HIGH POWER LASER
SCIENCE AND ENGINEERING

RESEARCH ARTICLE

A rasterization-based ray-tracing method for laser–plasma interactions

Tao Tao [1], Zhujun Li [2], Kejian Qian[2], Xian Jiang[2], Guannan Zheng[1], Rui Yan[2,3], Haoran Liu[2], Qing Jia[1], Jun Li[1], Hang Ding[2], and Jian Zheng[1,3]

[1]*Department of Plasma Physics and Fusion Engineering, University of Science and Technology of China, Hefei, China*

[2]*Department of Modern Mechanics, University of Science and Technology of China, Hefei, China*

[3]*Collaborative Innovation Center of IFSA, Shanghai Jiao Tong University, Shanghai, China*

Abstract

This paper introduces a novel ray-tracing methodology for various gradient-index materials, particularly plasmas. The proposed approach utilizes adaptive-step Runge–Kutta integration to compute ray trajectories while incorporating an innovative rasterization step for ray energy deposition. By removing the requirement for rays to terminate at cell interfaces – a limitation inherent in earlier cell-confined approaches – the numerical formulation of ray motion becomes independent of specific domain geometries. This facilitates a unified and concise tracing method compatible with all commonly used curvilinear coordinate systems in laser–plasma simulations, which were previously unsupported or prohibitively complex under cell-confined frameworks. Numerical experiments demonstrate the algorithm's stability and versatility in capturing diverse ray physics across reduced-dimensional planar, cylindrical and spherical coordinate systems. We anticipate that the rasterization-based approach will pave the way for the development of a generalized ray-tracing toolkit applicable to a broad range of fluid simulations and synthetic optical diagnostics.

Keywords: geometric optics; inertial confinement fusion; laser-produced plasmas; ray tracing

1. Introduction

The interaction between high-power lasers and plasmas plays a crucial role in inertial confinement fusion (ICF), laser wakefield acceleration and advanced diagnostic techniques. Tracing rays in laser-generated plasmas[1] is the primary focus of this paper. Plasma as a medium is characterized by its spatially varying refractive index, which belongs to a family termed gradient-index materials. Geometric optics are often used for gradient-index ray tracing. Although the theoretical foundations of geometric optics were established nearly two centuries ago through Hamiltonian optics[2,3], good-quality applications have only become feasible with the advent of modern computational power and algorithmic advancements, which enable us to synthesize images of black holes[4], study the architecture of biological eyes[5],

design advanced antennas[6] and lenses[7,8] and optimize laser energy deposition in fusion plasmas[9–11]. Beyond the inherently dynamic and inhomogeneous nature of flow fields, plasmas exhibit unique ray–medium interactions. Light induces electron oscillations and heats the plasma through collisions[12]; nonuniformities in the beam energy distribution give rise to various instabilities[13–15]; and frequency differences between beams can even facilitate energy transfer from one beam to another[16,17]. Ray-tracing algorithms designed to capture these phenomena must balance physical fidelity with computational efficiency.

When it comes to numerical implementation, if the medium's refractive index exhibits certain symmetry and trajectories are the sole quantities of interest, as in most lens designs, tracing packages employing Runge–Kutta (RK)[18] or advanced symplectic-RK methods[19] have demonstrated remarkable efficiency. In such cases, rays do not reciprocally alter medium properties. For plasmas, however, the spatial distribution of electron density determines the trajectory of incident laser rays. In turn, the energy loss of the laser becomes the critical physical quantity that corresponds
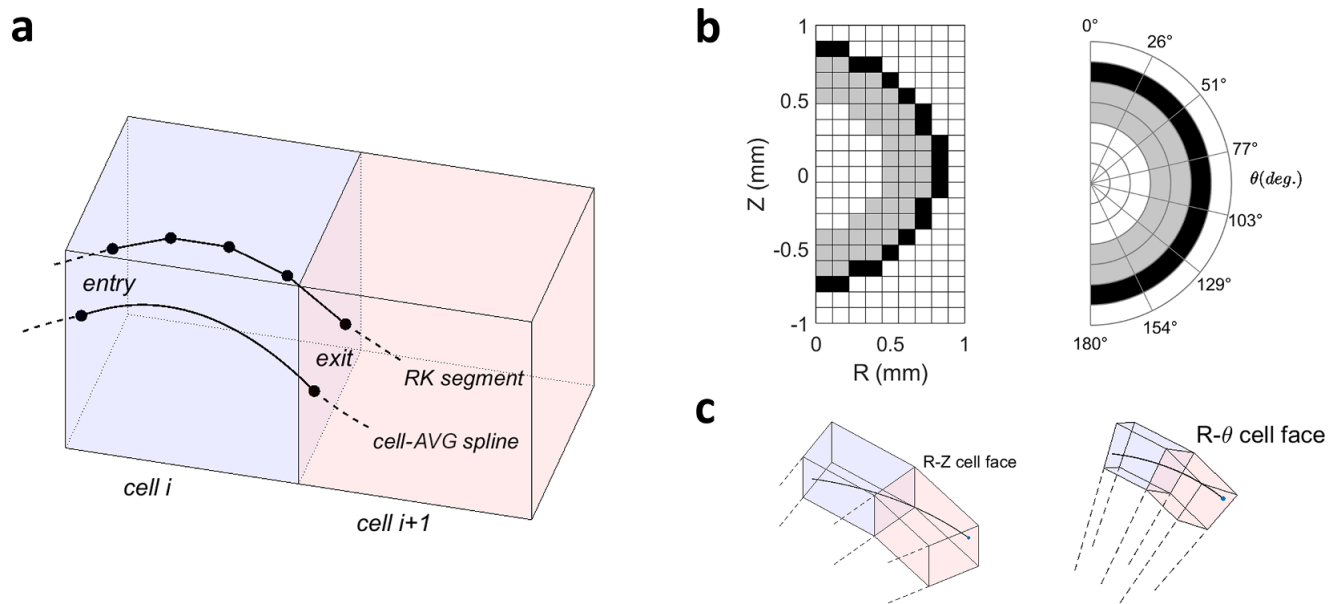
1

Figure 1. (a) Ray trajectories through cells: one follows the cell-AVG curved path, and the other represents the cell-confined RK straight-line path. (b) A spherical target represented in 2D $R-Z$ cylindrical coordinates and $R-\theta$ spherical coordinates. (c) Truncated-wedge cells derived from 2D cylindrical or spherical grids, lifted into a temporary 3D space for ray tracing.

to the thermal gain of the fluid. The challenge here lies in the fact that energy deposition and ray trajectories are defined on two distinct grid systems: deposition is a field quantity that follows the structured fluid cells, while ray trajectories are defined by a sequence of discrete nodes, with varying positions and numbers. Consequently, a tracing algorithm must determine which fluid cells should receive the energy loss associated with each small segment of the ray. In practice, selecting an appropriate tracing logic to ensure compatibility between the fluid grid and ray trajectory coordinate systems is the starting point for plasma ray tracing. In previous treatments, the most commonly used is the 'cell-confined' logic: a ray begins at the entry point on the cell face, propagates within the cell along a predefined path and exits at another point on the cell face. The term 'confined' emphasizes that ray nodes are precisely the intersection points of the ray with the cell faces. Rays are traced on a cell-by-cell basis and, at each step, the energy loss of the ray is deposited into the cell it currently traverses.

Many commonly used radiation-hydrodynamics (RHD) codes adopt cell-confined logic in their laser modules. MULTI-2D[20] assumes straight-line ray paths within each cell, ignoring refraction; FLASH[21,22] and RHDLPP[23] incorporate curved trajectories to account for refraction; while proprietary codes such as xRAGE[24] and DRACO[25] use the Mazinisin tracing module[26], which supports curved trajectories and energy transfer between rays. FLASH, RHDLPP and Mazinisin tracing can all be traced back to the 'cell-AVG' algorithm proposed by Kaiser[27]. Building on cell-confined methods, cell-AVG further assumes a linear variation of electron density within each cell.

In coordinate systems with globally parallel cell faces (e.g., Cartesian grids), cell-AVG ray trajectories can be analytically expressed as quadratic functions. As a result, the ray entry and exit points on cell faces (as shown in Figure 1(a)) can be efficiently determined with minimal computational cost.

Despite its success and widespread adoption, cell-confined tracing faces notable challenges in certain applications, for example, tracing rays in cylindrical and spherical coordinate systems. When plasmas exhibit symmetry, curved coordinate systems are often employed to reduce dimensionality, as illustrated in Figure 1(b) with a two-dimensional (2D) cylindrical or spherical coordinate system for an ICF layered target. In these cases, fluid cell faces are curved, and the size and orientation of cells are not globally uniform. Ray trajectories must therefore be constructed in the local reference frame of the individual fluid cells, and the entry and exit points of rays cannot be expressed analytically. Instead, iterative methods are required to approximate these intersection points (see Ref. [22], Section 18.4.9), which reduces computational performance and introduces significant parallelization overhead. Similarly, in 'reduced-dimensional' simulations, where lasers interact with one-dimensional (1D) or 2D fluid domains at oblique angles, the fluid cells must be temporarily extruded into truncated-wedge cells (TW-cells) and stacked up to approximate the three-dimensional (3D) structure, as shown in Figures 1(b) and 1(c), where ray calculations are performed at the entry and exit points on the surfaces of each wedge. The formation of TW-cells requires different tracing codes for every laser–fluid geometry combination, making implementation highly complex

and significantly increasing the difficulty of incorporating user-defined ray physics.

It is worth noting that, in addition to the cell-AVG method, the FLASH code provides an alternative RK-based tracing approach called 'CIRK' (see Section 18.4.4.3 of Ref. [22]). This method allows ray trajectories to be recorded in a 3D Cartesian coordinate system $[x, y, z]$, while fluid fields are recorded in a 2D cylindrical coordinate system $[r, z]$, with their interactions bridged through Jacobian transformations. This avoids the complexity of describing ray trajectory splines in curvilinear coordinates. However, its adoption is far less widespread than cell-AVG in plasma simulations due to inherent limitations. Firstly, CIRK relies on high-order interpolation of fluid fields to propagate rays, but such interpolated fields are susceptible to oscillations, potentially resulting in incorrect ray refraction. Secondly, CIRK still requires ray nodes to align with cell faces, achieved through iterative adjustments of the RK step size. Consequently, this method does not fundamentally resolve the coordination challenges between ray trajectories and fluid grids.

The limitations of these cell-confined methods have constrained the implementation of curved coordinate systems in mainstream RHD programs. For instance, MULTI-2D does not include refraction modeling in spherical coordinates, and FLASH lacks support for spherical ray tracing entirely. Considering the prevalent use of curved coordinate systems and the substantial computational burden associated with ray tracing in laser–plasma simulations, there remains a pressing need for further innovation in ray-tracing methodologies.

This paper proposes a novel ray-tracing approach based on rasterization, aimed at achieving high-performance plasma ray tracing in curved coordinate systems and reduced-dimensional simulations. The key feature of this method is the decoupling of ray trajectory calculation from energy deposition. In the trajectory-solving phase, ray propagation is performed using adaptive step-size RK–Fehlberg (RKF) integration[28], where the step size is determined physically by the gradient of the background medium rather than being constrained to terminate at cell boundaries. This eliminates the need for iterative searches for cell entry and exit points or the construction of TW-cells. In the deposition-solving phase, rasterization techniques are employed to segment the ray's energy loss based on its intersection with the fluid grid and redistribute it to the corresponding fluid cells, and the energy deposition field is accumulated accordingly.

