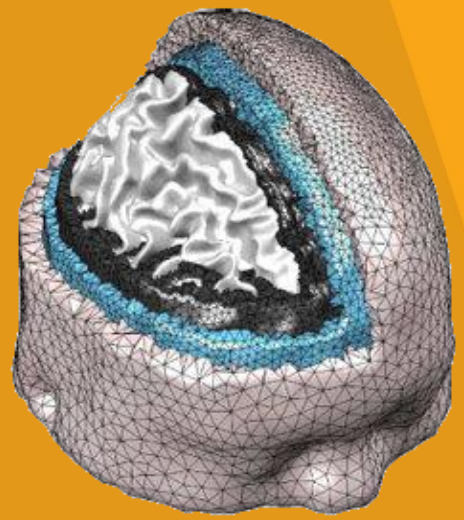
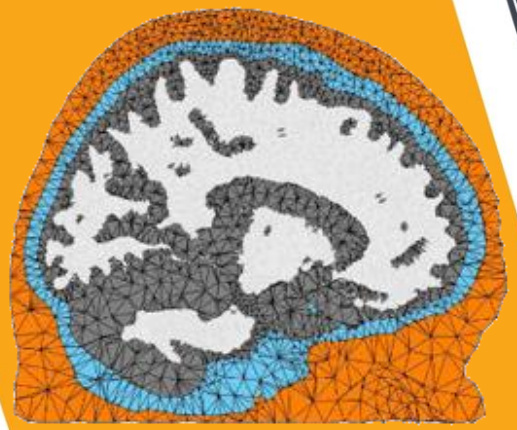
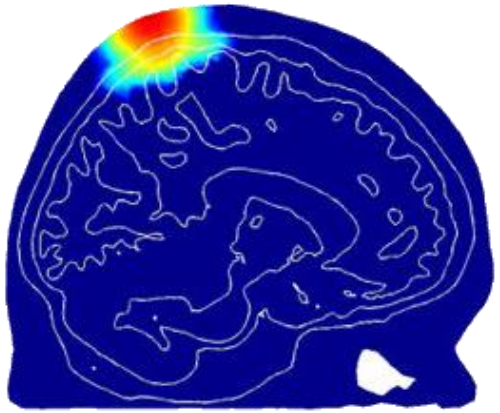






