An Accelerated Raytracing Method for Radiative Transfer Simulations

Patrick Hirling (EPFL)

SKACH Spring Meeting

June 1, 2023

EPFL

Context

- Epoch of Reionization (EoR, $5 < z < 25$): formation of the first luminous sources whose radiation transitioned the Universe from a cold neutral state to a hot, ionized one

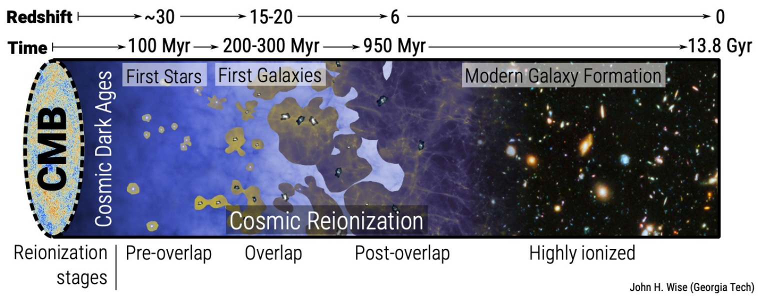
- Appearance and properties of the primordial sources can be probed by observing the 21-cm signal (HI) with radio interferometric telescopes (e.g. SKA-Low, LOFAR)

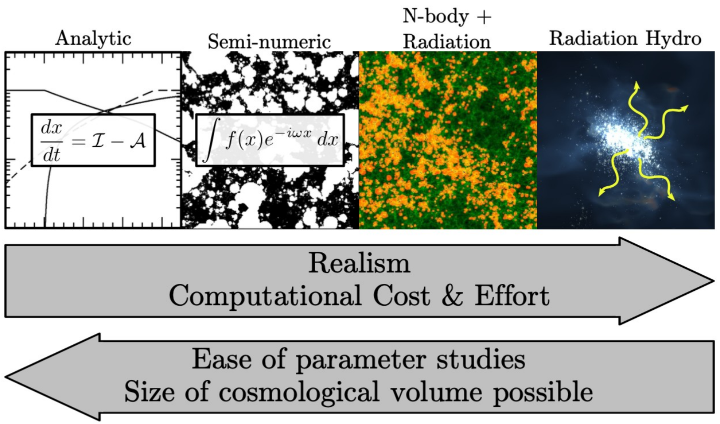
- Modeling the EoR accurately enough for these future observations requires realistic radiative transfer (RT) simulations on scales >100 Mpc with high resolution and hydrodynamic feedback

Fully-numerical codes

Moment Methods
- Fast but diffusive

Raytracing Methods
- Accurate but expensive





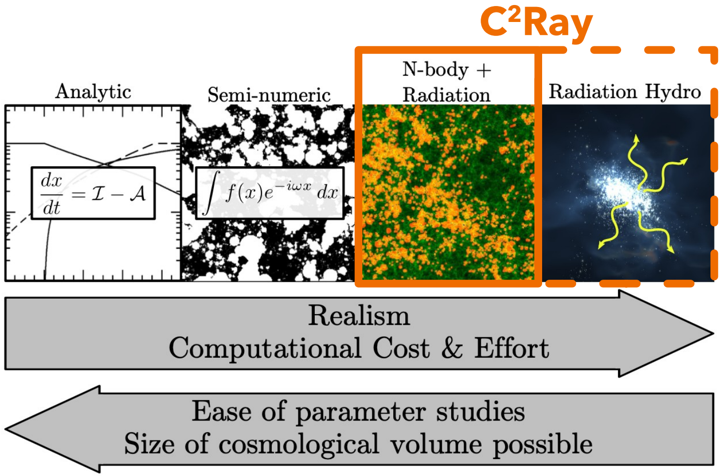
J. Wise (2019) arXiv:1907.06653

C^2Ray

C^2Ray (Mellema et al. 2005) is a **fully-numerical raytracing code** extensively used for large-scale EoR simulations

- **CPU-parallelized** using OpenMP with domain-decomposition & MPI-parallel treatment of sources
- Designed to be eventually coupled to hydrodynamics
- Even without this, a typical EoR simulation takes up to **millions of core hours** on several thousand nodes

Can it be made faster using GPU-parallelization ?



J. Wise (2019) arXiv:1907.06653

Reionization Simulations

Goal: Time evolution of ionized fraction of hydrogen *x* in a physical volume due to photoionization and recombination effects

$$\frac{\mathrm{d}x}{\mathrm{d}t} = (1-x)(\underbrace{\Gamma}_{\text{Photo-ionization}} + \underbrace{n_\mathrm{e} C_H}_{\text{Collisional ionization}}) - \underbrace{x n_\mathrm{e} \alpha_H}_{\text{Recombination}}$$

Γ: Photoionization Rate

Γ is due to ionizing flux from primordial sources:

$$\Gamma_\mathrm{local}(r) = \frac{1}{4\pi r^2} \int_{\nu_\mathrm{th}}^{\infty} \frac{L_\nu \sigma_\nu e^{-\tau_\nu(r)}}{h\nu} \mathrm{d}\nu$$