The key innovation of the rasterization-based method lies in the development of a unified and significantly simplified tracing framework, capable of handling various curved coordinate systems and reduced-dimensional simulations – an achievement unattainable with previous cell-confined approaches. Furthermore, rasterization-based tracing provides a practical advantage: ray trajectories are recorded globally in a Cartesian coordinate system rather than being constrained by the curved coordinates of the fluid. This greatly simplifies the incorporation of additional ray physics, making the process intuitive and user-friendly. Although designed primarily for plasma media, this rasterization-based approach is also applicable to a broader range of gradient-index materials.

The remainder of this paper is organized as follows. Section 2 outlines the physical foundations of ray motion and energy deposition. Section 3 details the complete workflow for constructing a rasterized ray-tracing program. Section 4 presents test cases demonstrating the program's capabilities in planar, spherical and cylindrical coordinates, including comparisons of ray trajectories, energy deposition and frequency shift physics with analytical results. Finally, Section 5 concludes the paper and discusses future perspectives.

2. Physics of ray motion and deposition in plasma

2.1. Ray motion equation

The ray motion equation can be derived from either the Hamiltonian geometric optics framework[19] or directly from wave equations. In this paper, we adopt the Helmholtz equation as the starting point. For a monochromatic wave $E(\vec{r}, t) = E_a(\vec{r}) e^{-i\omega t}$, the electric field amplitude E_a satisfies the scalar equation:

$$\nabla^2 E_a(\vec{r}) + k_0^2 \varepsilon(\omega, \vec{r}) E_a(\vec{r}) = 0, \qquad (1)$$

where c is the speed of light in vacuum, $k_0 = \omega/c$ is the vacuum wave number and $\varepsilon(\omega, \vec{r}) = \varepsilon'(\omega, \vec{r}) + i\varepsilon''(\omega, \vec{r})$ represents the medium's complex dielectric function. We assume the electric field can be expressed as a combination of a slowly varying envelope and a rapidly oscillating phase $E_a(\vec{r}) = A(\vec{r}) e^{ik_0 S(\vec{r})}$, where $S(\vec{r})$ is referred to as the 'eikonal'. Substituting this trial solution into Equation (1), the imaginary and real parts respectively yield the following:

$$2(\nabla A) \cdot (\nabla S) + A\nabla^2 S + k_0 \varepsilon'' A = 0, \qquad (2)$$

$$(\nabla S)^2 = \varepsilon'^2 = N^2, \qquad (3)$$

where N is defined the medium refractive index. Equation (2) is the transport equation, which can be rewritten as $\nabla \cdot (A^2 c\nabla S) = -ck_0 \varepsilon'' A^2$. If we draw an analogy between the scalar field S and a velocity potential, its gradient ∇S represents an irrotational steady flow field with a group velocity of $c\nabla S$. The flowing particles possess an energy density A^2, with an absorption rate per unit time given by $-ck_0 \varepsilon'' A^2$. Equation (3), known as the eikonal equation, belongs to the Hamilton–Jacobi family. The eikonal equation allows us to transition to a 'ray particle' perspective: let $\vec{r}(s)$ denote the position vector as a function of the arc length s, where $ds^2 = d\vec{r} \cdot d\vec{r}$. In this perspective, ray motion is

everywhere perpendicular to the surfaces of constant phase, with the motion vector given by $d\vec{r}/ds = \nabla S/N$.

Since solving directly for the eikonal S is often challenging, we take an additional derivative with respect to the arc length:

$$\frac{d}{ds}\left(N\frac{d\vec{r}}{ds}\right) = \frac{d}{ds}(\nabla S) = \left(\frac{\nabla S}{N} \cdot \nabla\right)\nabla S,$$

where the differential along the arc is transformed into a directional derivative along the velocity unit vector. Taking the gradient of the eikonal equation $\nabla N^2 = \nabla(\nabla S)^2 = \nabla(\nabla S \cdot \nabla S) = 2(\nabla S \cdot \nabla)\nabla S$, and substituting it into the right-hand side of the equation above, we obtain a ray motion equation that does not explicitly rely on the eikonal equation and is well-suited for numerical discretization:

$$\frac{d}{ds}\left(N\frac{d\vec{r}}{ds}\right) = \nabla N. \tag{4}$$

2.2. Plasma ray trajectory and energy deposition

The plasma dielectric function is expressed as

$$\varepsilon = 1 - \frac{n_e}{n_c}\frac{\omega}{\omega + i\nu_{ei}}, \tag{5}$$

where n_e is the electron density, critical density $n_c = \omega^2 m_e/(4\pi e^2)$, m_e is the electron mass, e is the electron charge and ν_{ei} is the electron collision frequency (refer to Ref. [1], equation 10.133):

$$\nu_{ei} = \frac{4}{3}\left(\frac{2\pi}{m_e}\right)^{1/2}\frac{n_e Z e^4 \ln\Lambda}{(k_B T_e)^{3/2}}, \tag{6}$$

$$\ln\Lambda = \ln\left[\frac{3}{2Ze^3}\left(\frac{k_B^3 T_e^3}{\pi n_e}\right)^{1/2}\right], \tag{7}$$

where Z is the effective charge, $k_B T_e$ represents the electron energy and $\ln\Lambda$ is the Coulomb logarithm. When the electron collision frequency is much smaller than the wave frequency (a condition satisfied in most plasma corona absorption; for complete field reconstruction and proper absorption treatment, see Ref. [29]), the dielectric function simplifies to $\varepsilon \approx 1 - n_e/n_c(1 - i\nu_{ei}/\omega)$.

To calculate inverse-bremsstrahlung absorption, we define a virtual time $dt = ds/|\vec{v}|$ where $\vec{v} = c\nabla S = cN\hat{v}$ is the group velocity of the electromagnetic wave in plasma. Substituting the imaginary part of the dielectric function into the transport equation, and integrating over the trajectory parameter t (rather than s, since absorption depends on the time the wave packet resides in the absorbing medium), we obtain the evolution of the ray energy $P \propto A^2$:

$$P(t_1) = P(t_0) + \int_{t_0}^{t_1} -\frac{n_e(t)}{n_c}\nu_{ei}(t)P(t)dt. \tag{8}$$

Using the virtual time t in place of ds in Equation (4), and incorporating the plasma refractive index, Equation (4) can be reformulated as two first-order ordinary differential equations:

$$\frac{d\vec{r}}{dt} = \vec{v}, \tag{9}$$

$$\frac{d\vec{v}}{dt} = -\frac{c^2 \nabla n_e(\vec{r})}{2n_c}. \tag{10}$$

Given the initial ray position and speed, Equations (9) and (10) delineate a trajectory. A large ensemble of rays collectively models the refraction and reflection of the entire laser beam. In numerical solutions, ray trajectories are represented as a sequence of discrete spatial nodes connected by small line segments, whose shapes are determined by locally interpolated plasma fields and their gradients.

2.3. Rasterization ray-trace concept

Ray trajectory nodes are inherently unstructured in space; forcing them to conform to the structured fluid cells is the primary source of complexity in cell-confined algorithms. This raises an important question: can the constraints imposed by fluid cells on rays be removed?

A similar problem arises in computer graphics, where 3D unstructured models are projected onto structured 2D observation grids. For example, capturing an image of a 3D luminous object with a virtual camera involves determining the model's vertices, surface emissivity and ray transport trajectories through the medium, followed by a rasterization step (see Ref. [30], Chapter 7, and Ref. [31], Section 15.6) to accumulate signals received by camera pixels. This concept can be directly applied to ray tracing in plasma: the refractive medium and the camera correspond to the fluid domain, while pixel counts are analogous to cell-accumulated quantities such as deposited energy, light intensity, polarization and frequency shifts.

The core idea of rasterization is that ray stepping is unconstrained, and deposition is computed by accumulating contributions within the fluid grid. This approach introduces several key differences from traditional methods: the decoupling of ray trajectories from the fluid geometry, with rays always described in global Cartesian coordinates; the interpolation of a continuous fluid background field to maintain domain-wide continuity, enabling unconstrained RK advancement; the flexibility to freely position ray nodes in space, connected by straight-line segments; and the need to determine the intersection relationships between rays and grid cells during deposition.

3. Full implementation of rasterized ray tracing

This section outlines the logical sequence for implementing the rasterized ray-tracing algorithm. The tracing code consists of four modules: the lens system, field interpolation, adaptive ray stepping and rasterized deposition. These modules work together to execute the tracing algorithm in curvilinear coordinates.

3.1. Lens system

The lens system serves two primary functions: initializing ray directions and power, and placing rays at their intersection with the domain boundary while properly initializing their velocities. Two circular planes represent the focusing lens and focal spot, divided into subregions, each carrying a fraction of the spot energy:

$$P_i = \frac{S_i I_i}{\Sigma S_i I_i} P_b, \tag{11}$$

where P_i is ray power, P_b is beam power, S_i is subregion area and I_i is intensity at the grid center. Subregion division methods include rectangular grids, polar coordinate grids and finite element grids. Figure 2(a) shows a rectangular division template for a super-Gaussian spot, with the red circle indicating the lens boundary. The lens and focal plane circles share the same division template, scaled according to their radii.