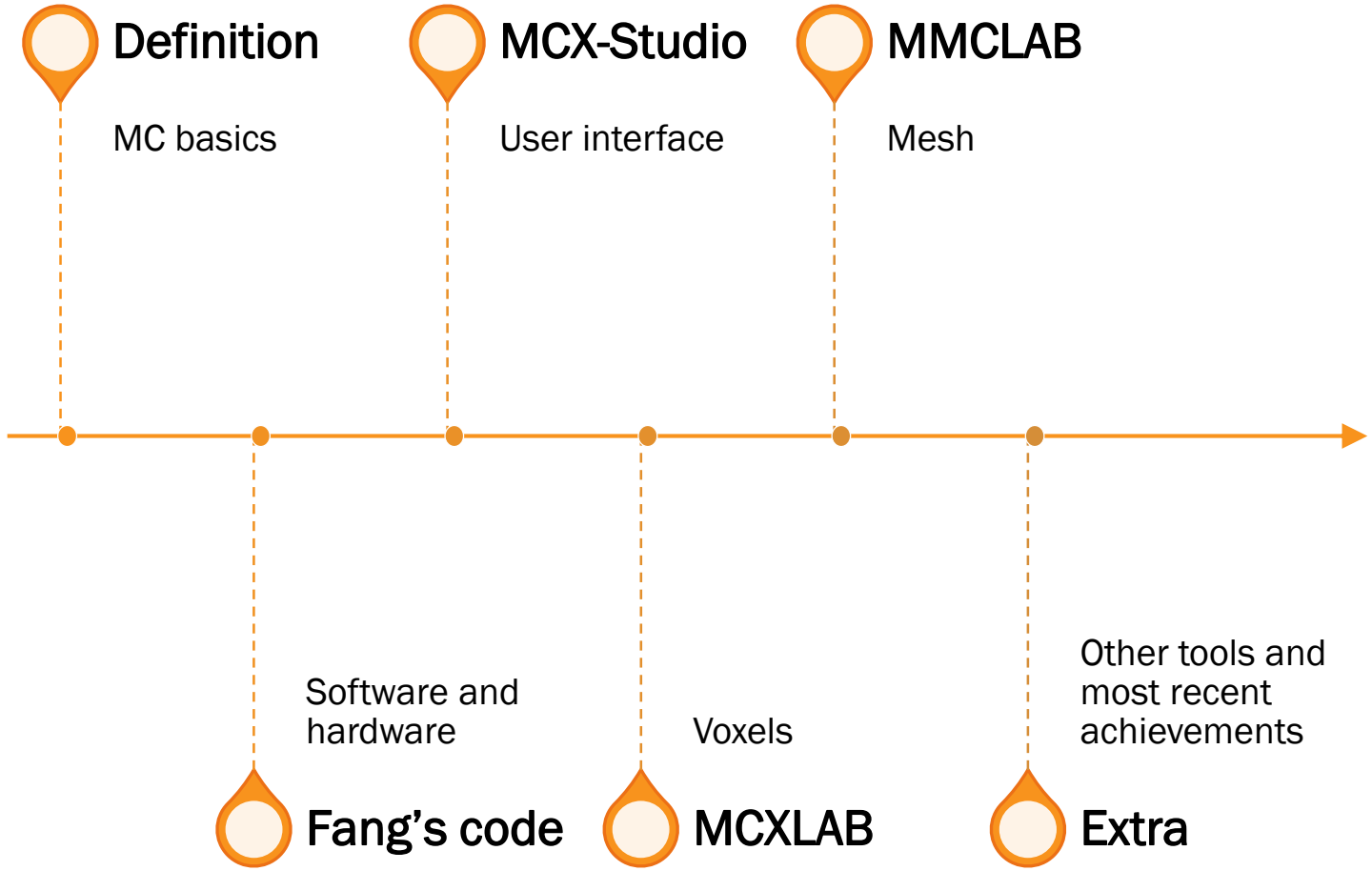
Monte Carlo introduction

16th December 2021

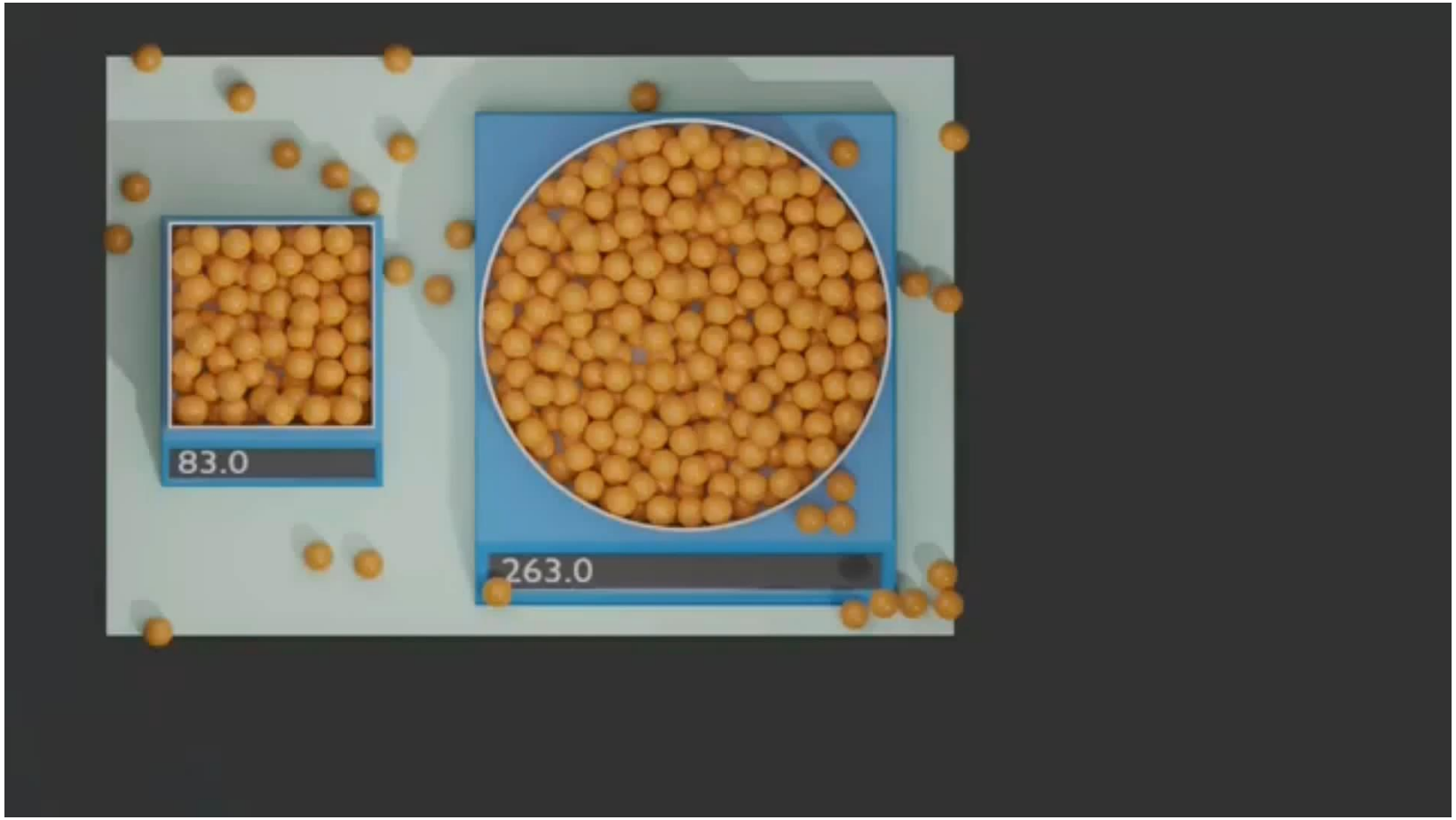
CATERINA AMENDOLA

GIULIA MAFFEIS

Outline

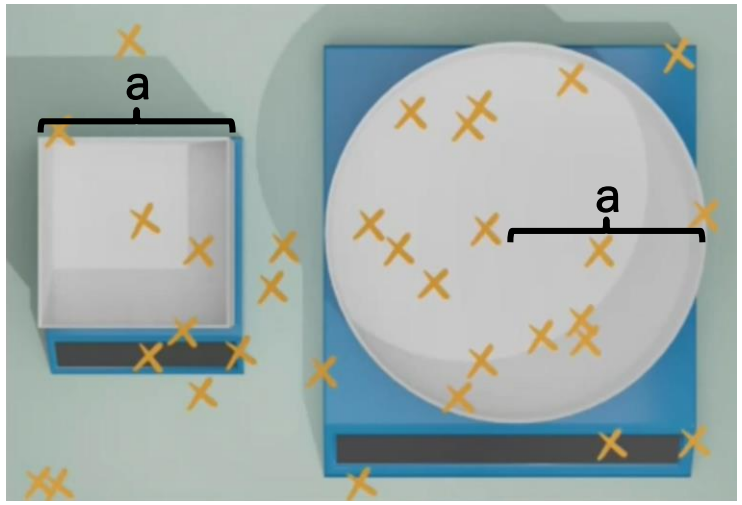

Definition
MC basics
MCX-Studio
User interface
MMCLAB
Mesh
Software and hardware
Fang's code
Voxels
MCXLAB
Other tools and most recent achievements
Extra

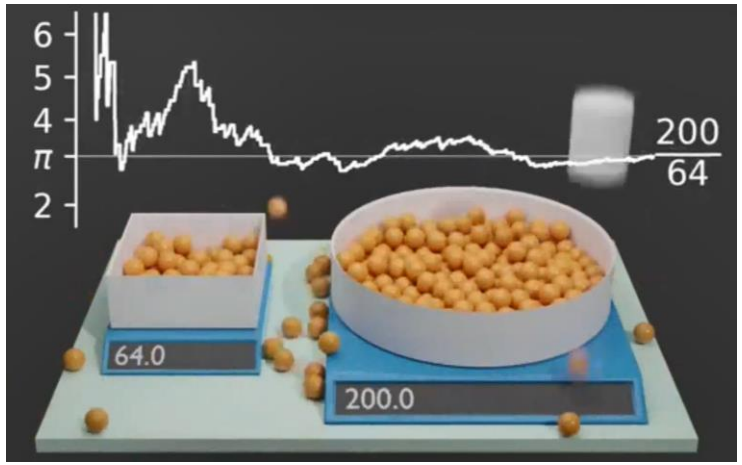
Definition



