FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

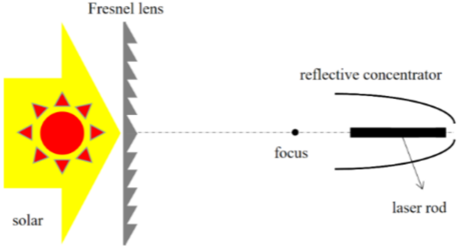
Ray Tracing Simulation of optically pumped Laser Crystals

Master's Thesis

Matthias Koenig
Chair for Computer Science 10, System Simulation, Friedrich-Alexander University of Erlangen-Nuremberg

May 15, 2022

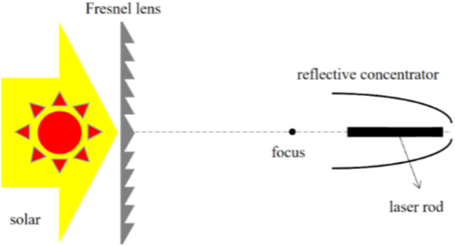
Motivation



Problems to solve:

1. Build a framework for physically accurate raytracing
2. Calculate absorbed power
3. Optimize mirror shape

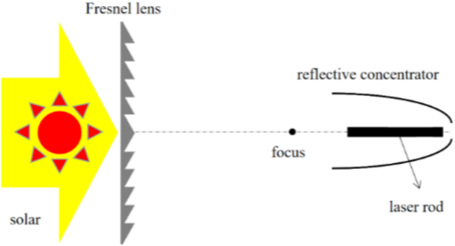
Motivation



Problems to solve:

1. Build a framework for physically accurate raytracing
2. Calculate absorbed power
3. Optimize mirror shape

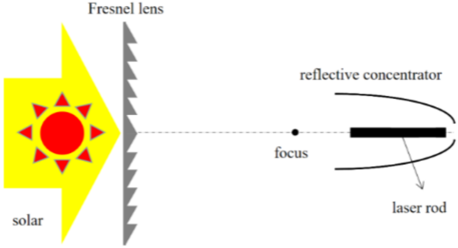
Motivation



Problems to solve:

1. Build a framework for physically accurate raytracing
2. Calculate absorbed power
3. Optimize mirror shape

Motivation



Problems to solve:

1. Build a framework for physically accurate raytracing
2. Calculate absorbed power
3. Optimize mirror shape

Outline

Optics

Reflection

A ray is reflected by creating a new ray with the origin at the intersection point and the direction determined by the incident angle.

$$\theta_1 = \theta_2$$

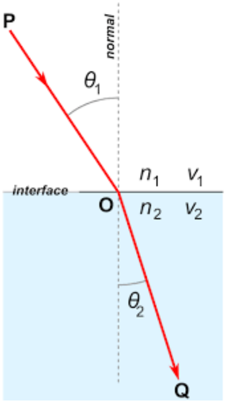
where θ_1 is the incident angle and θ_2 is the reflection angle.

Refraction

Refraction is modelled accurately by Snells' law:

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2)$$

where n_1, n_2 are the indices of refraction.



Fresnel Laws

The transmitted and reflected power can be calculated with the transmission- and reflection rates given by Fresnel's laws.
These are dependent on the orientation of the polarization of the incident ray (perpendicular or parallel) to the surface:

$$R_{\perp} = \frac{\sin^2(\theta_1 - \theta_2)}{\sin^2(\theta_1 + \theta_2)}$$

$$R_{\parallel} = \frac{\tan^2(\theta_1 - \theta_2)}{\tan^2(\theta_1 + \theta_2)}$$

$$T_{\perp} = 1 - R_{\perp}$$

$$T_{\parallel} = 1 - R_{\parallel}$$

Fresnel Laws contd.

For now unpolarized light is assumed and only one refraction takes place so the total rates are:

$$R_{total} = \frac{R_\perp + R_\parallel}{2}$$

$$T_{total} = \frac{T_\perp + T_\parallel}{2}$$

Sellmeier Equation

Dependency of the refractive index on the wavelength of light is modelled using the Sellmeier equation.

$$n^2(\lambda) = 1 + \sum_i \frac{B_i \lambda^2}{\lambda^2 - C_i}$$

Here the B_i and C_i are empirically determined coefficients.