Rays spawn at subregion centers, initially distributed in the $Z = 0$ plane with a normal direction along $Z+$. Rotation and translation are applied to position the rays on the user-defined 3D lens and focal planes. For a given polar angle θ_B and azimuthal angle ϕ_B of the line connecting the lens and focal plane centers, a corresponding 3D rotation is constructed:
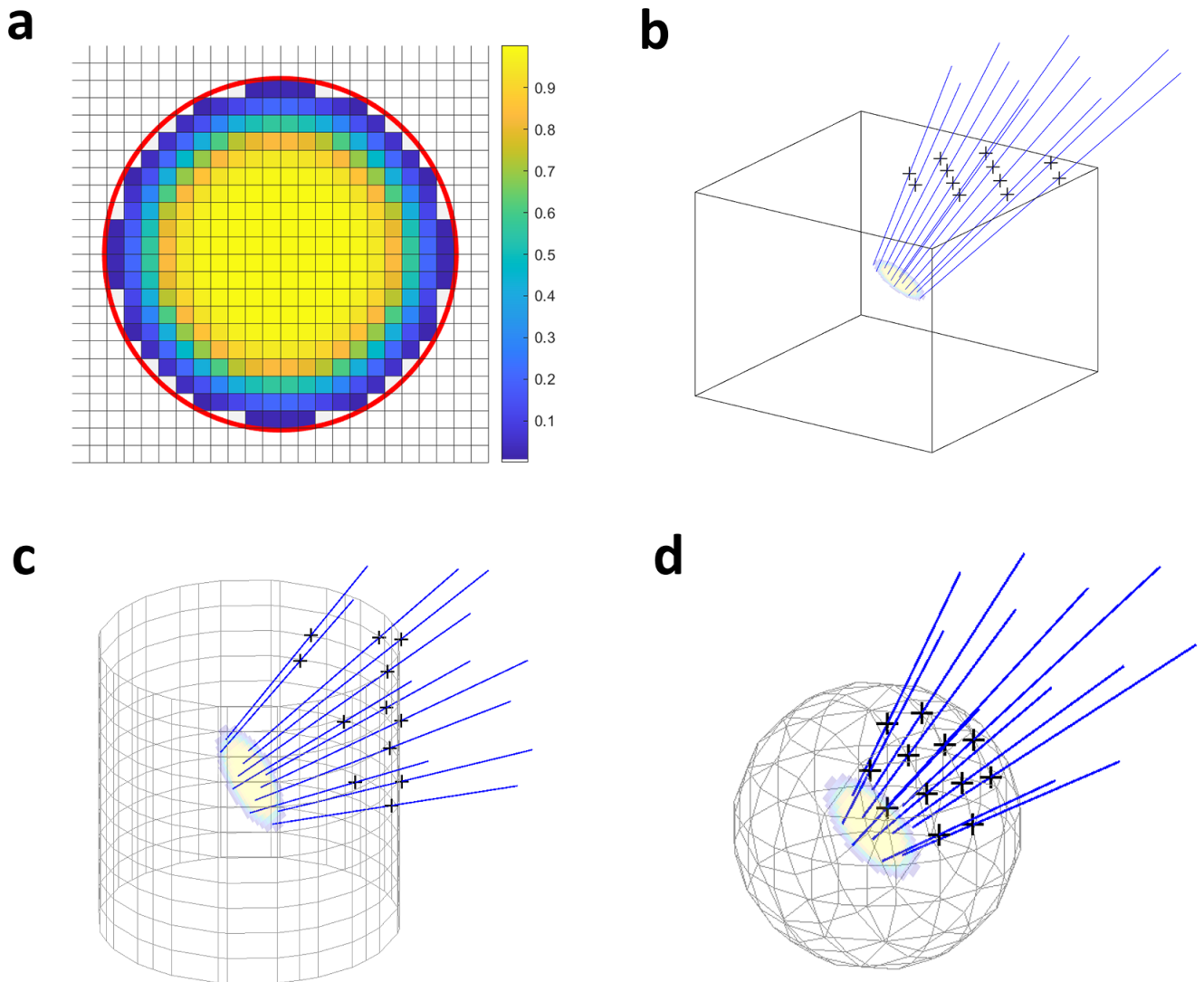

Figure 2. (a) Power partitioning of the laser focal spot. Intersection points (marked with '+') of rays with (b) rectangular, (c) cylindrical and (d) spherical computational domains between the lens and focal planes.
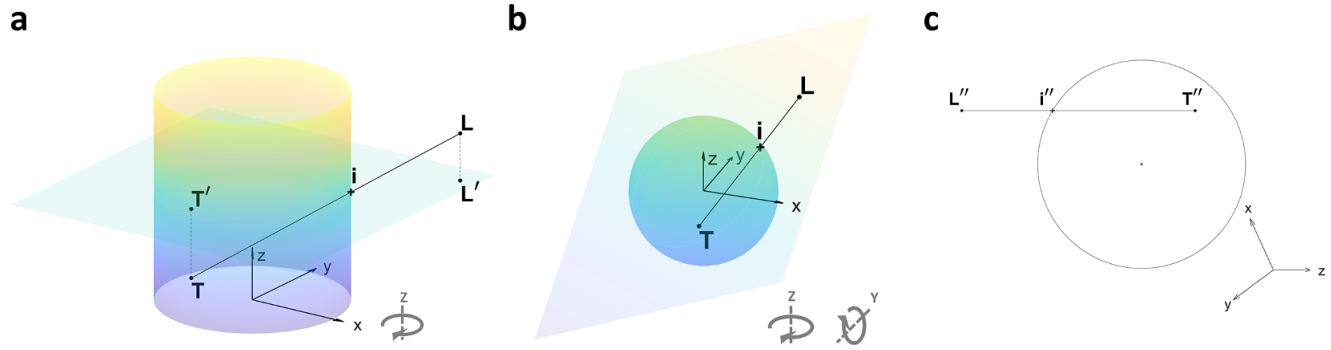
Figure 3. (a) Intersection of the connecting line with a cylindrical surface, where $\vec{T}$ and $\vec{L}$ are projected onto the solution plane followed by a Z-rotation. (b) Intersection of the connecting line with a spherical surface, where $\vec{T}$ and $\vec{L}$ are mapped to the solution plane through a Z-rotation followed by a Y-rotation. (c) In solution plane, the line connecting T'', L'' and i'' is parallel to the X-axis, enabling straightforward calculation of the intersection point i'' coordinates.

$$\mathcal{R}_B = \begin{bmatrix} \cos(\pi+\theta_B) & -\sin(\pi+\theta_B) & 0 \\ \sin(\pi+\theta_B) & \cos(\pi+\theta_B) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$\times \begin{bmatrix} \cos(\pi/2+\phi_B) & 0 & \sin(\pi/2+\phi_B) \\ 0 & 1 & 0 \\ -\sin(\pi/2+\phi_B) & 0 & \cos(\pi/2+\phi_B) \end{bmatrix}.$$

In $\mathcal{R}_B$, the second matrix performs a rotation around the Y-axis to adjust the inclination, followed by the first matrix rotating around the Z-axis to adjust the azimuth. The rotated points are then translated to position the rays on the lens and focal planes in 3D space.

Ray intersections with the computational domain mark the starting points for ray tracing, as illustrated in Figures 2(b)–2(d) with '+' signs. In Cartesian coordinates, determining the starting points $\vec{i}$ is straightforward, as discussed in the cell-AVG introduction. However, this process becomes more complex in curvilinear coordinates. Given two points $\vec{T}$ (inside) and $\vec{L}$ (outside) relative to a surface of radius r, the aiming vector from the lens to the focal plane is $\vec{a} = \vec{T} - \vec{L}$. Here, vector components are denoted by subscripts (e.g., $\vec{a} = (a_x, a_y, a_z)$), uppercase letters X, Y, Z represent coordinate axes and $\|\vec{a}\|$ denotes the vector magnitude. The intersection-finding procedure preserves the curved geometry of radial cell faces, distinguishing it from methods used for co-planar TW-cell faces.

(I) For the cylindrical domain shown in Figure 3(a), construct the projection and rotation matrices:

$$\mathcal{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}; \quad \mathcal{R} = \begin{bmatrix} a_x & -a_y \\ a_y & a_x \end{bmatrix}/\sqrt{a_x^2+a_y^2}, \quad (12)$$

where $\mathcal{P}$ represents a projection onto the X–Y plane, yielding $\vec{T}' = \mathcal{P}\vec{T}$, $\vec{L}' = \mathcal{P}\vec{L}$. Here, $\mathcal{R}$ is a rotation in the X–Y plane aligning the unit vector $(1,0)$ to the direction (a_x, a_y), while $\mathcal{R}^{-1}$ rotates any (a_x, a_y) back to $(1,0)$. This produces $\vec{T}'' = \mathcal{R}^{-1}\vec{T}'$ and $\vec{L}'' = \mathcal{R}^{-1}\vec{L}'$. The points T'', L'', i'' are thus mapped onto the 'solution plane' shown in Figure 3(c), where the line connecting these points is guaranteed to be

parallel to the local X''-axis, allowing the intersection point coordinates i'' to be directly determined:

$$\vec{i}'' = \begin{bmatrix} \mathrm{sgn}(L_x'')\sqrt{r^2-L_y''^2} \\ L_y'' \end{bmatrix}. \quad (13)$$

Rotating i'' back and scaling along the aiming vector $\vec{a}$ gives the intersection point in the fluid reference frame:

$$\vec{i} = \vec{L} + \frac{\left\| \mathcal{R}\vec{i}'' - \vec{L}' \right\|}{\left\| \vec{L}' - \vec{T}' \right\|}\vec{a}. \quad (14)$$

For cylindrical domains, additional considerations are necessary. Firstly, rays may intersect the flat faces of the cylinder. The scaling factors to the top and bottom faces are $|(L_z - Z_{\max})/(T_z - L_z)|$ and $|(T_z - Z_{\min})/(T_z - L_z)|$, respectively. The minimum of these three factors, including the one for the curved surface, should be chosen to scale $\vec{a}$, as the true intersection is the closest to the lens. Secondly, when laser incidence aligns with the cylindrical axis ($\sqrt{a_x^2+a_y^2}=0$), the rotation matrix $\mathcal{R}$ must be set to the identity matrix $\mathcal{I}$ to avoid singularity.

(II) For the spherical domain shown in Figure 3(b), two rotation matrices are constructed:

$$\mathcal{R}_{\mathcal{Y}} = \begin{bmatrix} \sqrt{a_x^2+a_y^2}/\|\vec{a}\| & 0 & -a_z/\|\vec{a}\| \\ 0 & 1 & 0 \\ a_z/\|\vec{a}\| & 0 & 0 \end{bmatrix};$$

$$\mathcal{R}_{\mathcal{Z}} = \begin{bmatrix} a_x/\sqrt{a_x^2+a_y^2} & -a_y/\sqrt{a_x^2+a_y^2} & 0 \\ a_y/\sqrt{a_x^2+a_y^2} & a_x/\sqrt{a_x^2+a_y^2} & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

where $\mathcal{R}_{\mathcal{Y}}$ represents a rotation around the Y-axis by the polar angle of the connecting line, and $\mathcal{R}_{\mathcal{Z}}$ represents a rotation around the Z-axis by its azimuth angle. Applying these rotations to the points $\vec{L}, \vec{T}$ gives $\vec{T}'' = (\mathcal{R}_{\mathcal{Z}}\mathcal{R}_{\mathcal{Y}})^{-1}\vec{T}$,
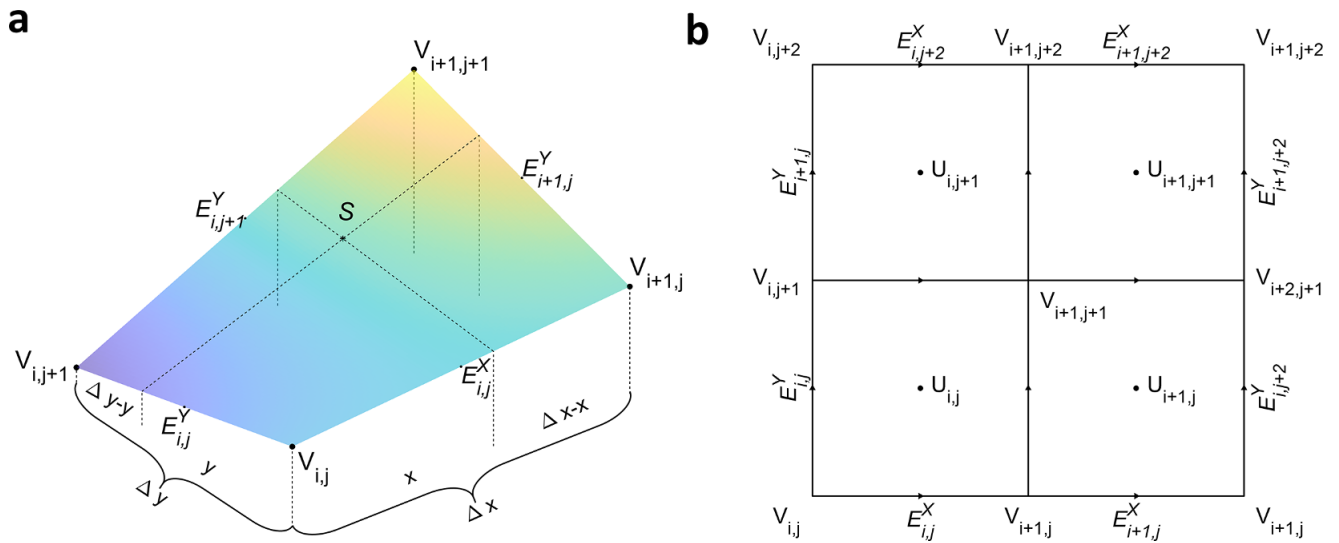
a

b

Figure 4. (a) Bilinear interpolation for interior point S's value and spatial derivatives. Region dimensions are $\Delta x, \Delta y$, with S coordinates x, y (origin at the lower-left corner). Here, V represents cell vertex values and E^X, E^Y are edge-centered gradients. (b) Vertex values V and edge gradients E^X, E^Y are derived from known cell-centered values U.

