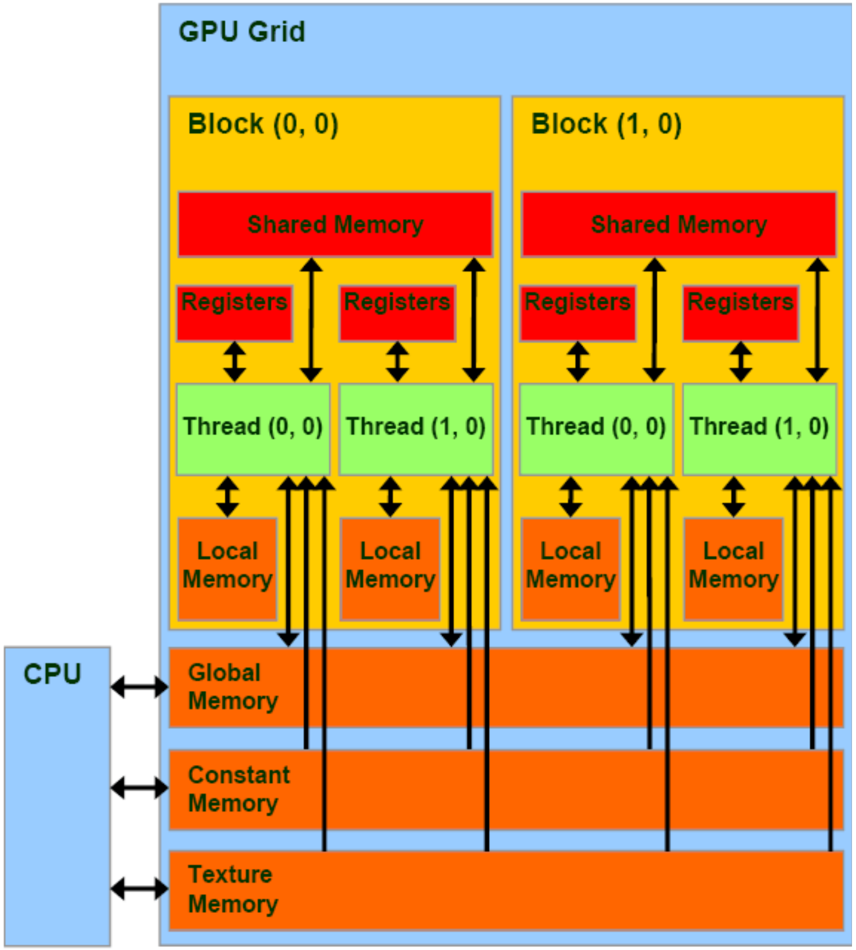
- Process many particles in parallel
- Limited geometry & material
  - voxelized region contained in a world volume
  - material is modeled as water with variable density
- Limited physics
  - low energy electro-magnetics
  - electron, positron, gamma
- Limited scoring
  - record energy dose in each voxel