Optical depth:
$\tau_\nu = \sigma_\nu N_{HI}$

**continuous form, will vary for a discrete grid but retains similar structure*

Reionization Simulations

Goal: Time evolution of ionized fraction of hydrogen *x* in a physical volume due to photoionization and recombination effects

$$\frac{\mathrm{d}x}{\mathrm{d}t} = (1 - x)(\underbrace{\Gamma}_{\text{Photo-ionization}} + \underbrace{n_\mathrm{e} C_H}_{\text{Collisional ionization}}) - \underbrace{x n_\mathrm{e} \alpha_H}_{\text{Recombination}}$$

Γ: Photoionization Rate

Γ is due to ionizing flux from primordial sources:

$$\Gamma_{\mathrm{local}}(r) = \frac{1}{4\pi r^2} \int_{\nu_{\mathrm{th}}}^{\infty} \frac{L_\nu \sigma_\nu e^{-\tau_\nu(r)}}{h\nu} \mathrm{d}\nu$$

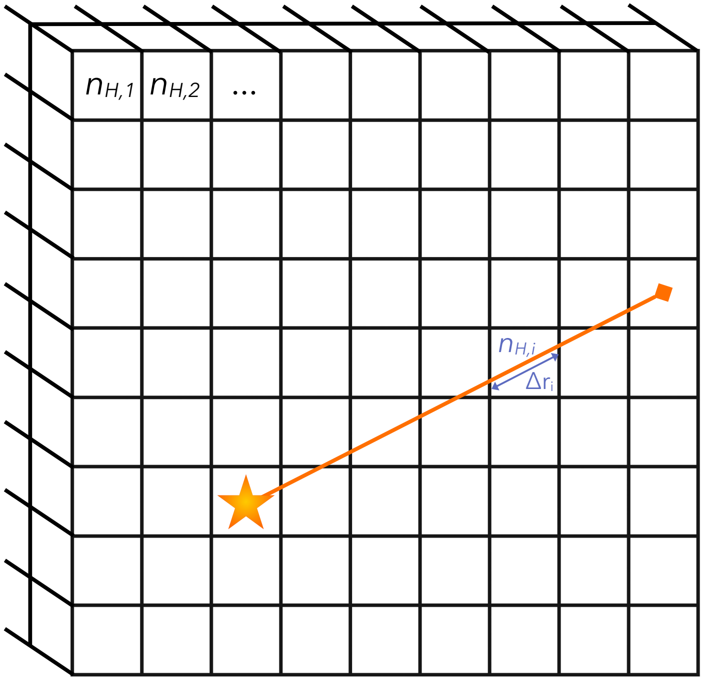
Optical depth:
$\tau_\nu = \sigma_\nu N_{HI}$

Key Quantity: column density of neutral hydrogen N_{HI}

**continuous form, will vary for a discrete grid but retains similar structure*

Raytracing on a grid

- Need the column density of hydrogen between source and all grid points, for possibly many sources



$$N_H = \int_{\mathrm{Ray}} n_H(s)ds = \sum_{\mathrm{Crossed\ Cells}} n_{H,i}\Delta r_i$$

Treating all points (cells) by direct computation is **accurate** but **very expensive**. Several types of raytracing methods exist:

→ Long characteristics (e.g. Feautrier 1964)

→ Short characteristics (e.g. Kunasz & Auer 1988)

→ Adaptive Raytracing (e.g. Abel & Wandelt 2002)

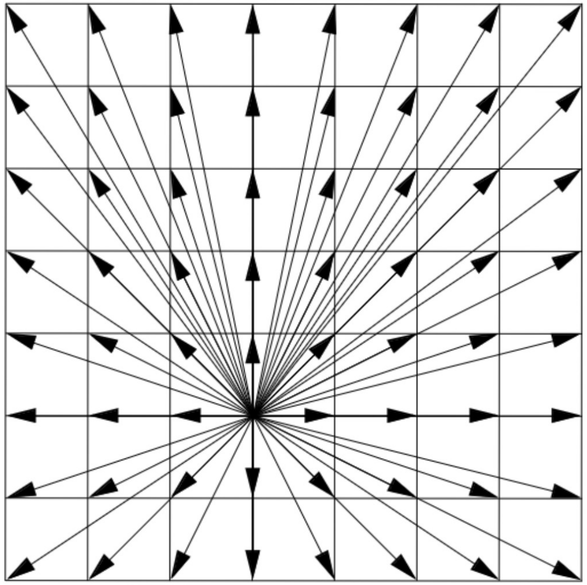
→ Cones (e.g. Petkowa & Springel 2011)

→ Hybrid Methods (e.g. Rijkshorst et al. 2006)

Short vs Long Characteristics

Idea: Compute N_{HI} *up to* a cell by interpolation with *n* neighbours closer to the source, add part *within* the cell by direct computation