$\vec{L}'' = (\mathcal{R}_{\mathcal{Z}}\mathcal{R}_{\mathcal{Y}})^{-1}\vec{L}$. The intersection point coordinates are then determined on the solution plane as follows:

$$\vec{i}'' = \begin{bmatrix} \mathrm{sgn}\left(L''_x\right)\sqrt{r^2 - L''^2_x - L''^2_z} \\ L''_y \\ L''_z \end{bmatrix}. \tag{15}$$

Rotating i'' back to the fluid reference frame gives the spherical intersection:

$$\vec{i} = (\mathcal{R}_{\mathcal{Z}}\mathcal{R}_{\mathcal{Y}})\vec{i}''. \tag{16}$$

This establishes the ray starting points on the domain boundary. Alternatively, starting points can be determined by solving the quadratic equation of a line and a curved surface. However, matrix operations are more compact and efficient due to simplification and pre-evaluation. They also enable highly vectorized data flows, leveraging modern hardware for improved performance.

Finally, the ray velocity must be properly initialized, as rays may originate inside the plasma corona. The velocity magnitude is determined by the local refractive index N, where $v_0 = cN = c\sqrt{1 - n_\mathrm{e}/n_\mathrm{c}}$. The velocity direction is aligned with the aiming vector, given by $v_{0x} = cNa_x/\|\vec{a}\|$, $v_{0y} = cNa_y/\|\vec{a}\|$, $v_{0z} = cNa_z/\|\vec{a}\|$.

3.2. Field interpolation

Ray advancement requires the field value at each node in space. The general principle of interpolation is to ensure continuity and minimize oscillation. Testing has shown that linear (1D), bilinear (2D) and trilinear (3D) interpolation schemes are optimal for this purpose.

For clarity, a 2D bilinear interpolation case is illustrated in Figure 4(a). Cell edge lengths are $\Delta x, \Delta y$, with V as the base value at the cell vertex. The distances from the interior sampling point S to cell walls are x, y:

$$S = V_{i,j}\frac{\Delta x - x}{\Delta x}\frac{\Delta y - y}{\Delta y} + V_{i+1,j}\frac{x}{\Delta x}\frac{\Delta y - y}{\Delta y}$$
$$+ V_{i,j+1}\frac{\Delta x - x}{\Delta x}\frac{y}{\Delta y} + V_{i,j}\frac{x}{\Delta x}\frac{y}{\Delta y}.$$

Interpolation weights for each vertex are proportional to the areas of the quadrilaterals opposite them. Notably, despite the term 'linear', interpolated fine meshes are generally not planar, as shown in Figure 4(a). Since bilinear and trilinear interpolations ensure continuity across cells, rays crossing cell boundaries do not require total reflection checks.

Field gradients are essential for advancing the ray and are calculated as follows:

$$\frac{\partial S}{\partial x} = \frac{S(x+\varepsilon, y) - S(x-\varepsilon, y)}{2\varepsilon}$$
$$= E^X_{i,j}\frac{\Delta y - y}{\Delta y} + E^X_{i,j+1}\frac{y}{\Delta y},$$
$$E^X_{i+1,j} = \frac{V_{i+1,j} - V_{i,j}}{\Delta x},$$
$$E^X_{i,j} = \frac{V_{i+1,j+1} - V_{i,j+1}}{\Delta x},$$

where ε is a small step size, $E_{i,j}, E_{i,j+1}$ are gradient values at edge centers and superscripts denote direction. The derivation shows that the gradient at any point within the cell is a linear combination of edge-centered gradients. The expression for $\partial S/\partial y$ takes a similar form, with indices appropriately interchanged.
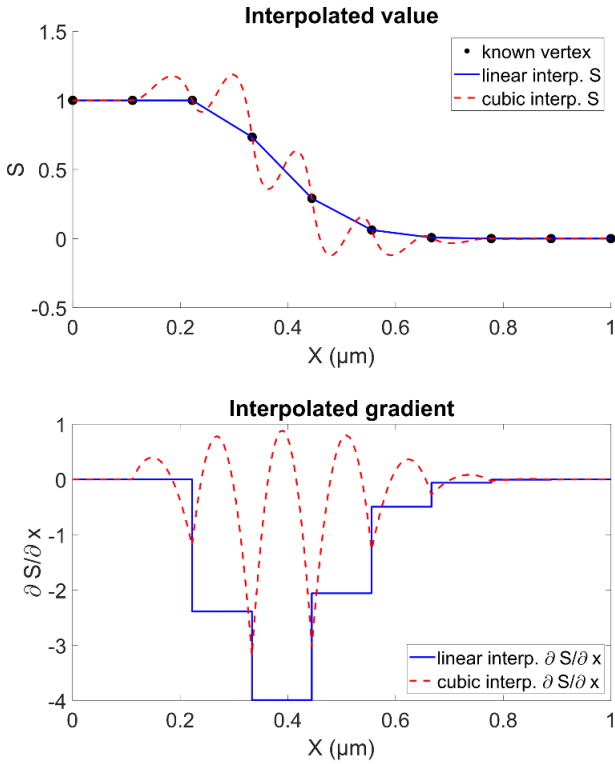
Figure 5. Comparison of linear interpolation and cubic interpolation methods for constructing internal field values S and gradient values $\partial S/\partial x$ of a rarefaction wave.

For specific ray-tracing tasks, physical quantities relevant to laser trajectories and energy deposition – such as density, temperature and effective charge state – are required. As shown in Figure 4(b), these fluid quantities are defined at cell centers, denoted as U. During domain initialization, cell-center interpolation is used to compute vertex values V (outermost extrapolated vertices take the nearest cell-center values). Edge-centered gradients E^X, E^Y are then calculated by dividing vertex value differences by edge lengths $\Delta x, \Delta y$.

Piecewise linear interpolation does not ensure continuity of first-order derivatives, leading to the question: why not use higher-order interpolation schemes? Higher-order methods, however, are unsuitable for ray tracing in plasma due to severe Runge phenomena. Figure 5 demonstrates this issue using a rarefaction wave field with known values (black asterisks). Piecewise cubic interpolation is compared to linear interpolation. The cubic interpolation assumes an intra-cell field form $S(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$, where coefficients $a_0 - a_3$ are determined by four conditions: field values $S(0), S(1)$ and gradient values $\partial S(0)/\partial x, \partial S(1)/\partial x$ at cell corners.

Although cubic interpolation ensures continuity of first-order spatial derivatives, it introduces substantial oscillations, which worsen with cell refinement. Such oscillations can even produce negative values for inherently positive physical quantities, as shown in Figure 5. This leads to

incorrect ray refraction locations, making higher-order methods unreliable for this application. In contrast, bilinear/trilinear interpolation offers superior robustness. When combined with adaptive-step RK methods, the lack of first-order derivative continuity does not significantly affect ray-tracing accuracy.

3.3. Adaptive ray stepping

The ray trajectories are solved using the RKF method with adaptive step size. Like all RK methods, the RKF method approximates ray curves through a straight-line segments sequence ordered according to the time parameter t. The advantage of the RKF method lies in its dynamic adjustment of the step size Δt, allowing larger steps in regions with gradual field variations and smaller steps in regions with sharp refraction. In plasmas, where the refractive index distribution is highly nonuniform, adaptive step sizing can often reduce computational costs by one to two orders of magnitude. The system of ordinary differential equations for ray trajectories is as follows:

$$\dot{U} = \frac{d}{dt}\begin{bmatrix} \vec{r} \\ \vec{v} \\ P \\ \dots \end{bmatrix} = \begin{bmatrix} \vec{v} \\ \vec{g} \\ -v_{ei}P \\ \dots \end{bmatrix} = \begin{bmatrix} \vec{v} \\ -c^2 \nabla n_e(\vec{r})/2n_c \\ -v_{ei}(\vec{r})P \\ \dots \end{bmatrix}, \quad (17)$$

where $\vec{U}$ represents the solution vector, ray acceleration is $\vec{g} = \vec{f}/m$ with $m = 1$, P denotes ray power and v_{ei} is the absorption coefficient, calculated via field interpolation. The ellipsis indicates other physical quantities varying along the ray path. The RK4(5) coefficients from Fehlberg's original work[28] are used to estimate the solution gradient at the initial condition $\vec{U}^n$:

$$\vec{k}_1 = \dot{\vec{U}}_n(\vec{r}_0),$$

$$\vec{k}_2 = \dot{\vec{U}}_n(\vec{r}_0) + (2/9)\vec{k}_1 \Delta t,$$

$$\vec{k}_3 = \dot{\vec{U}}_n(\vec{r}_0) + \left((1/12)\vec{k}_1 + (1/4)\vec{k}_2\right)\Delta t,$$

$$\vec{k}_4 = \dot{\vec{U}}_n(\vec{r}_0) + \left((69/128)\vec{k}_1 + (-243/128)\vec{k}_2 \right.$$
$$\left. + (135/64)\vec{k}_3\right)\Delta t,$$

$$\vec{k}_5 = \dot{\vec{U}}_n(\vec{r}_0) + \left((-17/12)\vec{k}_1 + (27/4)\vec{k}_2 \right.$$
$$\left. + (-27/5)\vec{k}_3 + (16/15)\vec{k}_4\right)\Delta t,$$

$$\vec{k}_6 = \dot{\vec{U}}_n(\vec{r}_0) + \left((65/432)\vec{k}_1 + (-5/16)\vec{k}_2 \right.$$
$$\left. + (13/16)\vec{k}_3 + (4/27)\vec{k}_4 + (5/144)\vec{k}_5\right)\Delta t.$$

The RKF method estimates the single-step error by comparing the solution difference between fourth- and fifth-order

approximations:

$$\text{Err} = \Big((1/150)\vec{k}_1 + (-3/100)\vec{k}_3 + (16/75)\vec{k}_4$$
$$+ (1/20)\vec{k}_5 + (-6/25)\vec{k}_6 \Big) \Delta t.$$

If the error is below the tolerance Tol, this step is accepted, producing $\vec{U}_{n+1}$ for the next time step:

$$\vec{U}_{n+1} = \vec{U}_n + \Big((47/150)\vec{k}_1 + (12/25)\vec{k}_3$$
$$+ (32/225)\vec{k}_4 + (1/30)\vec{k}_5 + (6/25)\vec{k}_6 \Big) \Delta t.$$

The time step is dynamically adjusted as follows:

$$\Delta t_{\text{new}} = C \Delta t (\text{Err}/\text{Tol})^{1/5}.$$

The default safety factor is $C = 0.9$. The initial step size can be conservatively estimated as $\Delta t_{\text{init.}} = 0.5N\Delta X/c$ (ΔX is the cell edge length). For accepted steps, the step size increases but is capped at $\Delta t_{\text{max}} = \Delta X/c$ to ensure the ray does not cross more than two cells per coordinate direction in a single RK step. If the tolerance is not met, the calculation restarts from $\vec{U}_n$ with progressively smaller trial step sizes.