NVidia Kepler

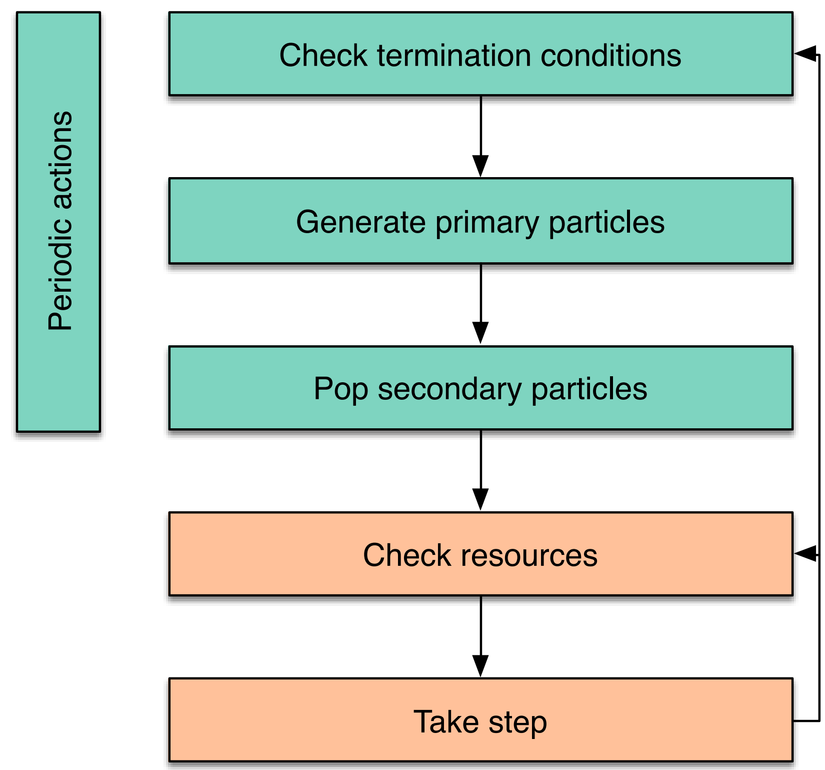
CUDA Basics

- "SIMD" architecture
  - ideal: same instruction on multiple pieces of data
- Memory hierarchy
  - global > cache > register
  - cache: {shared, constant, texture}
- Memory access pattern
  - want single read to satisfy many threads
- Race conditions
  - arise when multiple threads attempt to write to same location in global memory
  - may happen in dose accumulation
  - CUDA supports atomic operations

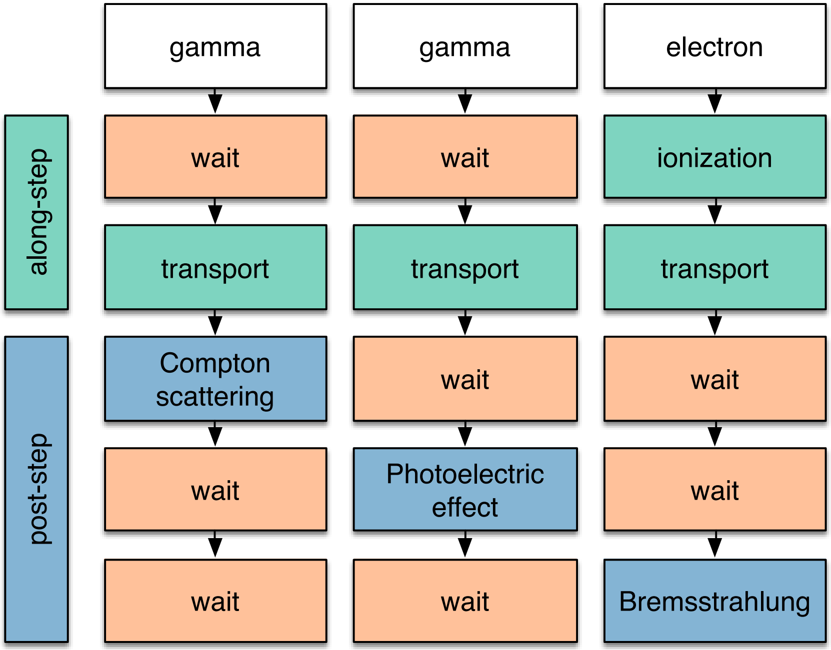


G4CU: CUDA implementation

- Each GPU thread processes a single track until the track exits the geometry
- Each thread has a stack for secondary particles
- Every thread takes a step in each iteration
- Algorithm is periodically interrupted for various actions