https://www.youtube.com/watch?v=7ESK5SaP-bc&t=47s&ab_channel=MarbleScience

Definition

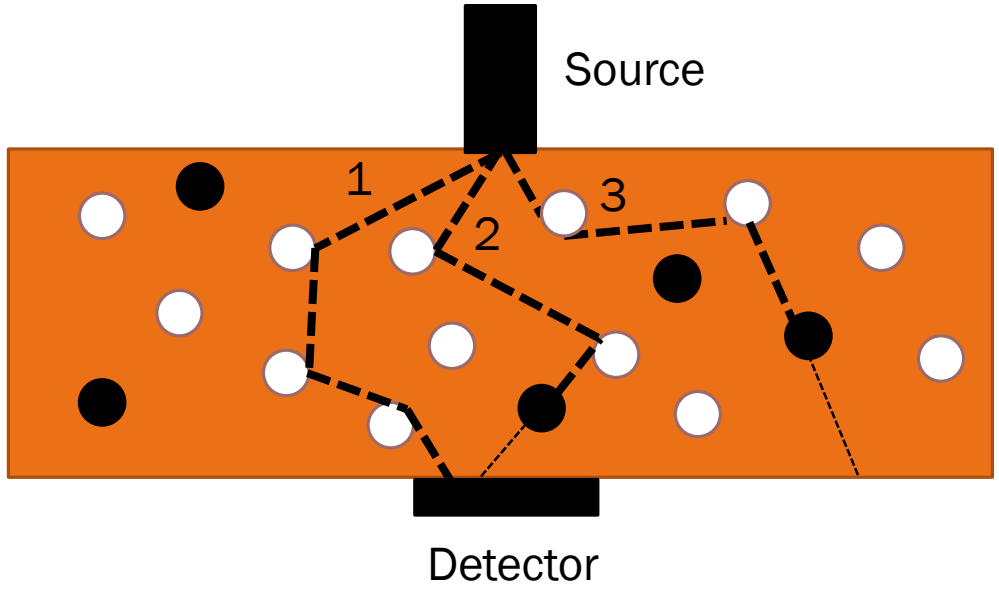




- Randomness
- Sparcity
- Finiteness

Fang's code

Open-source light transport simulator



Scattering centre

Absorption centre

SEED

Initial number of the **pseudorandom number generator.** It returns a sequence of numbers that looks random, but **always generates the same sequence for a given value**.