Ray Tracing

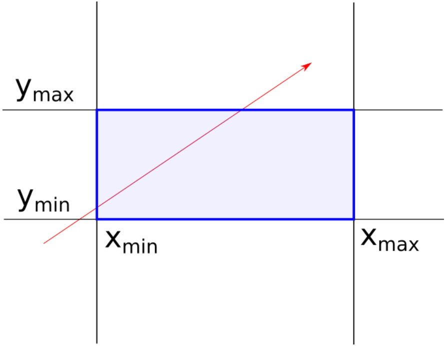
Parametrization

All points along a ray are described as follows:

$$\boldsymbol{r}(t) = \boldsymbol{o} + t \cdot \boldsymbol{d}$$

Testing intersections against primitives involves solving for the parameter t.

Example: Axis aligned box intersection



Parametrization contd.

Solution:

```
    float tx1 = (xmin - ray.origin.x) / ray.direction.x;
    float tx2 = (xmax - ray.origin.x) / ray.direction.x;

    float tmin = min(tx1, tx2);
    float tmax = max(tx1, tx2);

    float ty1 = (ymin - ray.origin.y) / ray.direction.y;
    float ty2 = (ymax - ray.origin.y) / ray.direction.y;

    tmin = max(tmin, glm::min(ty1, ty2));
    tmax = min(tmax, glm::max(ty1, ty2));
```

Other primitives in 2D can be lines, cricles, etc.
Or in 3D triangles, quads, spheres, etc.

All objects in a scene need to be built with a collection of such primitives.

Scene Tracing

If a ray hits an object new rays are genertated according to its type of surface (reflection, refraction).

These new rays are traced again through the scene.

$\Longrightarrow$ Recurse until a desired "depth".

An object is intersected if one of its primitives is hit.

$\Longrightarrow$ Need to check each primitive of every object in the scene.

Runtime of a scene tracing step with N objects with M primitives each:

$$O\left(N * M\right)$$

Hierarchical Bounding Volumes

Performance optimization:

1. Preprocessing: Attach a bounding box around each object and recursively subdivide.
2. Tracing: Check if ray hits bounding box. If yes recursively check its subdivisions.

Runtime of a scene tracing step with N objects with M primitives each and 5 recursive subdivisions:

$$O(N * (5 * 4 + M/4^5)) = O(N * M/1024)$$

Hierarchical Bounding Volumes

Performance optimization:

1. Preprocessing: Attach a bounding box around each object and recursively subdivide.
2. Tracing: Check if ray hits bounding box. If yes recursively check its subdivisions.

Runtime of a scene tracing step with N objects with M primitives each and 5 recursive subdivisions:

$$O(N * (5 * 4 + M/4^5)) = O(N * M/1024)$$

Hierarchical Bounding Volumes

Performance optimization:

1. Preprocessing: Attach a bounding box around each object and recursively subdivide.
2. Tracing: Check if ray hits bounding box. If yes recursively check its subdivisions.

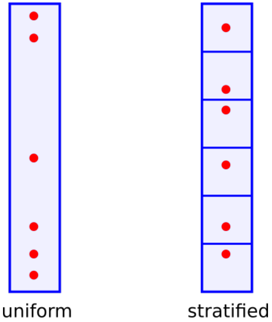
Runtime of a scene tracing step with N objects with M primitives each and 5 recursive subdivisions:

$$O(N * (5 * 4 + M/4^5)) = O(N * M/1024)$$

Generating Rays and Random Sampling

The goal is to randomly generate a cone of rays originating in the focus of the fresnel lense.

$\Longrightarrow$ Uniformly sample the opening angle around a direction vector.



$\Longrightarrow$ Not ideal in this case (big gaps between rays)

$\Longrightarrow$ Better: Stratified Uniform Sampling

Inversion Method

In reality sunlight consists of unpolarized light with a specific frequency spectrum. Thus the rays need to carry information about their power, frequency and polarity.

$\Longrightarrow$ Need mechanism to generate random samples x according to a given distribution density function $p(x)$ (gauss, poisson, sun spectrum, etc.)