G4CU: parallel execution



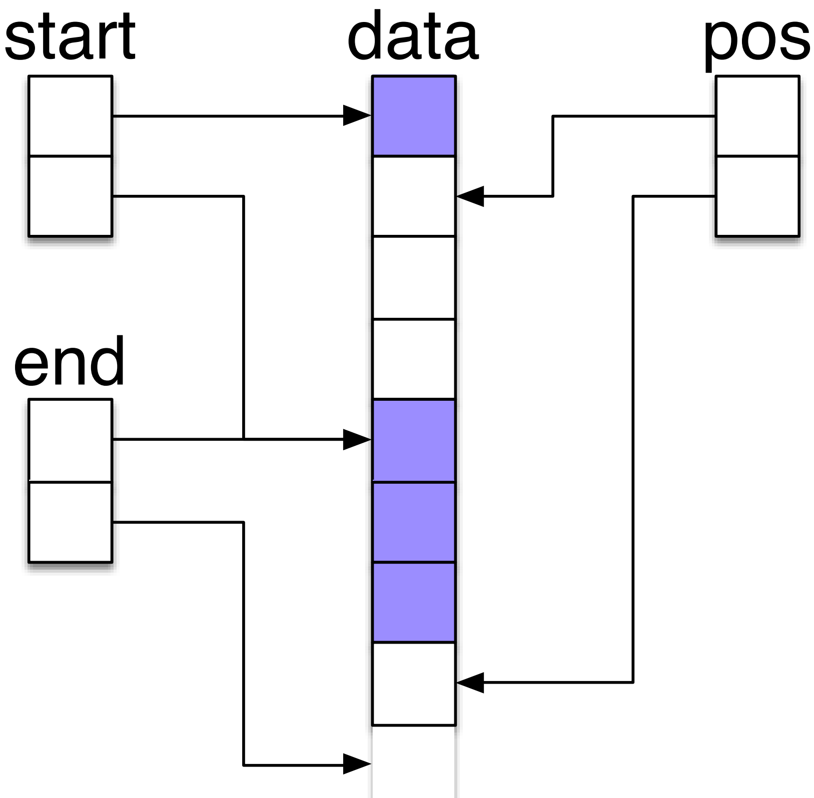
- CUDA kernels are applied to all particles
- Threads must wait if physics process does not apply to particle
- Parallel execution is achieved:
  - maintenance operations
  - transport process
  - same process on same particle

G4CU: struct of arrays

```
struct ParticleArray {
  // length of arrays
  int length;
  // kind of particle
  ParticleKind *kind;
  // position
  float *x, *y, *z;
  // direction
  float *dx, *dy, *dz;
  // particle energy
  float *energy;
};
```

- Common pattern in CUDA to allow for coalesced memory access
- Experiments with transport showed this to be 3-4x faster than an array of structs

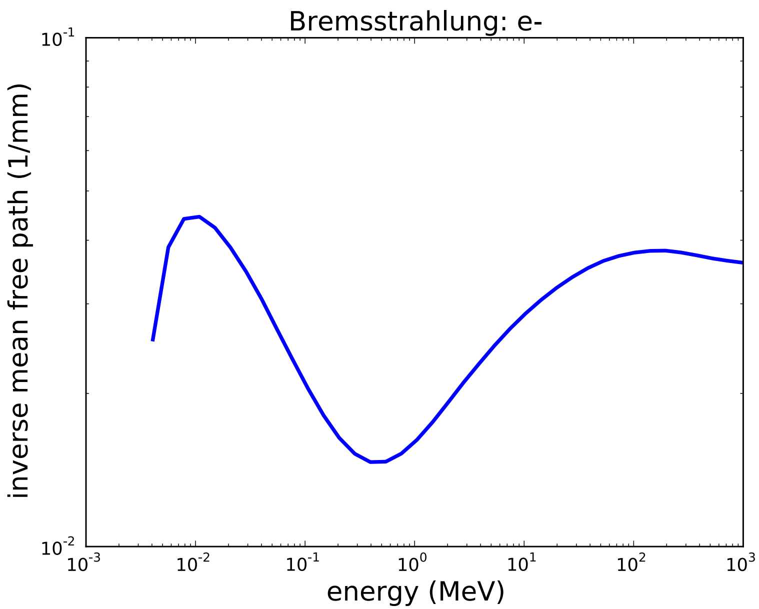
G4CU: parallel stack




```
template <typename T>
struct pstack {
   // number of stacks
   int stack_num;
   // size of each stack
   int stack_size;
   // stack ranges
   int *start, *end;
   // stack positions
   int *pos;
   // stack data array
   T *data;
};
```

G4CU: look-up tables

- Used for cross sections, dE/dx, (inverse-)ranges
  - 40 bins, log spaced (1keV – 1GeV)
  - linear and spline interpolation
  - retrieve data from G4 for a certain cut (1mm)
  - prepared for standard water
- Other tables: Bremsstrahlung uses 2D (surface) interpolation