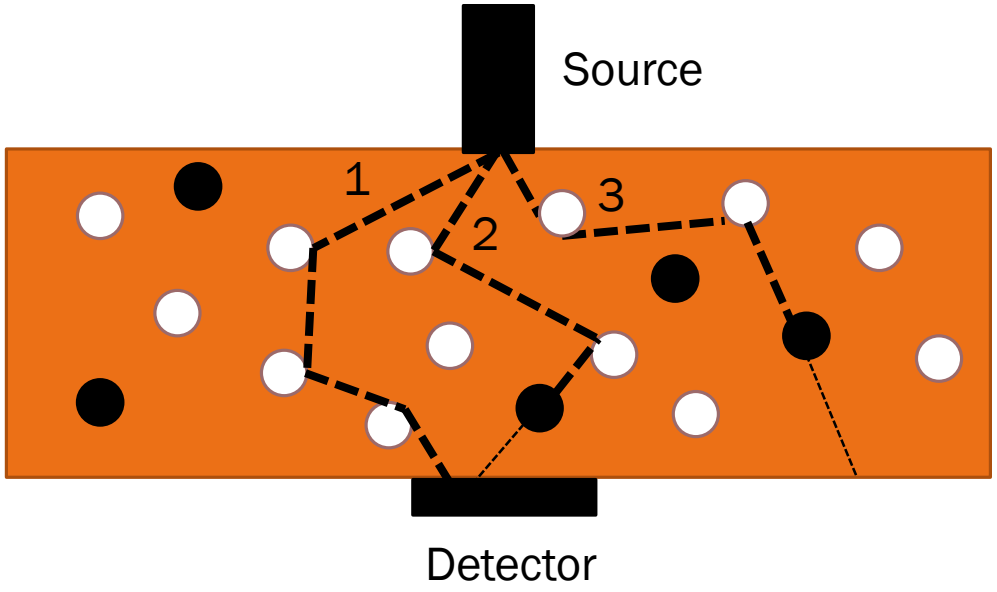
WEIGHT

Initial	For each photon, W = 1
Final	Affected by absorption

http://mcx.space/

Fang's code

Open-source light transport simulator



Scattering centre

Absorption centre

PROPAGATION

Scattering — Deviation in direction

Absorption — Reduction in photon weight

ATTENTION! No annihilation!

DETECTION

In — Collected photon (1, 2)

Out — Lost photon (3)

Final curve = Weight distribution!

Fang's code

WHY DID WE CHOOSE IT?

- Above all, MCX is open-source and free for everyone!
- Time-resolved photon transport simulations
- Supporting arbitrarily complex 3-D volume using a voxelated/meshed domains
- Supporting various boundary conditions, source forms and media (also continuously varying)
- Recording rich set of detected photon information (fluence, energy, jacobian…)
- MATLAB/GNU Octave/JSON/ Python languages
- Supporting GPU acceleration using a single or multiple GPUs (

→ NVIDIA GeForce RTX 3080 Ti on Eulero and Keplero!

→ ATTENTION! Always monitor GPU updates!

Fang's code

GENERAL INFORMATION

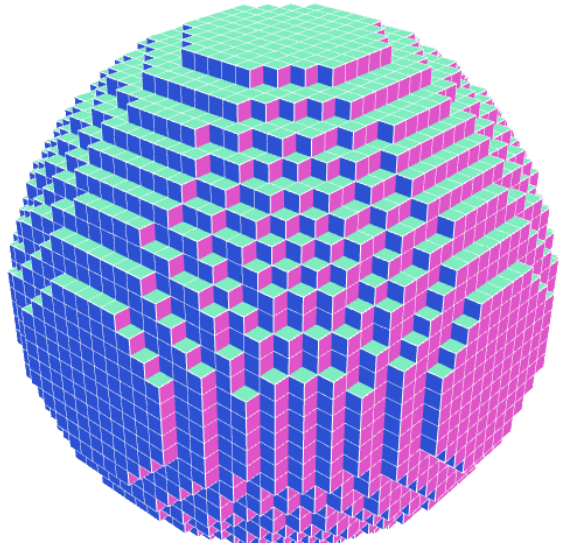
- Web page: http://mcx.space/wiki/index.cgi?Home
- Download the last version of MCX/MMC/MCXSTUDIO: http://mcx.space/wiki/index.cgi?Get
- Tutorials and installation guide: http://mcx.space/wiki/index.cgi?Learn#mcxstudio
- Iso2mesh: http://iso2mesh.sourceforge.net/cgi-bin/index.cgi
- Forum MCX USERS: https://groups.google.com/g/mcx-users?pli=1
- Speed information: http://mcx.space/wiki/index.cgi?Speed