Long Characteristics

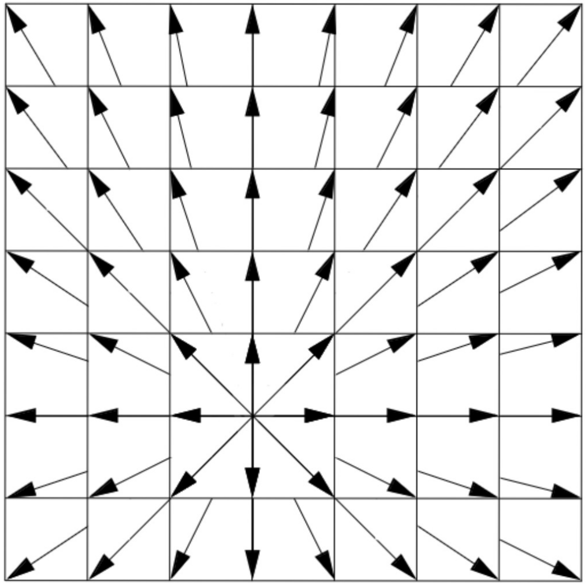


Direct Computation

Accurate

$O(N^4)$

Short Characteristics



Interpolative

More Diffusive

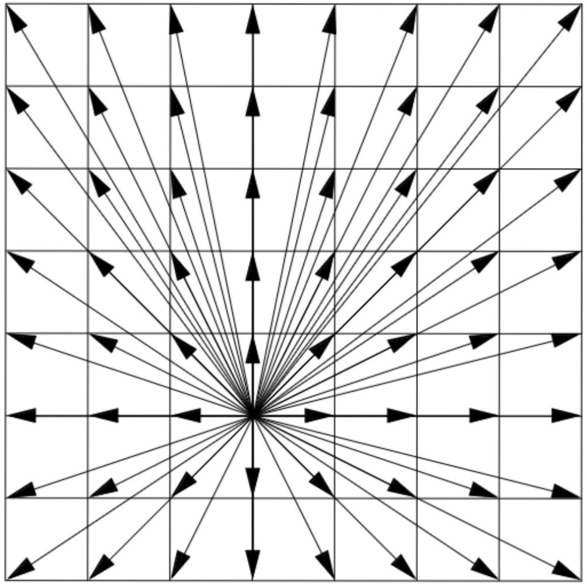
$O(N^3)$

Adapted from Rijkhorst et al. (2006)

Short vs Long Characteristics

Idea: Compute N_{HI} *up to* a cell by interpolation with *n* neighbours closer to the source, add part *within* the cell by direct computation

Long Characteristics

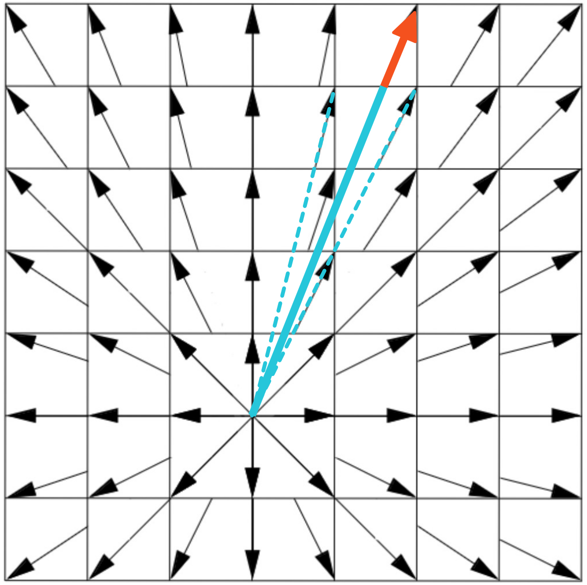


Direct Computation

Accurate

$O(N^4)$

Short Characteristics



Interpolative

More Diffusive

$O(N^3)$

Adapted from Rijkhorst et al. (2006)

Short vs Long Characteristics

Idea: Compute N_{HI} *up to* a cell by interpolation with *n* neighbours closer to the source, add part *within* the cell by direct computation

Interpolative nature of SC methods imposes a causal order in which the grid cells must be treated (need to know (A) before doing (B)

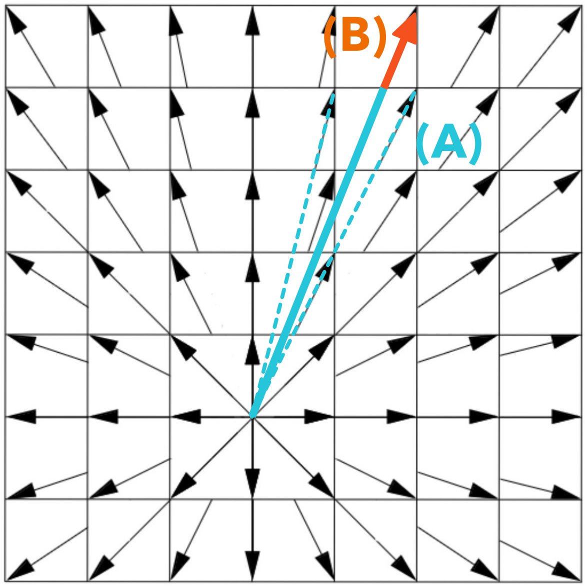
This makes parallelization more challenging