G4CU: dose accumulation

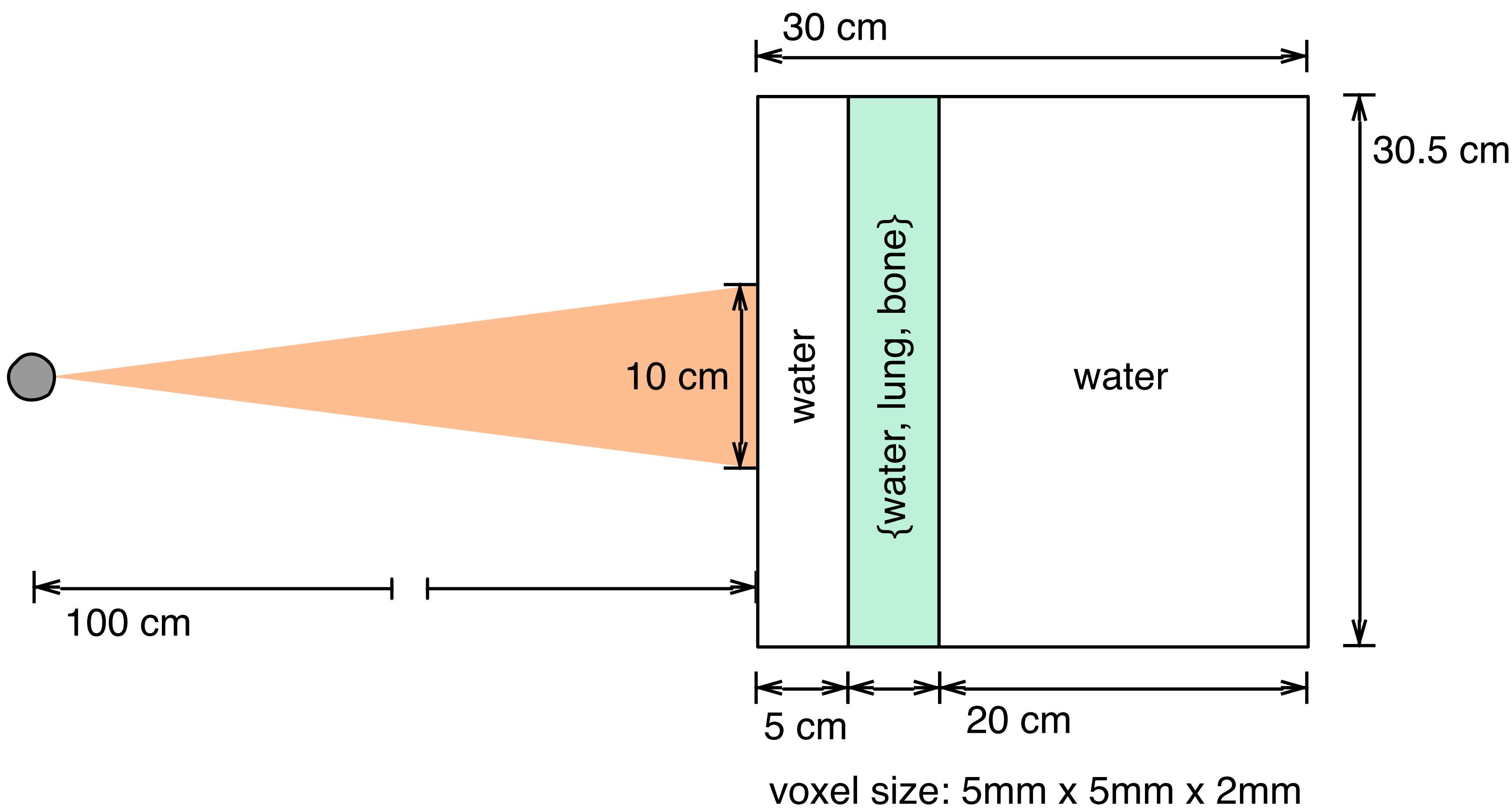
- Race conditions might arise when multiple CUDA threads attempt to write to the same location in global memory. We use CUDA's atomicAdd function
- Double precision is used for dose accumulation
  - single precision for everything else
  - prevents overflow in the case of small energy deposition to large accumulated dose