Fang's code

Voxel – MCX

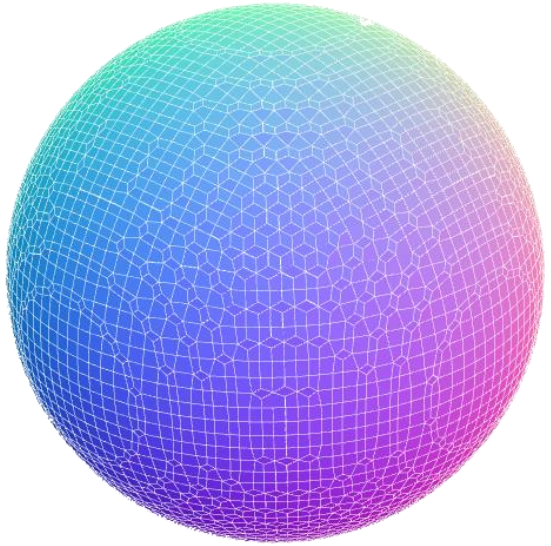
Monte Carlo eXtreme

The volume is discretised in cubes of same dimension and shape



Mesh - MMC

Mesh-based Monte Carlo



The volume is discretised in tetrahedrons of different dimensions and shapes

http://mcx.space/wiki/?Learn

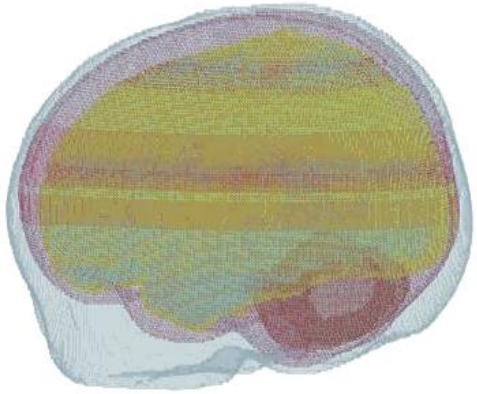
Fang's code

Voxel – MCX

Monte Carlo eXtreme

PRO
- Fast → **GPU!**
- Easy to implement

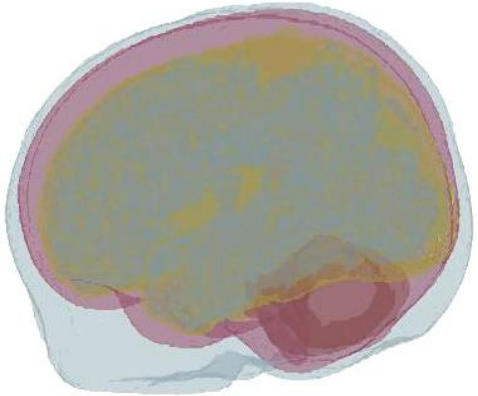
CON
- Lack of smoothness



2 min

Mesh - MMC

Mesh-based Monte Carlo

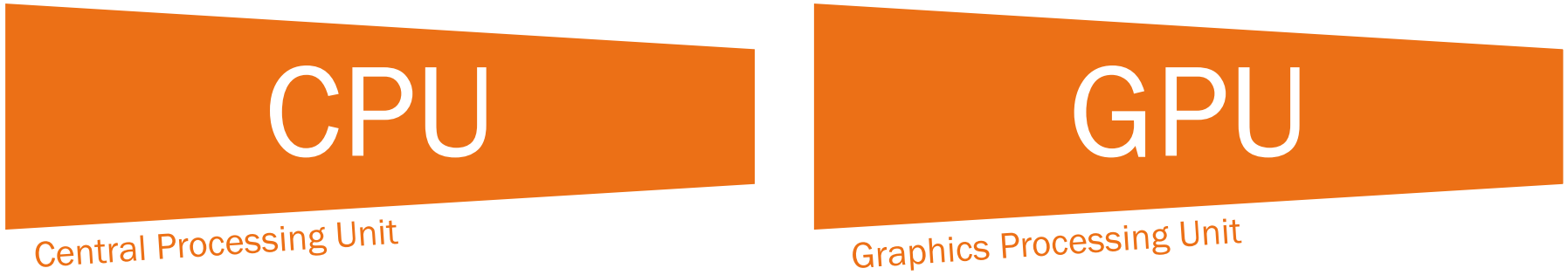


31 min

PRO
- Accurate
- Smooth surfaces

CON
- Long computational time → **CPU!**

Fang's code



Optimized for latency	Optimized for throughput
Mainly serial approach	Parallel approach
Versatile	Specific tasks
Smart	Repetitive tasks
Few units	Thousands of units
Es: PC "brain"	Es: machine learning, cryptocurrency mining, videogaming, scientific computation

Fang's code

OpenCL (Open Computing Language) is a framework for writing programs that execute across heterogeneous platforms, *e.g.* CPUs, GPUs, DSPs, FPGAs. It provides a standard interface for parallel computing using task- and data-based parallelism.