Short Characteristics

Interpolative

More Diffusive

$O(N^3)$



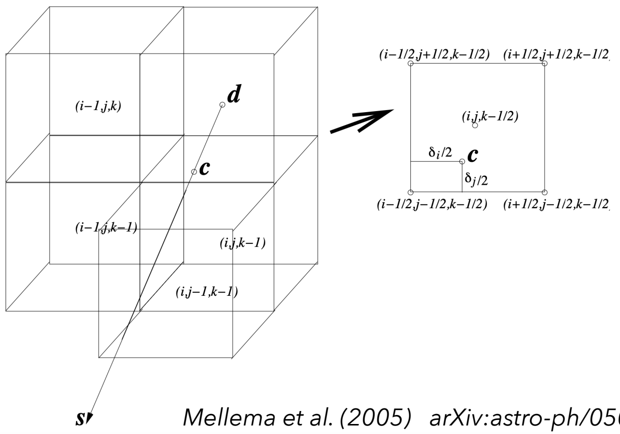
Adapted from Rijkhorst et al. (2006)

Short Characteristics in C^2Ray

C^2Ray uses a version with 4 neighbours (Raga et al. 1999):

$$e_1 = (i, j, k - \sigma_k)$$
$$e_2 = (i, j - \sigma_j, k)$$
$$e_3 = (i - \sigma_i, j, k)$$
$$e_4 = (i - \sigma_i, j - \sigma_j, k - \sigma_k)$$
$$\sigma_k = \mathrm{sign}(k - k_{src})$$

(+ special weighting)

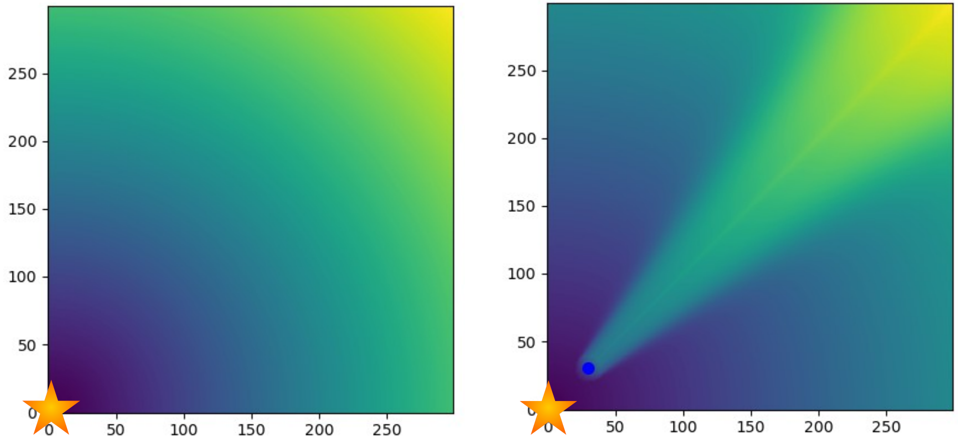


Mellema et al. (2005) arXiv:astro-ph/0508416

- Currently, grid is treated **in serial** in cubic (x → y → z) order *outward from the source*
- How to do it in parallel?

Examples of column density obtained using the 4-point SC method

Uniform Density Field + Dense Clump

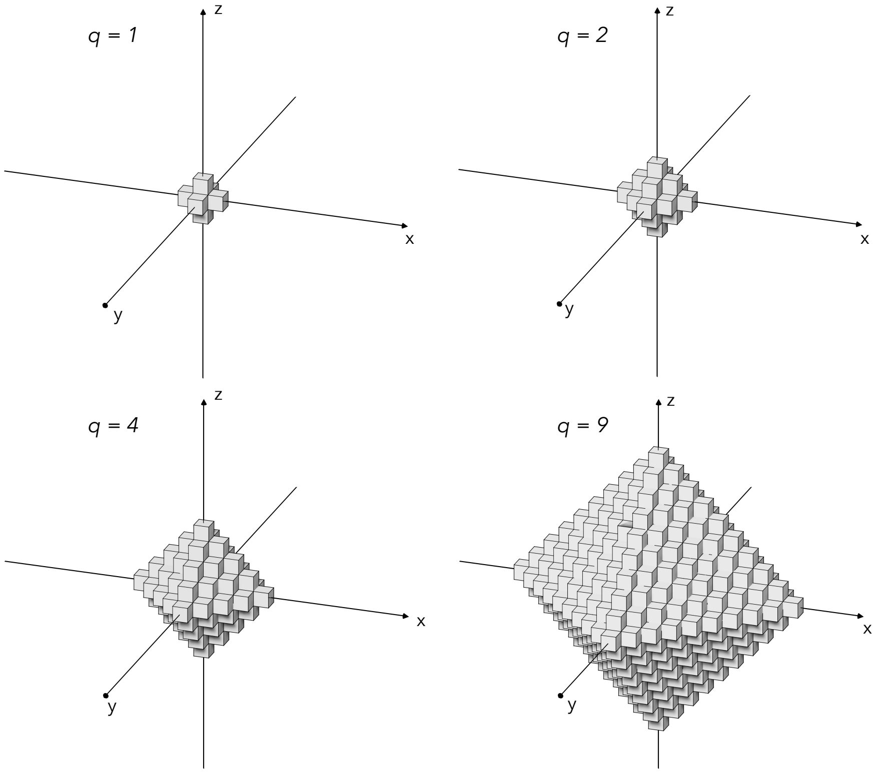


Our Method: *OCTA*

Shape that best follows the geometry of the 4-point method: **octahedral shells**:

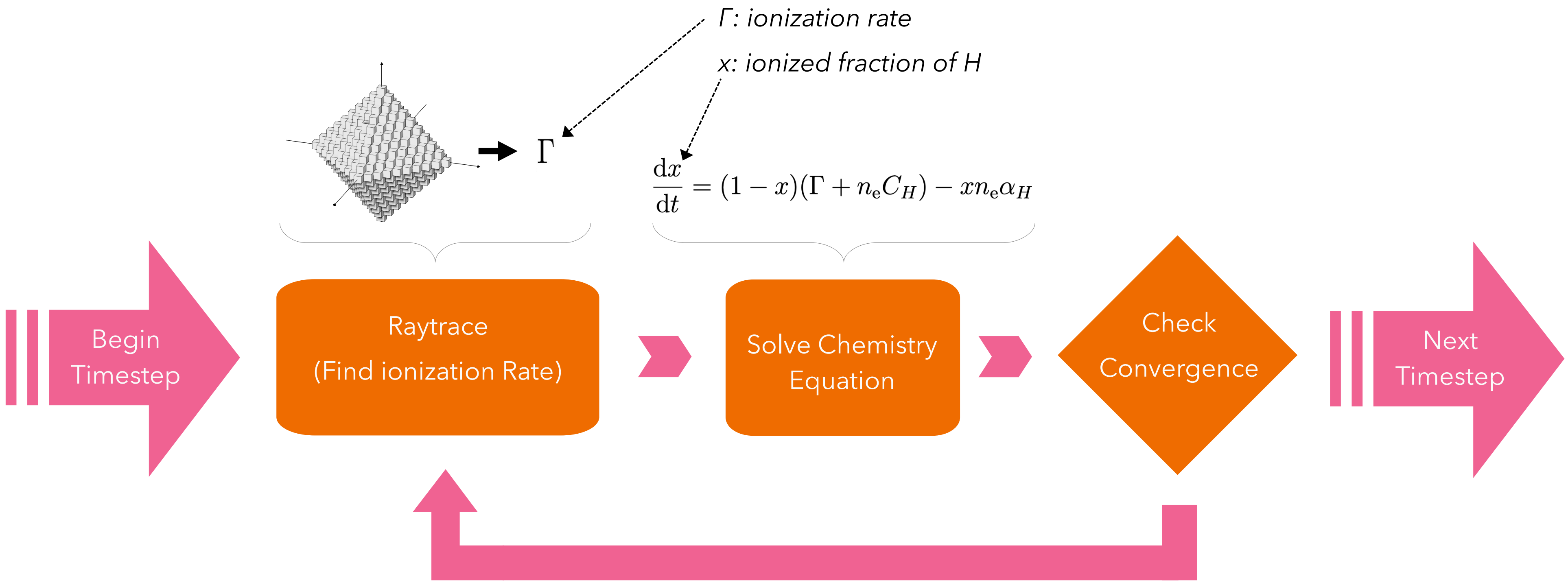
- On a shell, all cells are **independent**
- All 4 interpolants are "below" the current shell (i.e. already calculated)
- At each iteration ~q^2 cells can be treated **in parallel** on a GPU

(Source is at the origin)



...

Integration Into C2Ray



Can be easily adapted for other codes that require raytracing

Accuracy-Testing *C2Ray + OCTA*

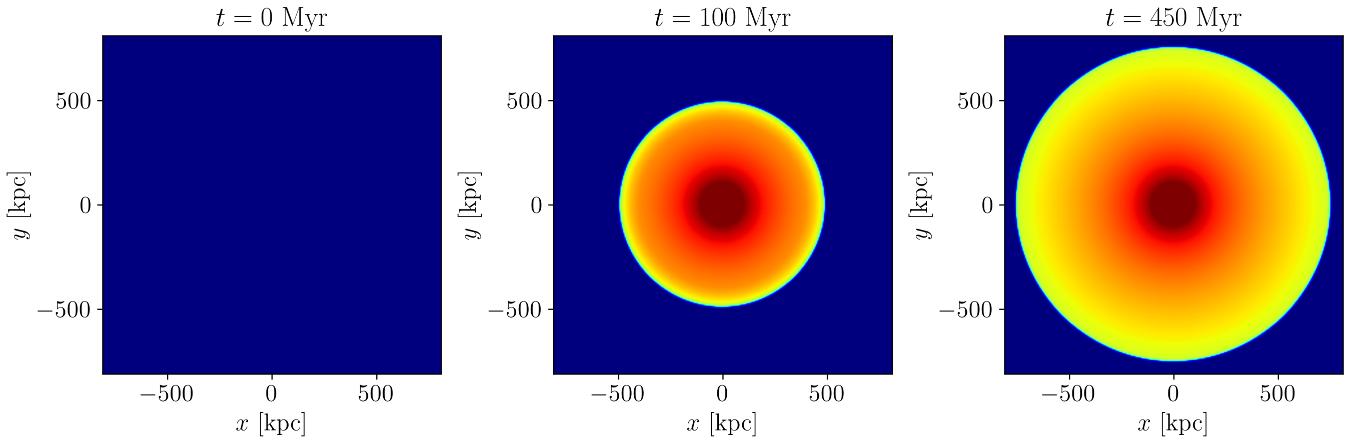
Strömgren Sphere (HII region expansion around single monochromatic source in uniform density field)