Performance and validation

Hardware and software:

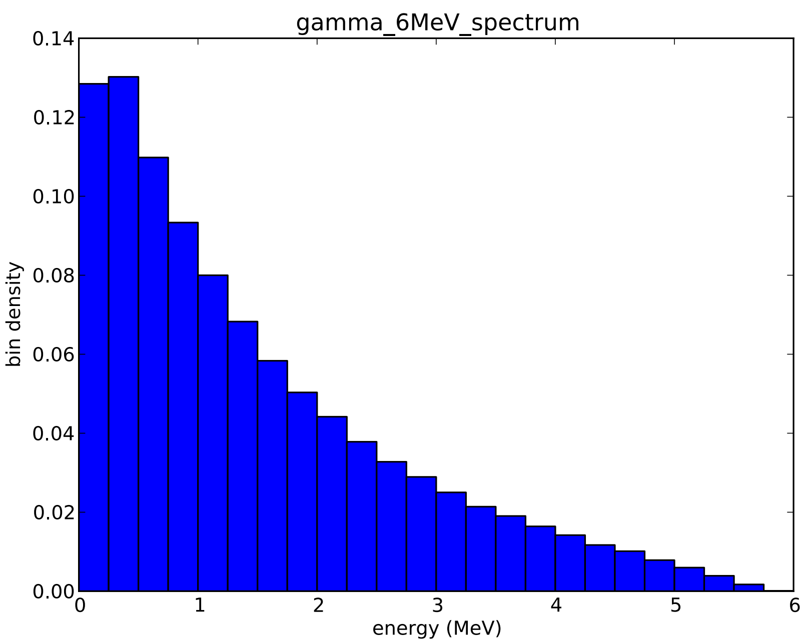
- GPU:
  - Tesla K20 Kepler
  - 2496 cores, 706 MHz, 5GB GDDR5 (ECC)
- CPU
  - Xeon X5680 (3.33GHz)
  - single thread operation
- SDK:
  - CUDA 5.5
  - Geant4 release 9.6 p2