MMCCL

MMC for OpenCL

PRO

- Accurate
- Smooth surfaces
- Faster than MMC → **CPU + GPU!**

CON

- Longer than MCX
- NVIDIA does not fully support OpenCL

MCXCL

MCX for OpenCL

PRO

- Unlike MCX (only NVIDIA GPUs), MCX-CL can be executed on up-to-date CPUs and GPUs

CON

- Longer than MCX
- NVIDIA does not fully support OpenCL

Voxel: MCX Studio

Graphical **user interface** for MCX.

Example #1: Head layered structure

- Trhree-layer cylinder: D = 60 mm, H_1 = 5 mm, H_2 = 10 mm, H_3 = 45 mm

	Background (0)	Layer (1)	Layer (2)	Layer (3)
mua [mm^{-1}]	0	0.01	0.02	0.03
mus [mm^{-1}]	0	4.55	9.09	13.64
g	1	0.89	0.89	0.89
n	1	1.4	1.3	1.4

- Reflectance geometry: unit given by unitinmm = 1

	x	y	z	R
Source	32.5	30	0	Pencil beam
Detector	27.5	30	0	2

Interfiber distance = 5 mm

- Nphoton: 10^7
- Tstart: 0
- Tstep: 25 ps
- Tend: 2 ns
- Seed: random
- Single detector, r=2 mm
- Outputtype: fluence and photon path

http://mcx.space/wiki/?Learn#mcxstudio
http://mcx.space/wiki/index.cgi?Demo/MCXStudio

Voxel: MCX Studio

10 minutes

DIY #1: Tumor in compressed breast

- Cube (6x6x4 cm^3) with a central spherical inclusion (r = 1 cm)

	Background (0)	Cube (1)	Sphere (2)
mua [mm^{-1}]	0	0.02	0.05
mus [mm^{-1}]	0	4.55	9.09
g	1	0.89	0.89
n	1	1.40	1.40

- Reflectance geometry: unit given by unitinmm = 1

	x	y	z	R
Source	20	30	0	Pencil beam
Detector	40	30	0	2

- Nphoton: 10^7
- Tstart: 0
- Tstep: 25 ps
- Tend: 2 ns
- Seed: random
- Single detector, r=2 mm
- Outputtype: fluence and photon path

Voxel: MCX Studio

HOW TO ACCESS RESULTS IN MATLAB

- Add the path `'...\MCXStudio\MCXSuite\mcx\utils'`
- Use the function `'loadmch.m'` to load data about detected photons → Useful mainly for MCX Studio!

USEFUL FUNCTIONS FOR DATA ELABORATION

- `'mcxdettime.m'`: to determine the time of flight of collected photons
- `'mcxdetweight.m'`: to determine the weight of collected photons
- `'mcxdettpsf.m'`: to reconstruct the output Time Point Spread Function (TPSF)

→ ATTENTION: BUG!

```
detp.ppath=detp.ppath(detp.detid==detnum,:);
detp.detid=detp.detid(detp.detid==detnum);
detp.w0=detp.w0(detp.detid==detnum);
```
→ MISSING IN OLD SOFTWARE VERSIONS!

Voxel: MCXLAB

MCXLAB is a **voxel-based** photon simulator based on Matlab.

REQUIRED

- `cfg.nphoton`: the total number of photons to be simulated (integer)
- `cfg.unitinmm`: defines the length unit for a grid edge length
- `cfg.vol`: a 3D array specifying the media index in the domain.

→ ATTENTION! Each subdomain is identified with a tag!

- `cfg.prop`: [mua, mus, g, n]

→ ATTENTION! mm^{-1}! mus, NOT musp!