Inversion Method:

1. Integrate(sum up) the distribution $p(x)$ in uniform steps x and save the value for each step resulting in $P(x)$.
2. Uniformly sample $\xi \in [0, 1]$ and figure out in which interval it lies.
3. Interpolate linearly within the interval and return resulting x value.

Frequencies and polarisations of rays are not implemented as of yet.

Inversion Method

In reality sunlight consists of unpolarized light with a specific frequency spectrum. Thus the rays need to carry information about their power, frequency and polarity.

$\Longrightarrow$ Need mechanism to generate random samples x according to a given distribution density function $p(x)$ (gauss, poisson, sun spectrum, etc.)

Inversion Method:

1. Integrate(sum up) the distribution $p(x)$ in uniform steps x and save the value for each step resulting in $P(x)$.
2. Uniformly sample $\xi \in [0, 1]$ and figure out in which interval it lies.
3. Interpolate linearly within the interval and return resulting x value.

Frequencies and polarisations of rays are not implemented as of yet.

Inversion Method

In reality sunlight consists of unpolarized light with a specific frequency spectrum. Thus the rays need to carry information about their power, frequency and polarity.

$\Longrightarrow$ Need mechanism to generate random samples x according to a given distribution density function $p(x)$ (gauss, poisson, sun spectrum, etc.)

Inversion Method:

1. Integrate(sum up) the distribution $p(x)$ in uniform steps x and save the value for each step resulting in $P(x)$.
2. Uniformly sample $\xi \in [0, 1]$ and figure out in which interval it lies.
3. Interpolate linearly within the interval and return resulting x value.

Frequencies and polarisations of rays are not implemented as of yet.

Inversion Method

In reality sunlight consists of unpolarized light with a specific frequency spectrum. Thus the rays need to carry information about their power, frequency and polarity.

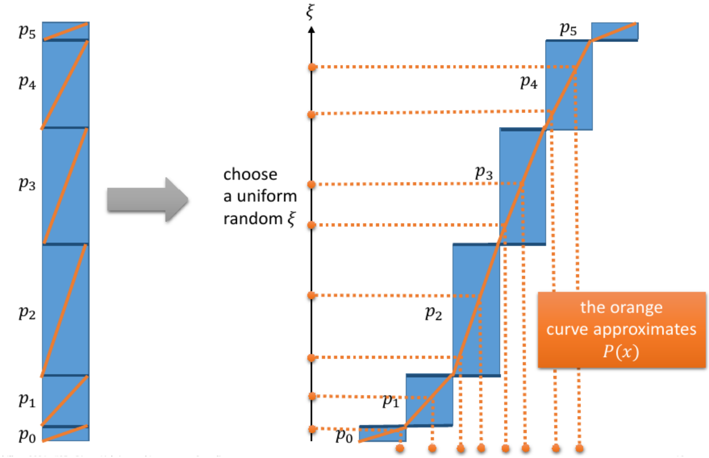
$\Longrightarrow$ Need mechanism to generate random samples x according to a given distribution density function $p(x)$ (gauss, poisson, sun spectrum, etc.)

Inversion Method:

1. Integrate(sum up) the distribution $p(x)$ in uniform steps x and save the value for each step resulting in $P(x)$.
2. Uniformly sample $\xi \in [0, 1]$ and figure out in which interval it lies.
3. Interpolate linearly within the interval and return resulting x value.

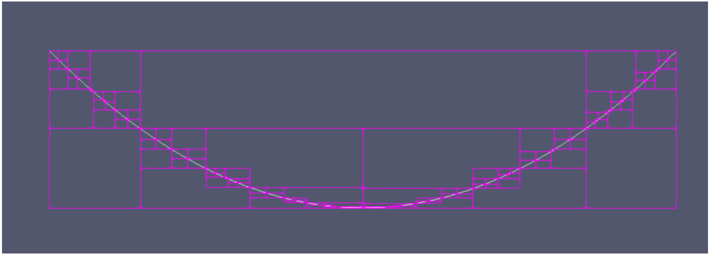
Frequencies and polarisations of rays are not implemented as of yet.

Inversion Method contd.