While not the fastest in convergence, Fehlberg's original coefficients offer the advantage of using non-negative weights for slope combinations, preserving the sign of physical quantities. For instance, predicted changes in ray power remain negative, maintaining physical consistency.

The adaptive RKF method naturally incorporates energy absorption into error control. However, a special case arises where RKF advancement may produce incorrect nodes despite satisfying the error tolerance, as shown in Figure 6. This typically occurs at density discontinuities near solid target surfaces. All six RKF midpoint evaluations may lie entirely within the low-density region, causing the ray to erroneously overshoot into the super-critical density region instead of reflecting at the interface.

At sharp interfaces, extremely small RK steps would be needed to accurately capture trajectory curves, which is computationally inefficient and physically unrealistic. RKF failure can also be identified by violations of energy conservation at nodes ($v/c \neq N$). To address this, a velocity correction patch is applied when energy conservation is violated:

$$\vec{v}_{1c} = \vec{v}_1 + (1 - \text{sgn}(\vec{v}_1 \cdot (-\nabla n_{\text{e}}))) \left(\vec{v}_1 \cdot \frac{\nabla n_{\text{e}}}{\|\nabla n_{\text{e}}\|} \right) \left(\frac{-\nabla n_{\text{e}}}{\|\nabla n_{\text{e}}\|} \right).$$

This patch reflects the ray velocity $\vec{v}_1$ to $\vec{v}_{1c}$ if it forms an obtuse angle with the acceleration direction $-\nabla n_{\text{e}}$, bending it back toward the lower-density region. No correction is applied for acute angles. This patch is particularly important
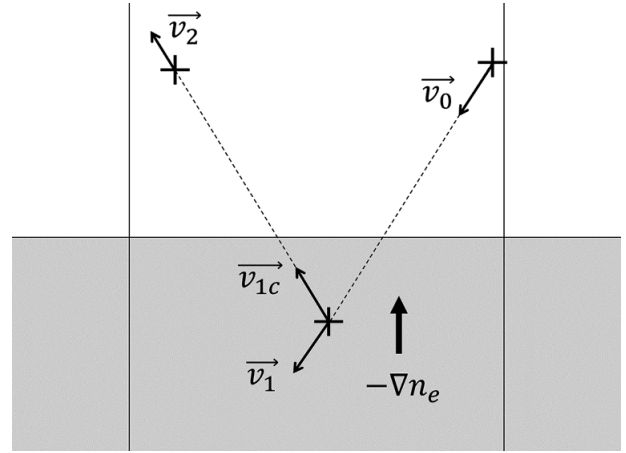

Figure 6. Ray reflection at a sharp interface. The lower part represents the high-density target. Here, $\vec{v}_0$ is the incident velocity, $\vec{v}_1$ the overshoot velocity, $\vec{v}_{1c}$ the corrected velocity and $\vec{v}_2$ the ejection velocity.

during the initial simulation frames and becomes unnecessary once plasma spans several cells.

The number of RKF steps for a single ray is approximately proportional to the cell count in the fluid domain, ensuring computational efficiency.

3.4. Rasterized deposition

The rasterization process handles individual RK line segments to allocate energy losses, as illustrated in Figure 7, which shows possible relative positions between line segments and 2D fluid cells:

(1) no crossing;

(2) single-cell-face crossing (X or Y);

(3) double-cell-face crossing (X–Y/Y–X).

When a line segment is divided into subsegments $\Delta l_1, \Delta l_2, \dots$ ($\Delta t = \Delta t_1 + \Delta t_2 + \dots$) by cell faces, the single-step energy deposition is distributed among subsegments as $\Delta P_1 \propto \overline{v}_{\text{ei}} P \Delta t_1, \Delta P_2 \propto \overline{v}_{\text{ei}} P \Delta t_2, \dots$, and so on. Since ray acceleration $\overline{g}$ remains constant within a single RK step, subsegment lengths can be expressed as $\Delta l_1 = v\Delta t_1 + 0.5\overline{g}\Delta t_1^2, \Delta l_2 = (v + \overline{g}\Delta t_1)\Delta t_2 + 0.5\overline{g}\Delta t_2^2, \dots$. The RKF method ensures $v > \overline{g}\Delta t$ everywhere except at normal-incidence reflection points. Neglecting higher-order terms of time, the approximation $\Delta P_1 : \Delta P_2 \approx \Delta l_1 : \Delta l_2$ indicates that energy deposition ratios across cells are proportional to subsegment lengths.

For 1D energy deposition, as shown in the flowchart in Figure 8, the process involves the following steps: given line segment endpoints $\vec{r}_0, \vec{r}_1$ and their corresponding cell indices i_0, i_1, potential cell face intersections are determined as $\vec{r}_x = X_{\text{face}}(\max(i_0, i_1))$ regardless of the ray's traveling direction. The array X_{face} stores the coordinates of all cell
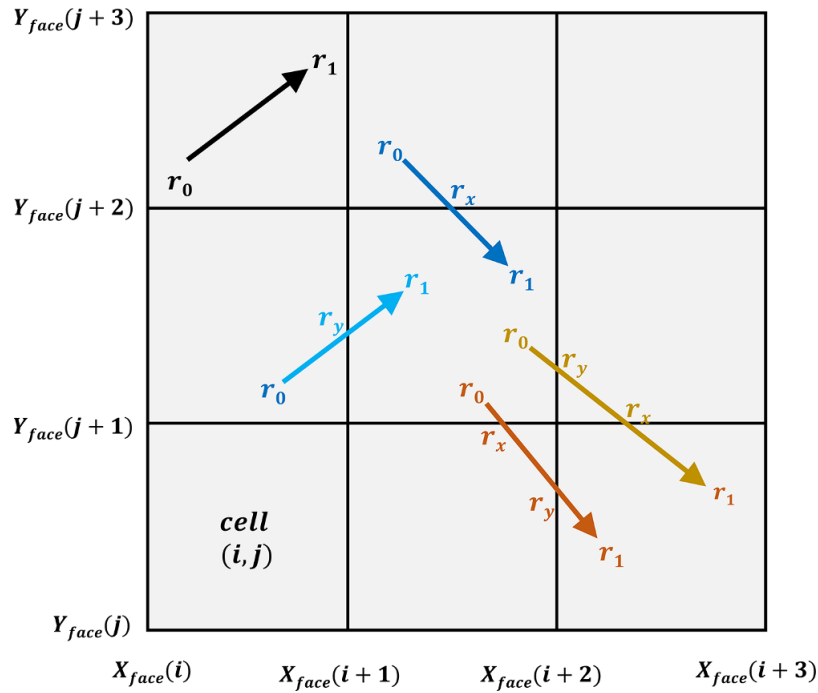
Figure 7. Possible scenarios of ray segments intersecting 2D fluid cells under the $\Delta t_{\max} = \Delta x / c$ constraint. Rectangular faces are shown, but sphere/cylinder faces are also applicable.
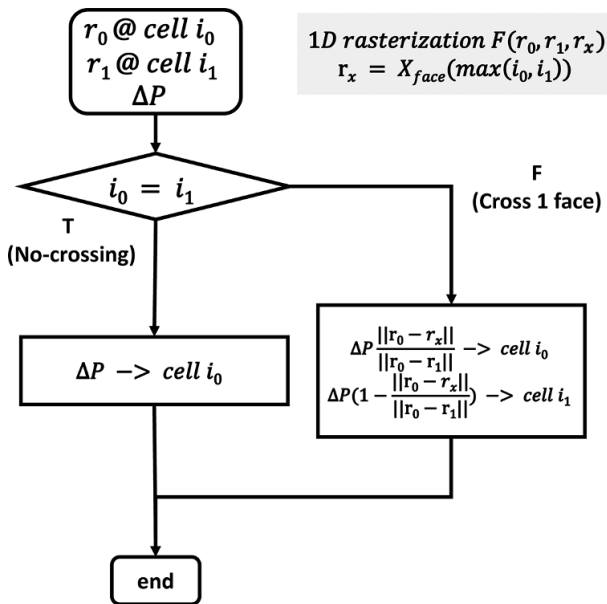

Figure 8. One-dimensional energy deposition rasterization allocation process.

(1) no crossing?

(2) single X-face crossing?

(3) single Y-face crossing?

(4) double-face crossing?

The first three cases reduce to 1D problems along the X- or Y-direction. For the fourth case (double-face crossing), intersections $\vec{r}_x, \vec{r}_y$ with X/Y faces are calculated using the same line–face intersection subroutine from the 'lens system' to determine the first intersected face.

(1) If the X-face is encountered first, the X-direction 1D problem is solved, then the starting point is updated $\vec{r}_0 \rightarrow \vec{r}_x$ and the Y-direction problem is solved with the remaining energy ΔP.

(2) If the Y-face is encountered first, the same logic applies, but in reverse order.

This approach ensures that 2D problems are always decomposed into a sequence of 1D solutions for computational simplicity and efficiency.