- `cfg.tstart, cfg.tstep, cfg.tend`: time-resolved curve features

SOURCE-DETECTOR PARAMETERS

- `cfg.detpos`: [x, y, z, radius]
- `cfg.srcpos, cfg.srcdir, cfg.srctype`: source position, direction and type

MC SIMULATION SETTINGS

- `cfg.seed`: seed for the random number generator (integer)

GPU SETTINGS

- `cfg.autopilot`: 1-automatically set threads and blocks

OUTPUT

- `cfg.outputtype`: 'flux' - fluence-rate, (default value)

  'fluence' - fluence integrated over time,

  'energy' - energy deposit per voxel

  'jacobian' or 'wl' - mua Jacobian

  'nscat' or 'wp' - weighted scattering counts

http://mcx.space/wiki/index.cgi?Doc/MCXLAB

Voxel: MCXLAB

TIPS & TRICKS

Computational time

Output signal level

- Launch **many simulations with a limited number of input photons**, rather than a single simulation with a large amount of photons
- Launch **"absorption-free" simulations** and add mua *a posteriori*
- Take advantage of domain **symmetry** to create many **detectors** and increase output signal level

Example #2: Again, trilayer cylinder (Example #1)

BUT

with central source and ring detector (r = 1mm)

→ Try same/different seed

→ Try with/without absorption

→ Plot TPSF

FUNCTIONS

- `mcxpreview:` domain plot
- `mcxlab:` launch simulation

MATRIX DIMENSIONS

- Output: # collected photons, included "absorbed" ones

  → **ATTENTION!**
  Max #collected photons by default = 1e6!!!

https://groups.google.com/g/mcx-users/c/jhBo-JMZdyg/m/Da-1VwQLBgAJ
https://www.osapublishing.org/josaa/abstract.cfm?uri=josaa-29-10-2110

Voxel: MCXLAB

15 minutes

DIY #2: Tumour in compressed breast

- Cube (xyz 6x6x4 cm^3) with a central spherical inclusion (r = 1 cm)

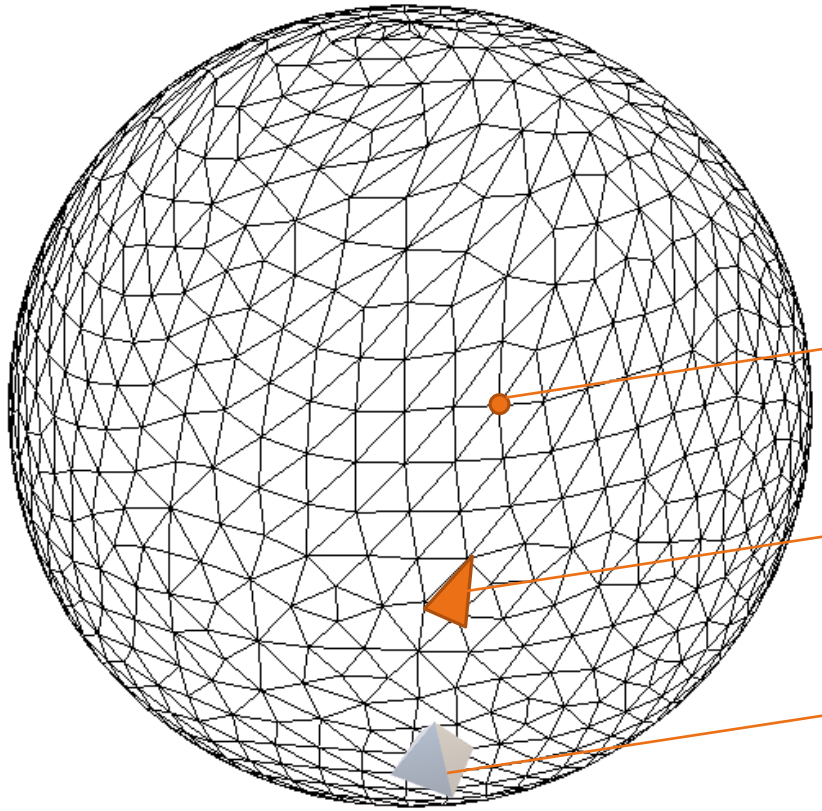
	Background (0)	Cube (1)	Sphere (2)
mua [mm^{-1}]	0	0.01	0.02
musp [mm^{-1}]	0	1.0	1.5
g	1	0.89	0.89
n	1	1.4	1.4

- Unit given in mm
- Transmittance geometry (source and detector aligned along z axis)
- Enable reflection at exterior boundary
- Enable reflection at interior boundary too
- Measure elapsed time (tic toc)
- Plot: fluence isolines (10 levels) at yz plane (x = X/2)

- Nphoton: 10^8
- Tstart: 0
- Tstep: 80 ps
- Tend: 14 ns
- Seed: random
- Detector area: 8 mm^2
- GPU autopilot: 1
- Outputtype: fluence

Mesh: MMCLAB

MMCLAB utilizes a **tetrahedral mesh** to model complex anatomical structures.



Node: intersection of edges

Face: triangular surface

Element: the tetrahedron

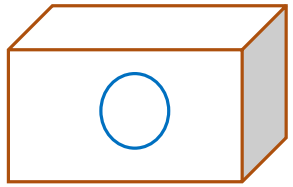
http://mcx.space/wiki/index.cgi?Learn#mmclab
http://mcx.space/wiki/index.cgi?MMC/Doc/MMCLAB

Mesh: MMCLAB - Iso2mesh

`iso2mesh` is a free Matlab/Octave-based mesh generation and processing toolbox.
It can create 3D tetrahedral finite element (FE) mesh from surfaces, 3D binary and gray-scale volumetric images such as segmented MRI/CT scans.