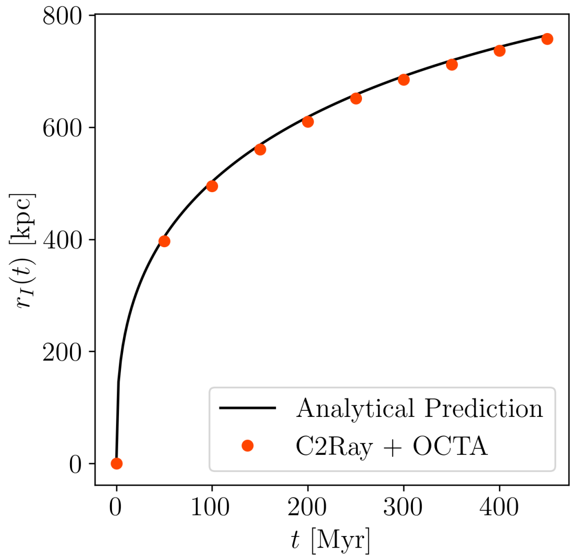
+ Direct comparison with original C2Ray in multi-source cases

$$r_{\mathrm{S}} = \left[\frac{3\dot{N}_\gamma}{4\pi\alpha_B(T)Cn_{\mathrm{H}}^2} \right]^{1/3} \qquad r_I = r_{\mathrm{S}} \left[1 - \exp(-t/t_{\mathrm{rec}})\right]^{1/3}$$

$$t_{\mathrm{rec}} = \left[C\alpha_B(T)n_H\right]^{-1}$$

(Ionized fraction of Hydrogen shown)



