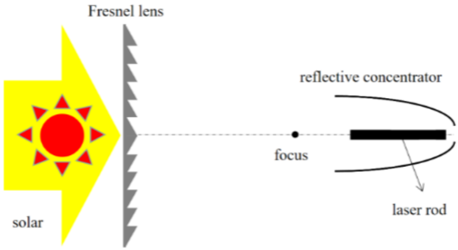
Ray Tracing Simulation of optically pumped Laser Crystals

Master's Thesis

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

May 19, 2022

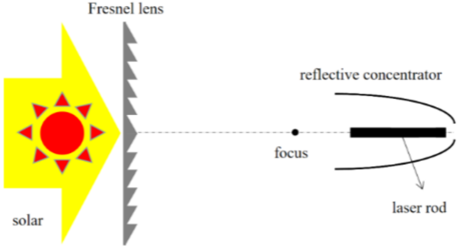
Motivation



Problems to solve:

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

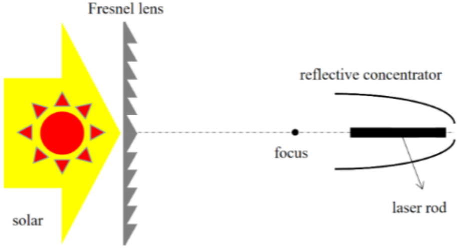
Motivation



Problems to solve:

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

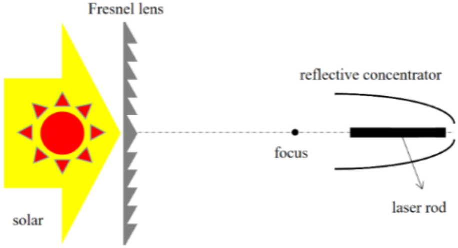
Motivation



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

Ray Tracing

Optimization

Results

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.

$$\theta_1 = \theta_2$$

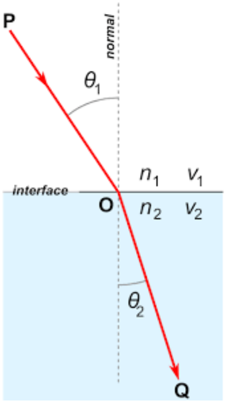
where θ_1 is the incident angle and θ_2 is the reflection angle.

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

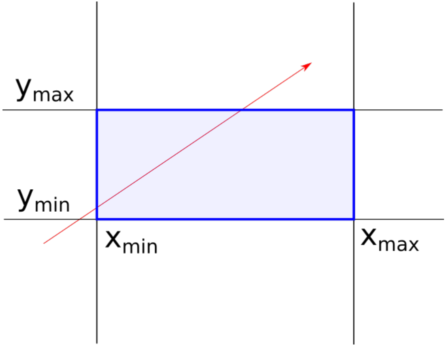
Parametrization

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:

$$O(N * M)$$

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.

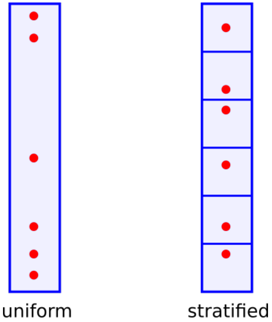
Runtime of a scene tracing step with N objects with M primitives each and 5 recursive subdivisions:

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

Generating Rays and Random Sampling

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

$\Longrightarrow$ 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.

$\implies$ 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.

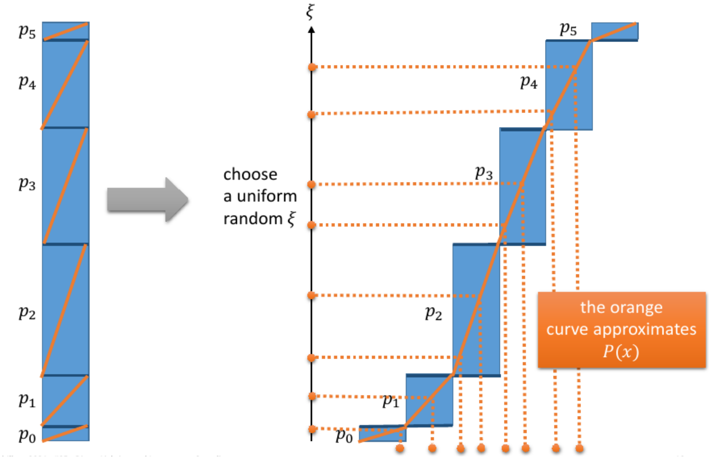
$\Longrightarrow$ Need mechanism to generate random samples x according to a given distribution density function $p(x)$ (gauss, poisson, sun spectrum, etc.)

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.

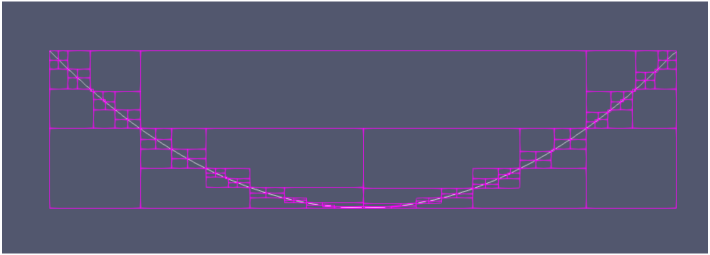
Frequencies and polarisations of rays are not implemented as of yet.

Inversion Method contd.