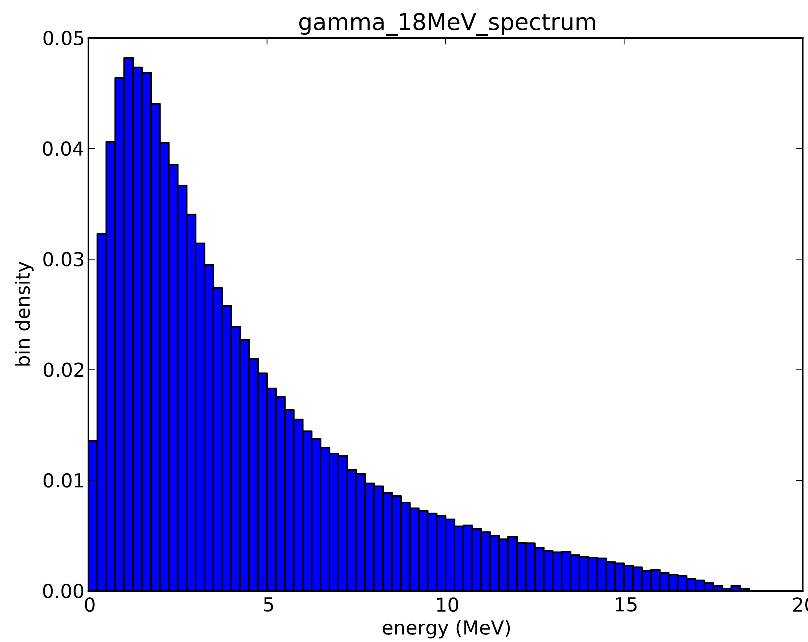
Phantom geometry



Particle sources

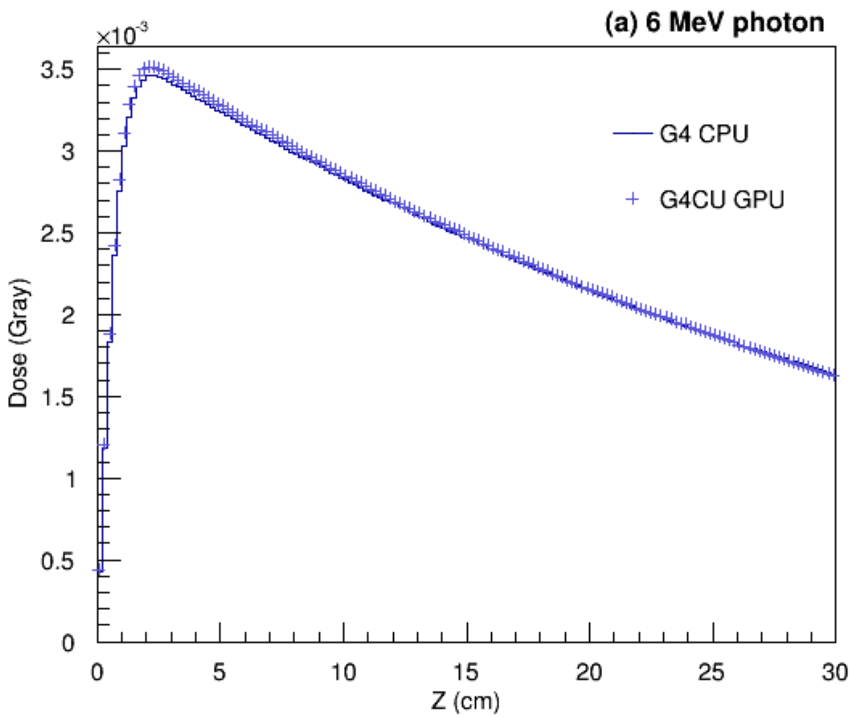
- 20 MeV electron / 6 MeV photons (mono-energy)
- 6 MV & 18 MV spectrum photons