While the flowchart for 3D fluid cell energy allocation would be extensive, the actual implementation is concise. 1D problems involve two branches, 2D problems have four and 3D problems can be treated as an eight-branch structure using two-layer decompositions: 3D to 2D, and 2D to 1D. All branches ultimately reduce to 1D problems, allowing recursive logic to efficiently construct energy allocation algorithms for any dimension.
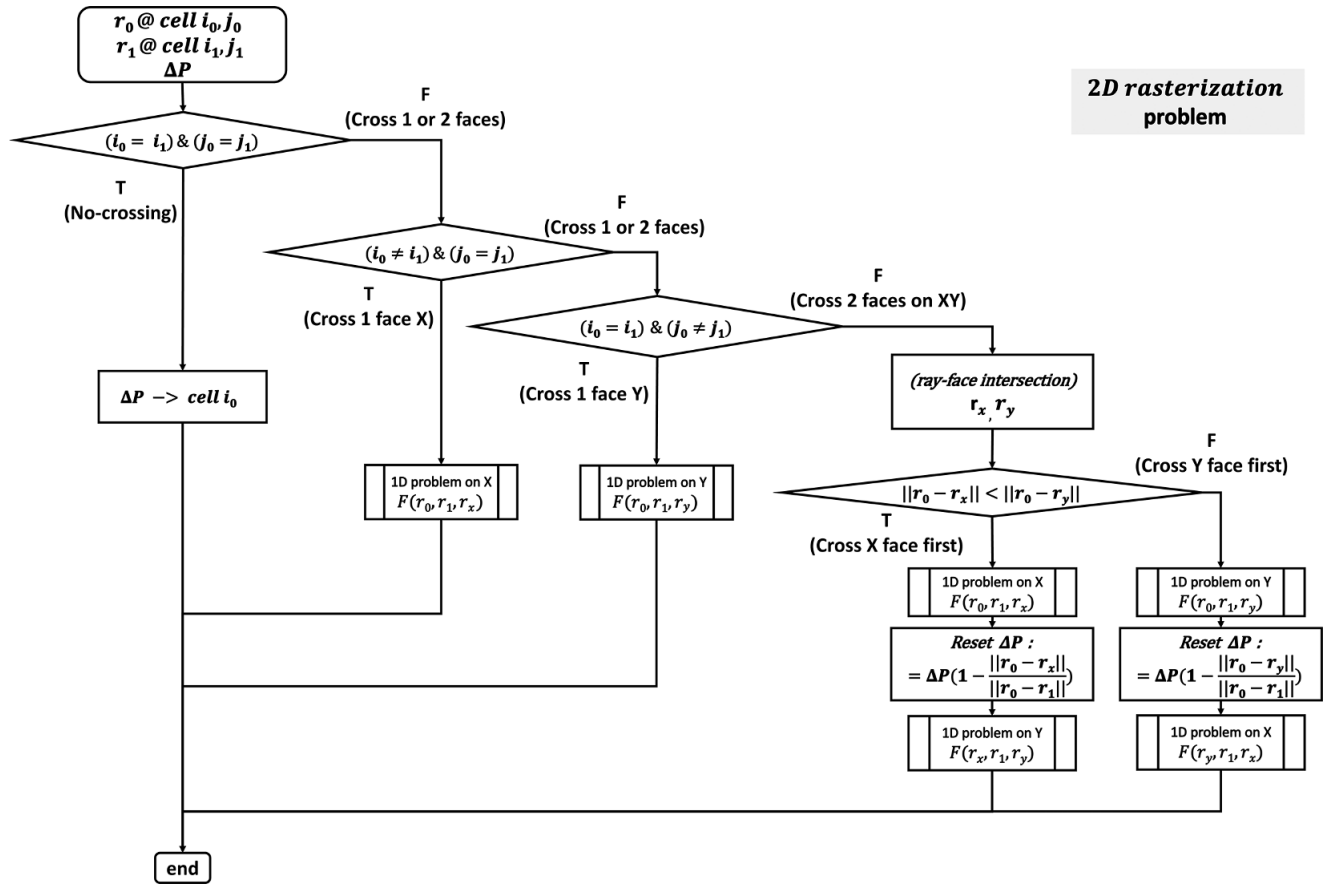
faces. If $r_0 = r_1$, indicating no crossing, the ray's energy loss is entirely deposited in local cell i_0. Otherwise, the energy is distributed between adjacent cells i_0 and i_1 in proportion to the subsegment lengths.

For 2D energy deposition, as outlined in the flowchart in Figure 9, the process begins by determining 2D cell indices i_0, j_0, i_1, j_1 from the endpoint coordinates $\vec{r}_0, \vec{r}_1$. The algorithm proceeds with branch case evaluation:

Figure 9. Two-dimensional energy deposition rasterization process.

Volume-averaged quantities, such as frequency shifts and polarization, can be computed using the same rasterization logic. By weighting with ray power first and then depositing onto cells, these quantities are easily calculated. For flux density, such as light intensity I, ray power through cell faces is tracked. At each interface crossing, ray power P is accumulated into the entered cell i_1. The intensity is then calculated as $I_{\text{cell}} = \Sigma_{\text{cross}} P_{\text{ray}} / S_{\text{proj}}$, where the projected area $S_{\text{proj}} = \cos(\min(\theta, \pi/2 - \theta)) \Delta X \Delta Y$ (in two dimensions), and θ is the angle between the ray and the cell face normal.

This concludes the description of an unconstrained, rasterization-based algorithm for ray tracing in gradient-index media. Aside from adaptive RK step-size adjustments, the algorithm is non-iterative and employs a unified tracing kernel applicable to all coordinate systems.

4. Test cases for rasterized ray tracing

4.1. Plasma Luneburg lens (ray 3D-in-1D sphere)

The Luneburg lens[32] is a spherical lens with a refractive index that gradually decreases from the center to the periphery. Parallel incident rays focus to a point on the lens surface, while a point source at the boundary produces parallel rays on the opposite side. This property makes the Luneburg lens ideal for beam forming and directional radiation applications. The refractive index distribution of a classical optical Luneburg lens is given by

$$N_{\text{opt}}(R) = \sqrt{2 - \left(\frac{R}{R_{\text{max}}}\right)^2},$$

where R is the radial coordinate of the spherical lens, with the refractive index N_{opt} maximized at the center and equal to unity at the boundary R_{max}.

Plasma refractive indices are inherently less than unity. However, it is possible to construct a 'plasma Luneburg lens' with equivalent focusing performance (for extended designs, see Ref. [33]). The refractive index distribution for a plasma Luneburg lens is given by the following:

$$N_{\text{pla}}(R) = \begin{cases} b\sqrt{2 - \left(\frac{R}{R_{\text{f}}}\right)^2}, & \text{if } R \leq R_{\text{f}}, \\ b, & \text{if } R > R_{\text{f}}, \end{cases} \quad (18)$$

where $b \in \left[1, 1/\sqrt{2}\right]$ is a scaling factor, and $R_{\text{f}} < R_{\text{max}}$ is the focal sphere radius. The corresponding electron density
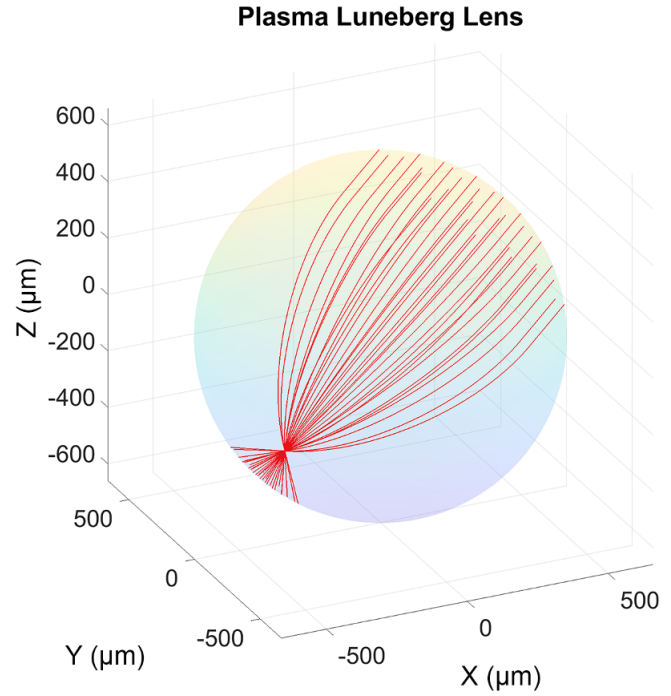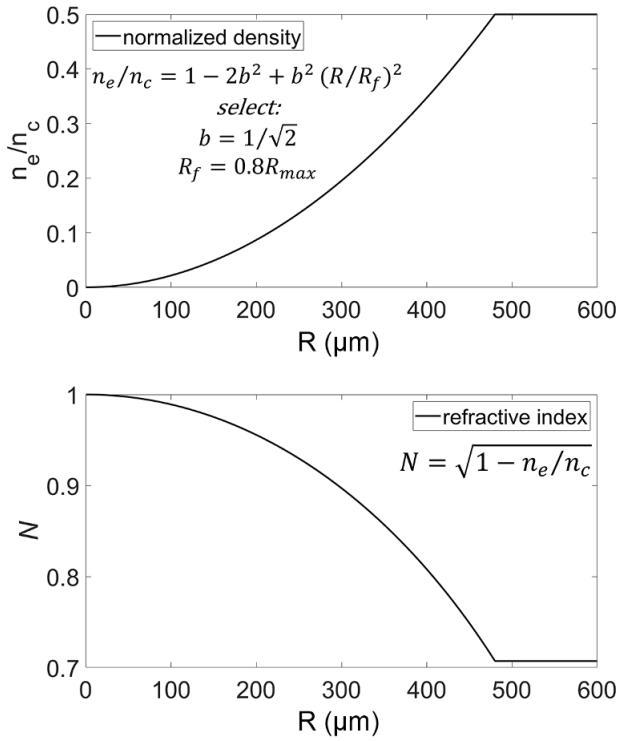
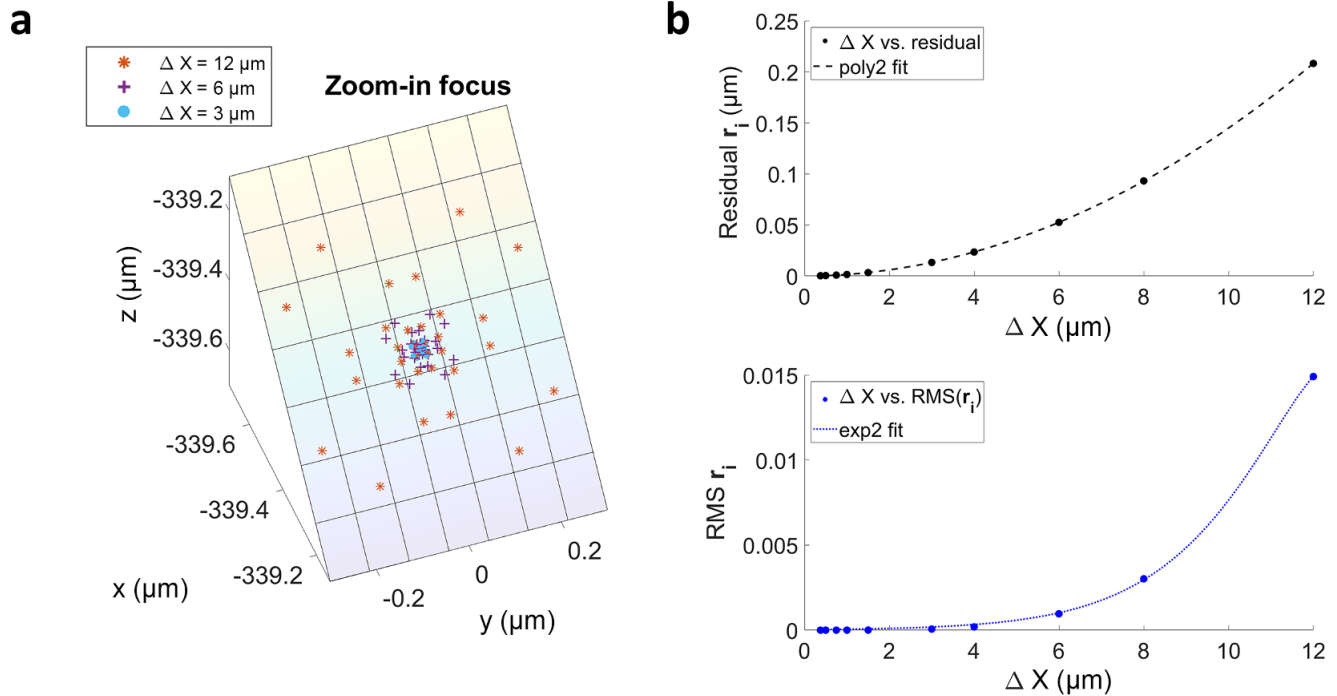**Figure 10.** Plasma Luneburg lens parameters and ray trajectories.