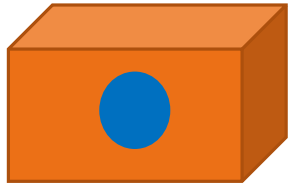
- `[node,face,elem]=meshabox(p0,p1,opt,nodesize)`

  To create the surface and tetrahedral mesh of a box geometry



- `[newnode,newelem]=mergemesh(node,elem,varargin)`

  To concatenate two or more tetrahedral meshes or triangular surfaces



- `[node,elem,face]=surf2mesh(v,f,p0,p1,keepratio,maxvol,regions,holes,forcebox)`

  To create quality volumetric mesh from isosurface patches

http://iso2mesh.sourceforge.net/cgi-bin/index.cgi?Doc/FunctionList

Mesh: MMCLAB

Example #3: Tumor in compressed breast

- Cube (xyz 6x6x4 cm^3) with spherical inclusion (r = 1 cm)

	Background (0)	Cube (1)	Sphere(2)
mua [mm^{-1}]	0	0.01	0.02
musp [mm^{-1}]	0	1.0	1.5
g	1	0.89	0.89
n	1	1.4	1.3

- Reflectance geometry: unit given by unitinmm

	x	y	z	R
Source	25	30	0	Pencil beam
Detector	35	30	2	2

- Nphoton: 10^8
- Tstart: 0
- Tstep: 25 ps
- Tend: 5 ns
- Detector: r = 4 mm
- Seed: random
- Autopilot: 1
- Outputtype: TPSF, fluence

Mesh: MMCLAB

DIY #3: AT HOME

- Bilayer cylinder: R = 60 mm, H_1 = 5 mm, H_2 = 30 mm

	Background (0)	Layer (1)	Layer (2)
mua [mm^{-1}]	0	0.01	0.02
musp [mm^{-1}]	0	1.0	1.5
g	1	0.89	0.89
n	1	1.4	1.4

- Reflectance geometry: unit given by unitinmm

	x	y	z	R
Source	35	30	0	Pencil beam
Detector	25	30	0	2

- Nphoton: 10^8
- Tstart: 0
- Tstep: 25 ps
- Tend: 2 ns
- Detector: r = 2 mm
- Seed: random
- Autopilot: 1
- Outputtype: fluence

From mesh to voxel

USEFUL FUNCTIONS

- `[img, v2smap] = surf2vol(node,face,xi,yi,zi,varargin)`

  To convert a triangular surface to a shell of voxels in a 3D image

- `[mask weight] = mesh2vol(node,elem,xi,yi,zi)`

  To enable a fast rasterization of a 3D mesh to a volume with tetrahedron index labels

- How to load and plot existing `cfg`:

```
load cfg;
mcxpreview(cfg);
```

Extra

OTHER TOOLS AVAILABLE

- **Photon replay:** It gives the possibility to retrace photon trajectories, obtaining photons' sensitivities (*i.e.* Jacobian) rather then weights.
  [Ruoyang Yao, Xavier Intes, Qianqian Fang, "A direct approach to compute Jacobians for diffuse optical tomography using perturbation Monte Carlo-based photon 'replay'," Biomed. Optics Express 9(10), 4588-4603, (2018)]*

- **iMMC:** It incorporates both mesh- and shape-based tissue representations to create highly complex (*e.g.* dense vessel networks and porous tissues), yet memory-efficient light transport simulations.
  [Yaoshen Yuan, Shijie Yan, and Qianqian Fang, "Light transport modeling in highly complex tissues using the implicit mesh-based Monte Carlo algorithm," Biomed. Optics Express, 12(1), 147-161, (2021)]*

- **SVMC:** Split Voxel MC is a hybrid algorithm combining meshes and voxels by means of the fast marching cubes method to greatly improve accuracy across curved boundaries while remaining highly flexible and efficient in parallel hardware.
  [*Qianqian Fang* and Shijie Yan, "*Hybrid mesh and voxel based Monte Carlo algorithm for accurate and efficient photon transport modeling in complex bio-tissues" J. of Biomedical Optics, 11(11), 6262, 6270 (2020).]*

- **DMMC:** Dual-grid mesh-based Monte Carlo to accelerate photon simulations using a coarsely tessellated tetrahedral mesh for ray-tracing computation and an independent voxelated grid for output data storage.
  [Shijie Yan, Anh Phong Tran, Qianqian Fang, "A dual-grid mesh-based Monte Carlo algorithm for efficient photon transport simulations in complex 3-D media," J. of Biomedical Optics, 24(2), 020503 (2019).]*

http://mcx.space/

Thank you!

16th December 2021

CATERINA AMENDOLA

GIULIA MAFFEIS