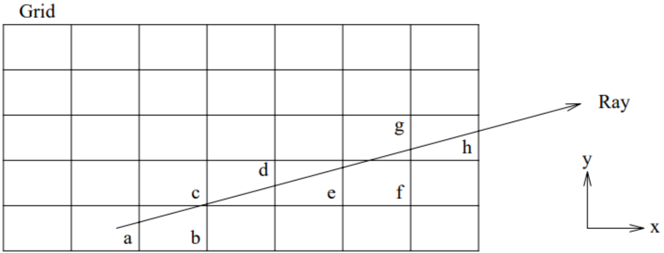
Mirror

Mirror consists of 2D line segments arranged by a 1D shape function (parabolic for testing purposes).



Crystal

The laser crystal is a 2D Box with an internal grid structure and grid tracing algorithm.
Rays need to be traced through cells **in order**[1] because of the absorbed energy calculation.



[1] A Fast Voxel Traversal Algorithm for Ray Tracing, John Amanatides, Andrew Woo, University of Toronto

Calculating Absorbed Power

The remaining power of a ray passing through the crystal is calculated by the Lambert law of absorption:

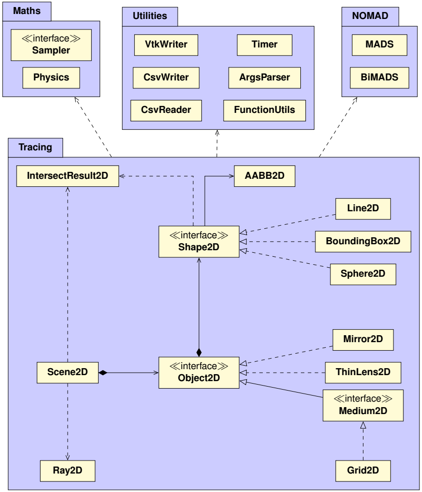
$$I_{out} = I_{in} \cdot e^{-\alpha d}$$

where α is the absorption coefficient d is the distance travelled through a cell. Thus the absorbed power is:

$$I_{abs} = I_{in} - I_{out}$$

In reality the α is frequency dependent but this is not implemented yet.

Framework Overview



Optimization

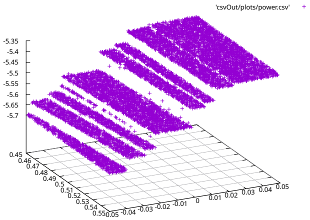
Problem

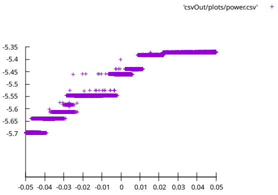
Goal: Optimize both total absorbed power and variance across the crystal

⇒ Should result in a better and more powerful beam (verification in ASLD)

⇒ Need an algorithm to handle two objective functions at the same time (biobjective optimization)

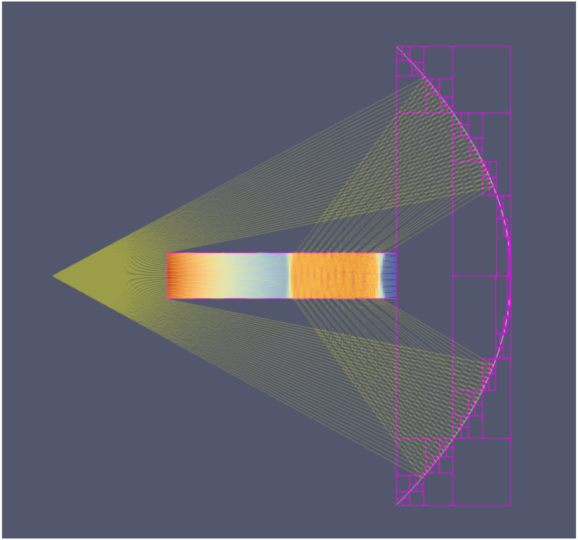
Additional problem: noisy and nonsmooth objective functions!





Results

Results



Outlook

Thanks for listening.

Any questions/suggestions?

References

References I

[1] A. W. John Amanatides, *A Fast Voxel Traversal Algorithm for Ray Tracing*. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.42.3443&rep=rep1&type=pdf.

[2] M. Stamminger, *Global Illumination SS21*. 2021. [Online]. Available: https://www.studon.fau.de/crs3792557.html.