Figure 11. (a) Ray impact points near the focus. Shown are 32 rays per beam. (b) Error in impact point distribution versus cell grid spacing.

distribution is as follows:

$$n_e(R) = \begin{cases} n_c \left(1 - 2b^2 + b^2(R/R_f)^2\right), & \text{if } R \le R_f, \\ n_c \left(1 - b^2\right), & \text{if } R > R_f. \end{cases} \quad (19)$$

Figure 10 presents an example within a spherical computational domain of 600 μm diameter. A parallel beam ($\Phi = 400$ μm) is incident obliquely from above and focuses at $0.8R_{max}$ on the opposite side.
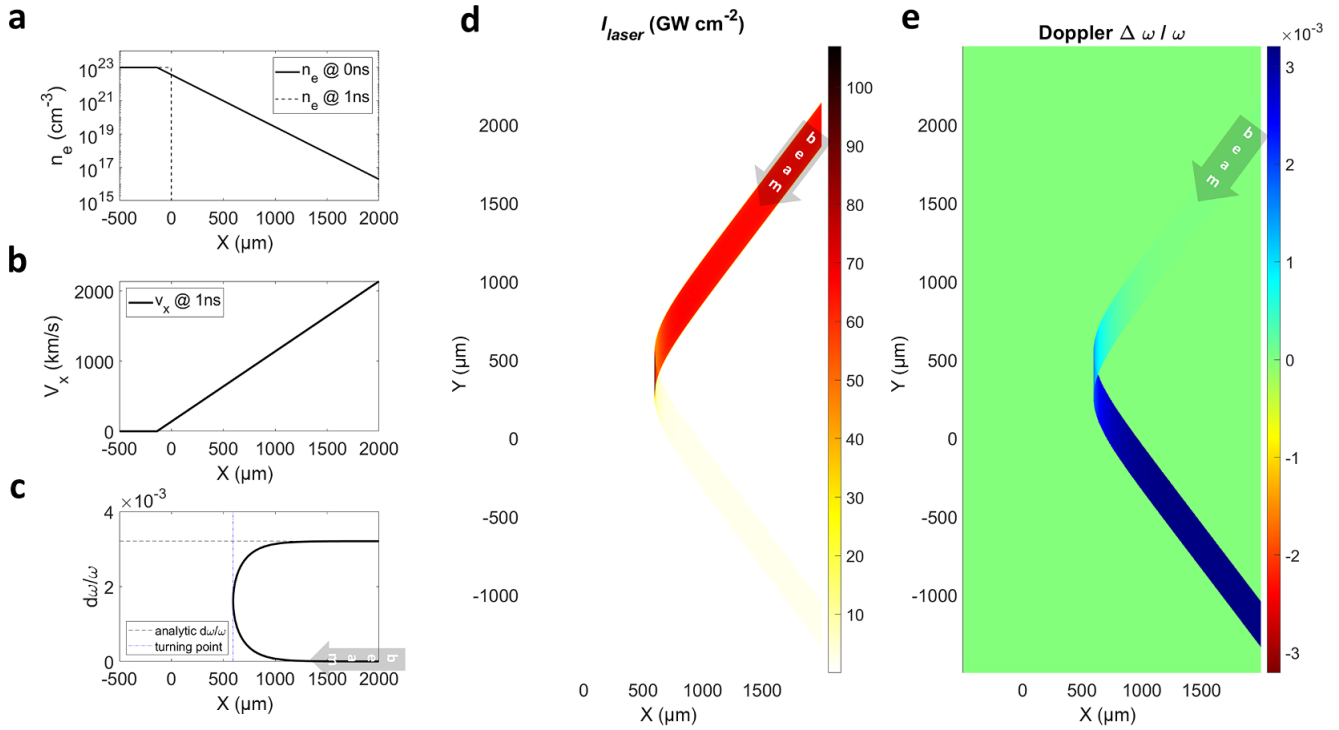
Figure 12. (a) Density and (b) flow velocity distributions of rarefied step-profile plasma. (c) Doppler frequency shifts during ray traversal through plasma. (d) Fluid domain light intensity. (e) Fluid domain frequency shift.

The plasma Luneburg lens provides a precision benchmark for ray-tracing algorithms. As shown in Figure 11(a), a grid with $\Delta X = 3\,\mu m$ (200 cells along R) achieves a focal spot smaller than $0.05\,\mu m$ in diameter. Errors, primarily from field interpolation and RK stepping, decrease with grid refinement. A grid spacing scan from 0.375 to 12 μm produces the error statistics in Figure 11(b).

(1) The residual $\Sigma_{ij,i>j}\left(\left\|\vec{r}_i - \vec{r}_j\right\|\right)/C^2_{n_{\text{ray}}}$, defined as the average distance between impact points i, j, $k, \dots$, converges at $\mathcal{O}\left(\Delta X^2\right)$.

(2) The root mean square (RMS) of impact point distribution shows exponential convergence, indicating a significantly higher ray density at the focal spot center compared to the surrounding halo.

This highlights the algorithm's precision and its efficiency in capturing fine-scale focusing behavior.

4.2. Plasma Doppler shift (ray 3D-in-2D plane)

When laser light propagates in the plasma, Doppler frequency shifts occur due to plasma motion[34]. These Doppler shifts influence the characteristics of laser–plasma interactions. In the fluid reference frame, the light wave frequency can be interpreted as the number of electric field peaks detected per unit time by a probe at the cell center: blue shifts occur when the phase velocity in plasma shows an

increasing trend (analogous to an accelerating wave-peak conveyor) or when reflecting surfaces move toward the wave source, resulting in more wave peaks detected per unit time; red shifts occur under the opposite conditions. The Doppler frequency shift is quantitatively described by the following:

$$\frac{\Delta\omega}{\omega} = -\frac{1}{c}\int_{\text{ray}} |\,\mathrm{d}\vec{r}\,| \frac{\partial N(\vec{r},t)}{\partial t}, \tag{20}$$

where ω is the central frequency. Integration along the ray path provides the frequency shift at any location, assuming the shift is small relative to the central frequency. Using the fluid continuity equation, the rate of change of the refractive index can be expressed in terms of the flow field divergence:

$$\frac{\partial N}{\partial t} = \frac{\partial\sqrt{1 - n_e/n_c}}{\partial t},$$
$$\frac{\partial n_e}{\partial t} = -\nabla\cdot(n_e\vec{v}).$$

The frequency shift in a fluid cell is calculated as the power-weighted average of all incident rays:

$$\Delta\omega_{\text{cell}} = \frac{\Sigma_{\text{cross}}\Delta\omega_{\text{ray}}P_{\text{ray}}}{\Sigma_{\text{cross}}P_{\text{ray}}}. \tag{21}$$

Figure 12 shows Doppler frequency shifts in a 2D planar rarefaction wave. Using the ideal gas equation of state for fully ionized hydrogen plasma, the density and velocity
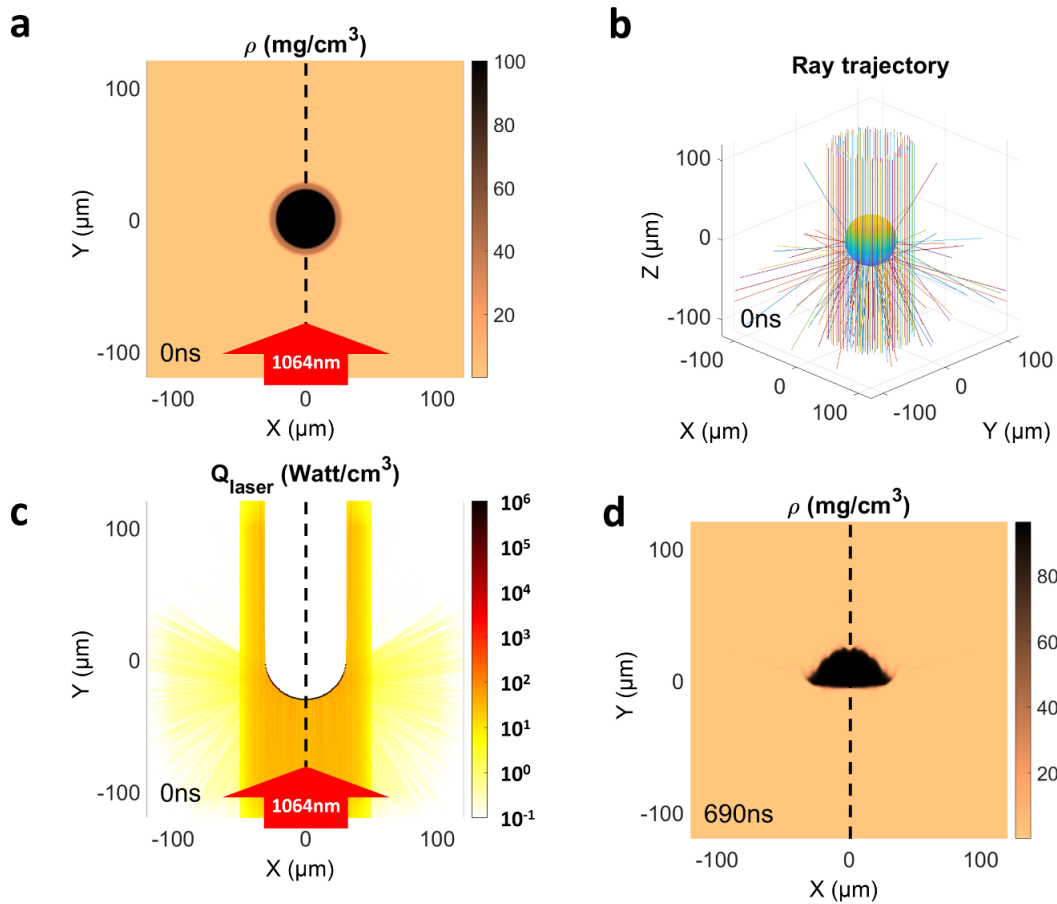
Figure 13. Laser-driven spherical water vapor target in $R - Z$ cylindrical coordinates: (a) initial density distribution, half-space mirrored; (b) 3D ray trajectories recorded in Cartesian coordinates; (c) laser volumetric heating power; (d) density distribution at 690 ns.

follow self-similar solutions:

$$n_e(x,t) = n_{e0}\exp\left(-\frac{x}{c_T t} - 1\right), \quad \text{if } x > -c_T t,$$

$$v_x(x,t) = \frac{x}{t} + c_T, \quad \text{if } x > -c_T t,$$

where c_T is the sound speed. For a step density $n_{e0} = 100 n_c$, temperature $T_0 = 100$ eV, and 1 ns rarefaction time, a $\lambda = 1064$ nm beam is incident at $45°$ from the right-hand boundary. Figure 12(c) depicts the frequency shift along the ray path. At a fluid cell spacing of $\Delta X = 3.125$ μm, ray tracing predicts $\Delta\omega/\omega = 3.2069 \times 10^{-3}$, corresponding to a wavelength shift of $\Delta\lambda = 34.12145$ Å. The analytical solution (Equation (6) in Ref. [34]), $\Delta\omega/\omega = (2\cos\theta/c)(v_{TP} + 2(1 - \ln2)c_T)$, gives $\Delta\lambda = 34.12464$ Å, where v_{TP} is the turning point velocity. The relative error is approximately 10^{-4}.