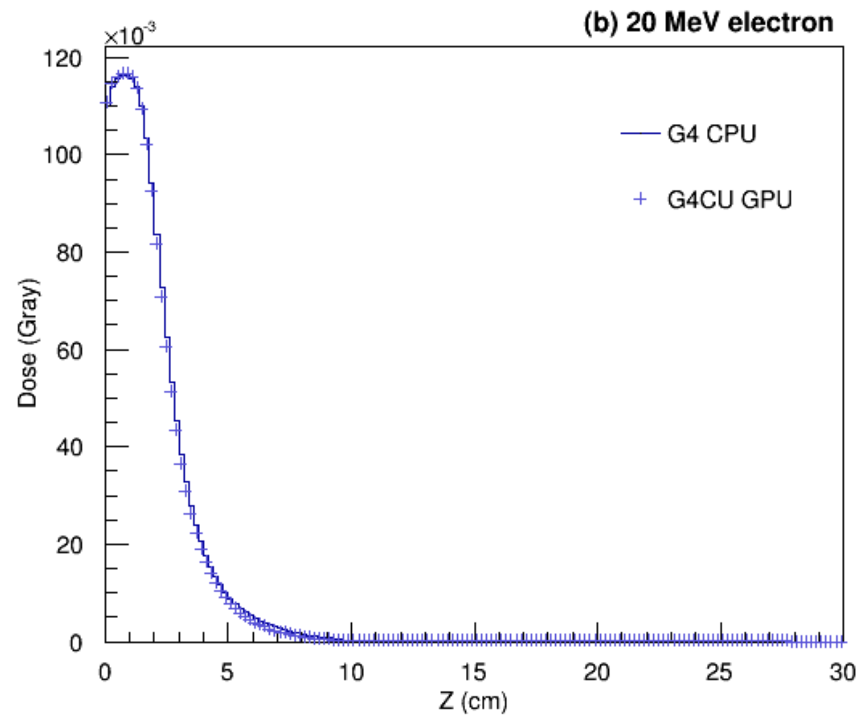




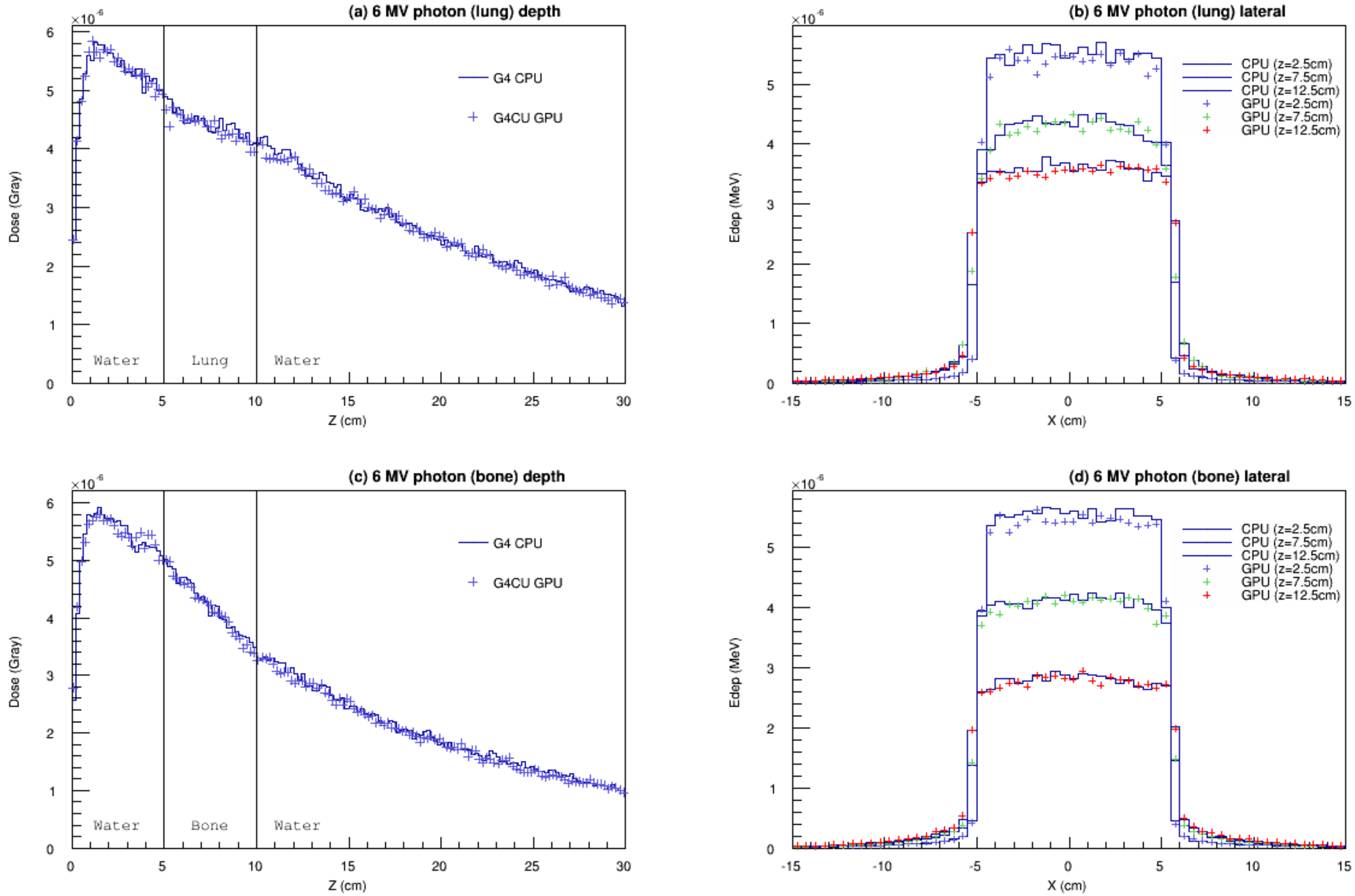
Depth dose distribution: water phantom

- 6MeV photon, 20 MeV electron
- dose along central axis





Comparisons for slab phantoms



6MV photon, Lung/Bone as inner slab material

Profile

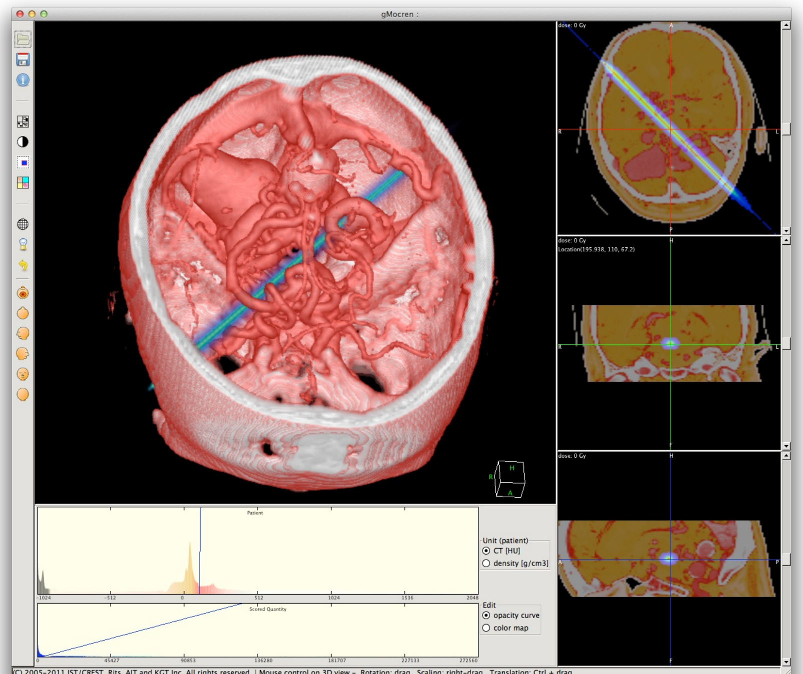
Process	Process total	Post step	PIL	Along step	Manag.	Init.	Step length	At-rest
Component total	**100.00**	**52.80**	**20.73**	**14.16**	**4.19**	**3.78**	**3.46**	**0.89**
Bremsstrahlung	**32.81**	29.83	2.23			0.75		
Ionization	**14.83**	1.70	3.50	8.84		0.79		
Photo-electric effect	**10.79**	8.80	1.57			0.41		
Gamma conversion	**10.67**	8.72	1.54			0.41		
Multiple scattering	**10.50**		3.43	4.58			2.49	
Transport	**8.67**	1.17	5.23	0.74		0.57	0.96	
Compton scattering	**4.20**	2.14	1.58			0.48		
Management	**4.19**				4.19			
Pair production	**2.56**	0.44	1.23			0.36		0.53
Electron deletion	**0.79**		0.43					0.36

Computation time comparison

Primary	Phantom	Time/History CPU (sec)	Time/History GPU (sec)	CPU/GPU
20 MeV electron (pencil)	Water	1.06E-03	2.52E-05	42.1