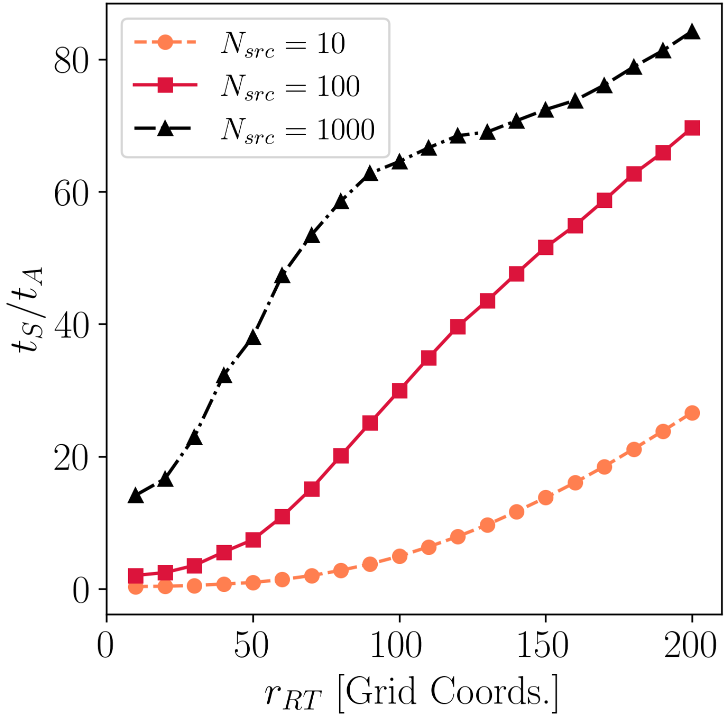


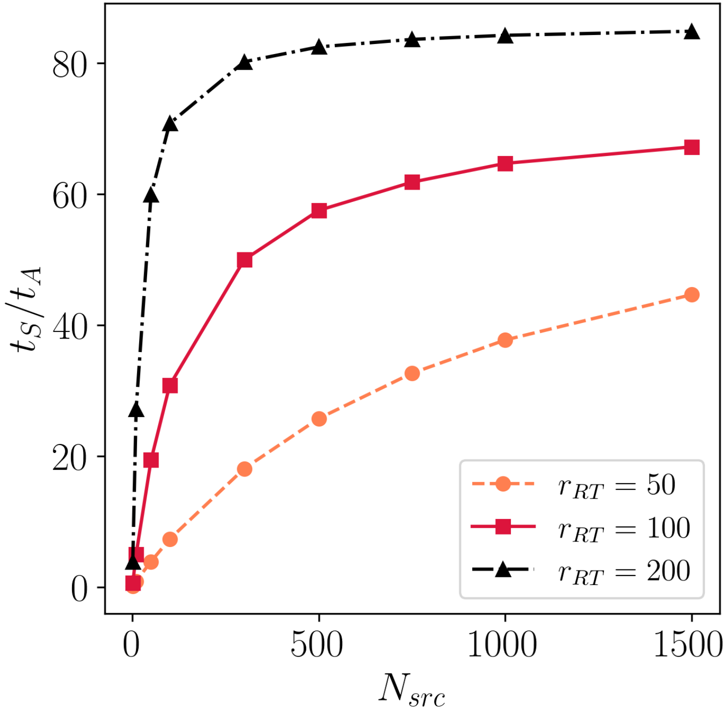
Performance Tests

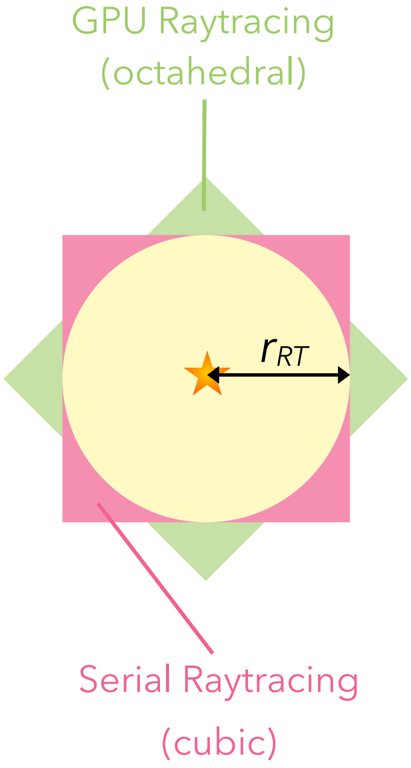
N_{src}: Number of sources randomly distributed in the volume

t_S : Serial execution time using original method

t_A: Accelerated (GPU) execution time using OCTA method



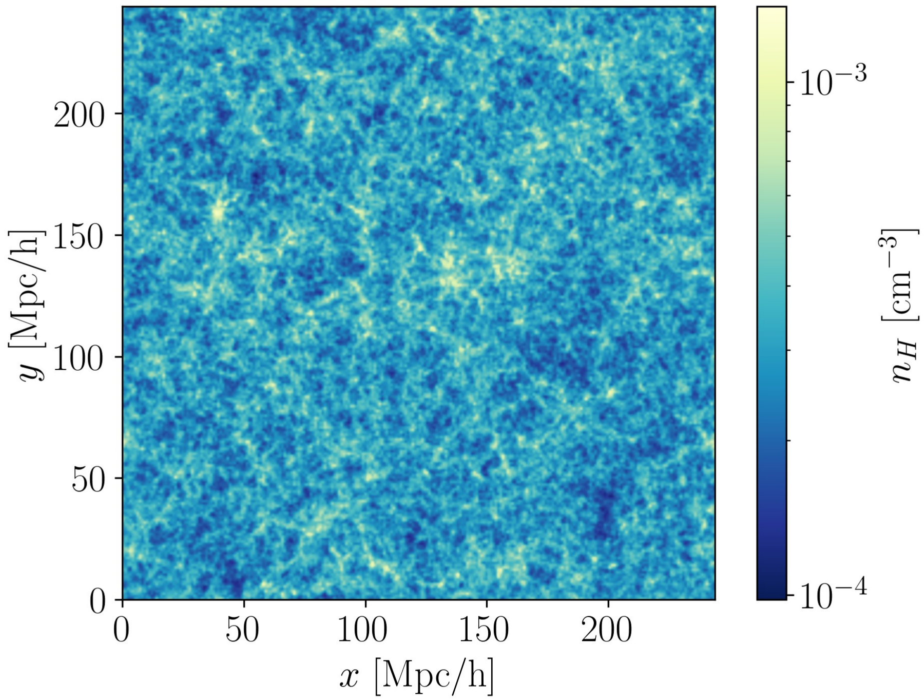




Platform: Piz Daint (1x Intel Haswell + 1x NVIDIA Tesla P100)

Illustrative Application

Fixed Density Field from (244 Mpc/h)^3 hydro simulation at z = 9



Find 10^5 most massive halos

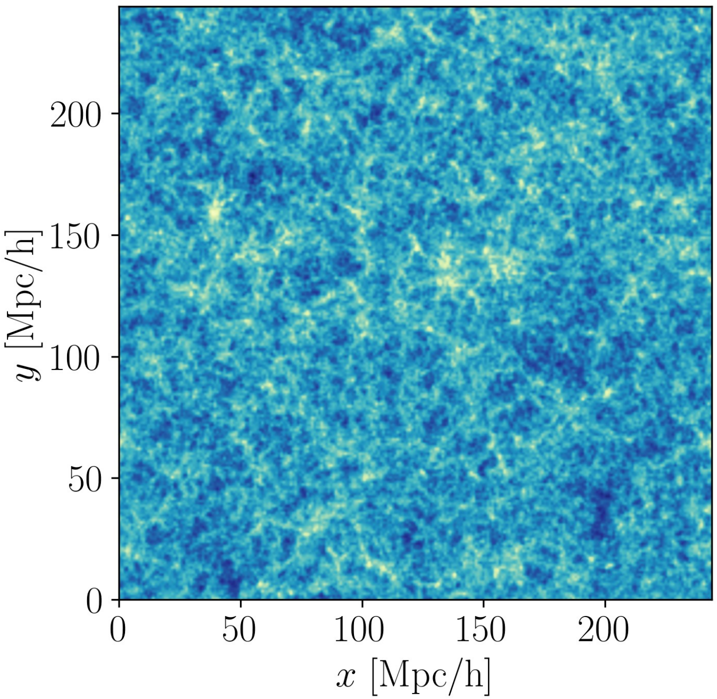
- Source model:

$$\dot{N}_\gamma = f_\gamma \frac{M\Omega_b}{\Omega_0 m_p t_s}$$

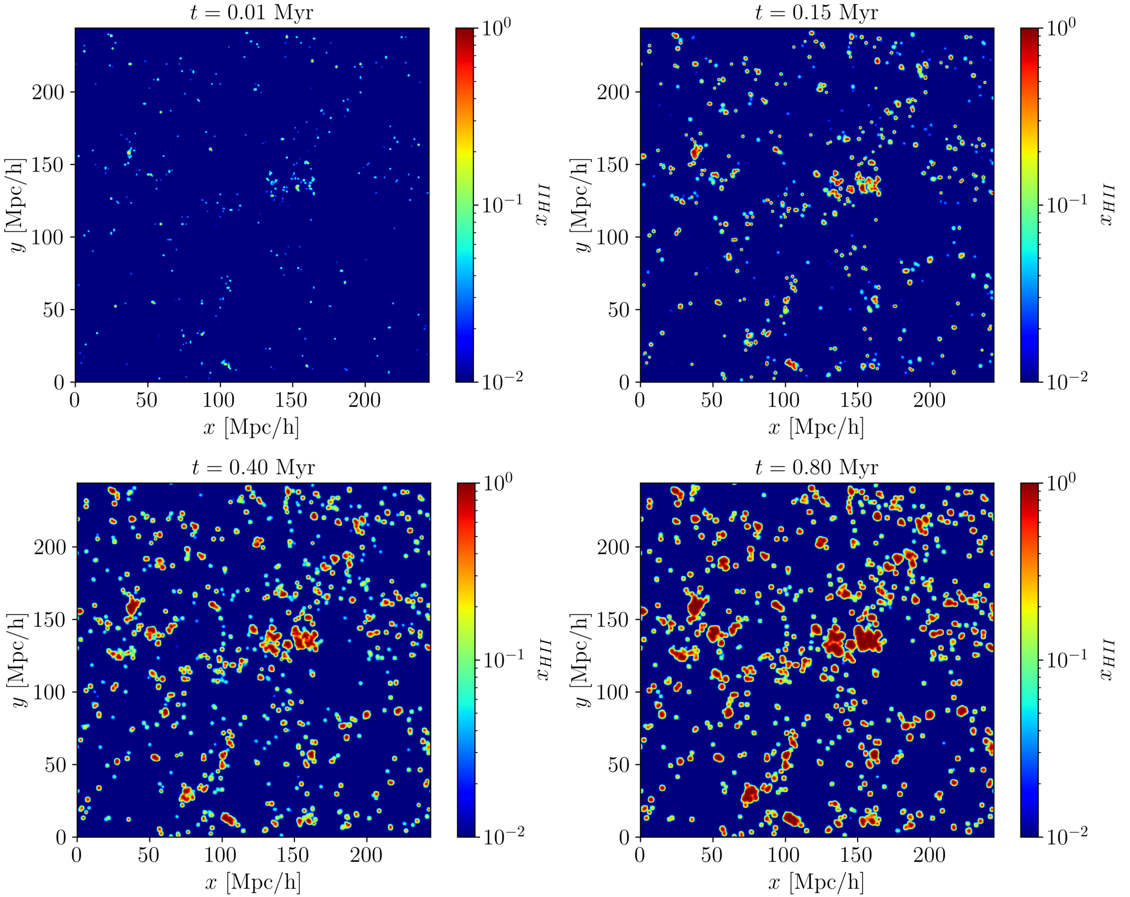
- $\boldsymbol{N_{grid}}$ **= 250,** $\boldsymbol{r_{RT}}$ **= 50**
- Use a black-body spectrum with $T = 10^5$ K
- Evolve for 0.8 Myr with Δt = 0.1 Myr

Illustrative Application

Fixed Density Field from (244 Mpc/h)³ hydro simulation at z = 9



*Computation Time: 25h on a **single node** on Piz Daint*



Implementation & Future Development

- ***OCTA* is implemented as a C++/CUDA library with python bindings**
- Developed for C2Ray but can easily be adapted for other codes and simulations (e.g. SWIFT, FIRE2)
- Future improvement ideas:
  - Treat multiple sources at once on one GPU
  - Ability to run on GPU clusters (multiple nodes)
  - Adaptive raytracing size (photon statistics)
  - Planing to apply for participation at Eurohack23 at CSCS to develop these ideas
- **Current goal:** do an actual cosmological EoR simulation and compare with existing result from C^2Ray (part of a SKACH Long HPC allocation at CSCS)

Thank you for your attention!