The Doppler shift is dominated by the near-critical to super-critical density transition region near the turning point. Accurate results require smooth ray trajectories in this region. In the cell-AVG algorithm, the density is discontinuous at the cell interface, leading to frequency shift errors up to 18% under identical conditions.

Figure 12(d) shows the fluid domain light intensity, with an incident power density of approximately 100 GW/cm^2 and about 95% energy absorption by the plasma. Figure 12(e) presents the frequency shift distribution in the fluid domain, indicating an overall blue shift after reflection.

4.3. Laser-driven gas sphere (ray 3D-in-2D cylinder)

The rasterization algorithm has been integrated into the multiphase RHD code DRIM (details to be published). A full-physics simulation of a laser-driven gas sphere demonstrates the robustness and efficiency of the laser module. The laser parameters are: 1064 nm wavelength, 80 μm focal spot diameter and 0.5 MW square pulse lasting 1 ns. The water vapor sphere has a core density of 100 mg/cm^3 within 25 μm radius, a 5 μm buffer layer at 40 mg/cm^3 and a background density of 1 μg/cm^3.

The simulation traces 3D laser beams in a 2D cylindrical fluid domain, allowing energy deposition for multiple beams at arbitrary incidence angles. Figure 13(b) shows 3D ray trajectories. Initially, the sharp density transition between the core and background causes velocity-corrected ray reflection

at the interface. The deposited energy is azimuthally averaged before coupling to the fluid. Figure 13(c) shows significant energy deposition at the sphere surface, driving material ablation and inward compression waves. After multiple internal reflections, these waves induce global deformation, as shown in Figure 13(d). Statistical analysis reveals that less than 10% of the laser energy converts to the gas target's internal and kinetic energy.

The principle of parallelizing ray tracing is to minimize the waiting time for each ray process. In cell-based algorithms (e.g., the cell-AVG method used in FLASH), the computational domain is spatially divided into many blocks (each containing multiple cells), and the smallest computational step involves a ray traversing a single block. This often leads to an inefficient situation where a few blocks have a large number of ray-tracing tasks, while most blocks have no rays at all but still need to wait due to parallel barriers. Our algorithm is slightly different: ray tracing adopts a hybrid Message Passing Interface (MPI)+OpenMP strategy. Each MPI task corresponds to one computational node (with multiple central processing unit (CPU) cores), which communicates via interlink to obtain the complete interpolated field information and is assigned a batch of rays to process. For example, if there are 4096 ray-tracing tasks, they can be distributed across eight nodes, with each MPI task processing a batch of 512 rays. Within each batch, OpenMP is used to trace each ray in parallel, achieving fine-grained task decomposition. If the communication time for the interpolated field is relatively low, the MPI+OpenMP strategy can achieve better load balancing.

MPI tasks access the full interpolation field and trace multiple rays per batch, with batch-wise depositions summed for the total energy. On an Intel Xeon 6326 CPU (2.9 GHz) with a 500×500 grid and 1024 rays per batch, wall-clock times for 1, 2, 4, 8 and 16 processes are 16.106, 8.622, 4.904, 3.046 and 2.219 seconds, respectively, demonstrating near-linear speedup. Convergence studies indicate gas target simulations require at least 16,384 rays, utilizing 256 CPU cores. For 3D fluid domains, the ray count requirement exceeds 10^6 rays per step.

5. Summary

This paper introduces a novel rasterization-based ray-tracing methodology that addresses key limitations of traditional cell-confined algorithms in gradient-index materials. The development of an adaptive-step integration approach eliminates the need for rays to terminate at cell interfaces, avoiding costly iterative cross-point calculations. A rasterization deposition method is introduced, decoupling the tracing algorithm from domain-specific geometries and enabling a unified, simplified computational framework for various curvilinear coordinate systems. The method's effectiveness is demonstrated through numerical experiments, showcasing robustness, rapid convergence and accurate handling of diverse ray physics, including beam intensity and frequency shifts. Simulations of planar, cylindrical and spherical laser-ablation scenarios further indicate its enormous potential in engineering applications.

Future developments may include extending the method to more complex optical phenomena, such as non-linear laser–plasma interactions, incorporating modules for synthetic optical diagnostics and optimizing for large-scale simulations.

References

1. S. Atzeni and J. Meyer-ter Vehn, *The Physics of Inertial Fusion: Beam Plasma Interaction, Hydrodynamics, Hot Dense Matter*, volume **125** (Oxford University Press, 2004).
2. W. R. Hamilton, in *The Transactions of the Royal Irish Academy* (JSTOR, 1828), pp. 69–174.
3. H. A. Buchdahl, *An Introduction to Hamiltonian Optics* (Taylor & Francis, 1993).
4. B. Crinquand, B. Cerutti, G. Dubus, K. Parfrey, and A. Philippov, Phys. Rev. Lett. **129**, 205101 (2022).
5. S. Ji, M. Ponting, R. S. Lepkowicz, A. Rosenberg, R. Flynn, G. Beadie, and E. Baer, Opt. Express **20**, 26746 (2012).
6. A. Demetriadou and Y. Hao, Opt. Express **19**, 19925 (2011).
7. W. Liu, H. Hu, F. Liu, and H. Zhao, Opt. Express **27**, 4714 (2019).
8. C. Flynn and A. V. Goncharov, Appl. Opt. **63**, 290 (2023).
9. R. Craxton, K. Anderson, T. Boehly, V. Goncharov, D. Harding, J. Knauer, R. McCrory, P. McKenty, D. Meyerhofer, J. Myatt, and A. Schmitt, Phys. Plasmas **22**, 110501 (2015).
10. R. Betti and O. Hurricane, Nat. Phys. **12**, 435 (2016).
11. P. B. Radha, V. N. Goncharov, T. J. B. Collins, J. A. Delettrez, Y. Elbaz, V. Y. Glebov, R. L. Keck, D. E. Keller, J. P. Knauer, J. A. Marozas, F. J. Marshall, P. W. McKenty, D. D. Meyerhofer, S. P. Regan, T. C. Sangster, D. Shvarts, S. Skupsky, Y. Srebro, R. P. J. Town, and C. Stoeckl, Phys. Plasmas **12**, 032702 (2005).
12. D. Turnbull, J. Katz, M. Sherlock, L. Divol, N. R. Shaffer, D. J. Strozzi, A. Colaïtis, D. H. Edgell, R. K. Follett, K. R. McMillen, P. Michel, A. L. Milder, and D. H. Froula, Phys. Rev. Lett. **130**, 145103 (2023).
13. B. M. Haines, D. Keller, J. Marozas, P. McKenty, K. Anderson, T. Collins, W. Dai, M. Hall, S. Jones, M. McKayJr, R. Rauenzahn, and D. Woods, Comput. Fluids **201**, 104478 (2020).
14. A. Colaïtis, T. Chapman, D. Strozzi, L. Divol, and P. Michel, Phys. Plasmas **25**, 033114 (2018).

15. A. Colaïtis, G. Duchateau, P. Nicolaï, and V. Tikhonchuk, Phys. Rev. E **89**, 033101 (2014).
16. J. A. Marozas, M. Hohenberger, M. J. Rosenberg, D. Turnbull, T. J. B. Collins, P. B. Radha, P. W. McKenty, J. D. Zuegel, F. J. Marshall, S. P. Regan, T. C. Sangster, W. Seka, E. M. Campbell, V. N. Goncharov, M. W. Bowers, J.-M. G. Di Nicola, G. Erbert, B. J. MacGowan, L. J. Pelz, J. Moody, and S. T. Yang, Phys. Plasmas **25**, 056314 (2018).
17. A. Colaïtis, S. Hüller, D. Pesme, G. Duchateau, and V. Tikhonchuk, Phys. Plasmas **23**, 032118 (2016).
18. J. Dexter and E. Agol, Astrophys. J. **696**, 1616 (2009).
19. B. McKeon and A. V. Goncharov, Appl. Opt. **62**, 8621 (2023).
20. R. Ramis, J. Meyer-ter Vehn, and J. Ramírez, Comput. Phys. Commun. **180**, 977 (2009).
21. B. Fryxell, K. Olson, P. Ricker, F. X. Timmes, M. Zingale, D. Lamb, P. MacNeice, R. Rosner, J. Truran, and H. Tufo, Astrophys. J. Suppl. Ser. **131**, 273 (2000).
22. Flash Center for Computational Science, *Flash User's Guide* (University of Rochester, 2024).
23. Q. Min, Z. Xu, S. He, H. Lu, X. Liu, R. Shen, Y. Wu, Q. Pan, C. Zhao, F. Chen, M. Su, and C. Dong, Comput. Phys. Commun. **302**, 109242 (2024).
24. M. Gittings, R. Weaver, M. Clover, T. Betlach, N. Byrne, R. Coker, E. Dendy, R. Hueckstaedt, K. New, W. R. Oakes, D. Ranta, and R. Stefan, Comput. Sci. Discovery **1**, 015005 (2008).
25. D. Keller, T. Collins, J. Delettrez, P. McKenty, P. Radha, R. Town, B. Whitney, and G. Moses, in *APS Division of Plasma Physics Meeting Abstracts*, volume **41** (American Physics Society, 1999), paper BP1–39.
26. J. A. Marozas, F. J. Marshall, R. S. Craxton, I. V. Igumenshchev, S. Skupsky, M. J. Bonino, T. J. B. Collins, R. Epstein, V. Y. Glebov, D. Jacobs-Perkins, J. P. Knauer, R. L. McCrory, P. W. McKenty, D. D. Meyerhofer, S. G. Noyes, P. B. Radha, T. C. Sangster, W. Seka, and V. A. Smalyuk, Phys. Plasmas **13**, 056311 (2006).
27. T. B. Kaiser, Phys. Rev. E **61**, 895 (2000).
28. E. Fehlberg, *Low-Order Classical Runge–Kutta Formulas with Stepsize Control and Their Application to Some Heat Transfer Problems*, Vol. **315** (National Aeronautics and Space Administration, 1969).
29. A. Colaïtis, J. Palastro, R. Follett, I. Igumenschev, and V. Goncharov, Phys. Plasmas **26**, 032301 (2019).
30. E. E. Catmull, *A Subdivision Algorithm for Computer Display of Curved Surfaces*, PhD Dissertation (The University of Utah, 1974).
31. J. D. Foley, *Computer Graphics: Principles and Practice* (Addison-Wesley Professional, 1996).
32. R. K. Luneburg, *Mathematical Theory of Optics* (University of California Press, 1966).
33. A. Gutman, J. Appl. Phys. **25**, 855 (1954).
34. O. Landen and W. Alley, Phys. Rev. A **46**, 5089 (1992).