20 MeV electron (beam)	Lung	1.20E-03	2.67E-05	44.9
20 MeV electron (beam)	Bone	9.76E-4	2.54E-05	38.4
6 MeV photon (pencil)	Water	4.47E-04	1.12E-05	39.9
6 MV photon (beam)	Lung	3.52E-04	9.16E-06	38.4
6 MV photon (beam)	Bone	3.59E-04	9.00E-06	39.9
18 MV photon (beam)	Lung	4.05E-04	1.12E-05	36.2
18 MV photon (beam)	Bone	4.29E-04	1.17E-05	36.7

Observed GPU speed up over CPU: ~40x

Visualization with gMocren



- Image for demonstration purposes only!
- 50M 6MeV photons
- Pencil beam configuration

Future

- Performance improvement
  - texture memory usage for interpolation
  - different stack management for each particle type
- Physics performance
  - more detailed / statistical comparison and verification
  - energy deposition and msc, esp. for lower energy region
- Code design
  - dynamic physics table generation
  - class packaging
  - interfacing with peripheral components

Acknowledgements

- NVidia for support of the CUDA Center of Excellence
- Koichi Murakami
- Makoto Asai, Joseph Pearl, Andrea Dotti @ SLAC
- Takashi Sasaki @ KEK
- Akinori Kimura, Ashikaga Institute of Technology