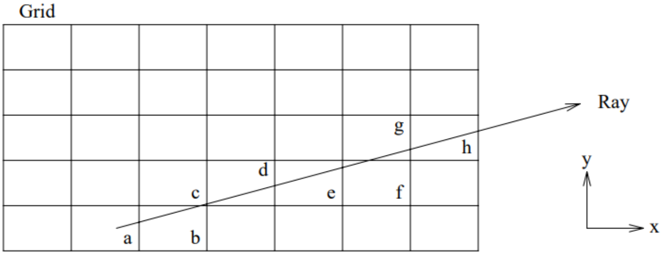
Mirror

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:

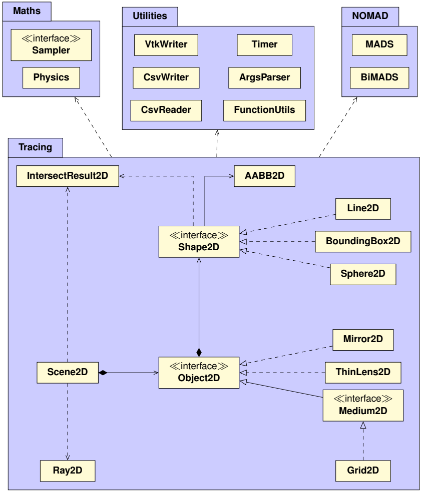
$$I_{out} = I_{in} \cdot e^{-\alpha d}$$

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

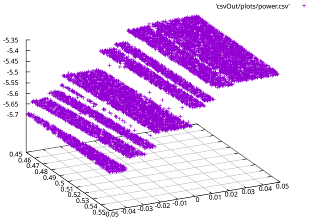
Problem

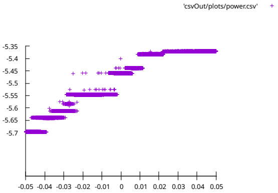
Goal: Optimize both total absorbed power and variance across the crystal

⇒ 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, computationally expensive objective functions!





Derivative-Free Optimization

Minimization problem:

$$\text{Find} \quad \mathbf{x}_{min} \in \Omega \subseteq \mathbb{R}^n \quad \text{s.t.}$$
$$f(\mathbf{x}_{min}) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega$$

Global optimization is not possible anyway for non-convex functions.
Problem: Gradient is not available, or easily computable!
Use Derivative-Free algorithms, not relying on derivatives of objective functions

$\Rightarrow$ Should be used as last resort, only if little or no information can be exploited

$\Rightarrow$ Objectives are treated as black-box functions, where the goal is to also use as little actual evaluations as possible (caching, etc.)

$\Rightarrow$ Optimality is often defined in an alternative way, to e.g. gradient search algorithms

Types of BBO: Simplex-Methods, Direct-Search-Methods, Model-Based-Methods etc.

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

$\Rightarrow$ Can be extended and used in a biobjective optimization problem!

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

$\Rightarrow$ Can be extended and used in a biobjective optimization problem!

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

$\Rightarrow$ Can be extended and used in a biobjective optimization problem!

MADS

Mesh Adaptive Direct Search is a Directional Direct Search method, using two different meshes in order to achieve a stronger optimality criterium than e.g. pattern search methods.

Main Idea:

1. Generate two different meshes (only conceptually)
2. Let mesh sizes shrink at different rates
3. Evaluate black-box functions only at intersections of the two meshes

$\Rightarrow$ Optimality defined by via the Clarke generalized gradient

$\Rightarrow$ Can be extended and used in a biobjective optimization problem!

Clarke's Calculus

Definition

Let X be Banach, $\mathbf{v} \in X$ some direction and $f : X \to \mathbb{R}$ a real valued function that is locally Lipschitz continuous. The generalized directional derivative or Clarke directional derivative of f at $\mathbf{x}$ in direction $\mathbf{v}$ is given by

$$f^{\circ}(\mathbf{x}; \mathbf{v}) = \lim_{y \to x; \lambda \downarrow 0} \sup \frac{f(\mathbf{y} + \lambda \mathbf{v}) - f(\mathbf{y})}{\lambda} \tag{1}$$

Definition

The Clarke generalized gradient of f at $\mathbf{x}$ is

$$\partial f(\mathbf{x}) = \{\xi \in X : f^{\circ}(\mathbf{x}; \mathbf{v}) > \langle \xi, \mathbf{v} \rangle\} \quad \forall \mathbf{v} \in X \tag{2}$$

Clarke's Calculus

Definition

A point $\mathbf{x}$ is called a Clarke stationary point, if the following holds

$$f^{\circ}(\mathbf{x};\mathbf{v}) \geq 0 \quad \forall \mathbf{v} \in \mathbb{R}^n \Longleftrightarrow 0 \in \partial f(\mathbf{x}) \tag{3}$$

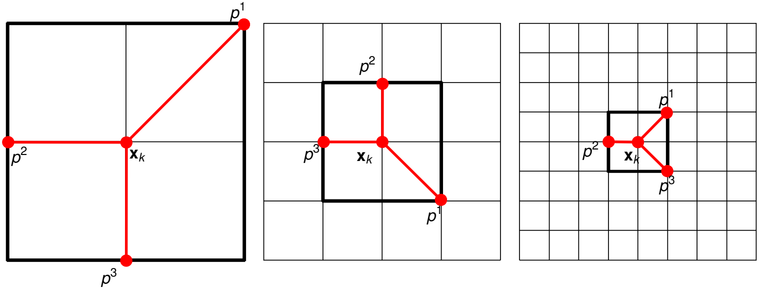
$\Rightarrow$ MADS produces Clarke stationary points!

$\Rightarrow$ Pattern serach does not in general!

$\Rightarrow$ Search directions need to be asymptotically dense!

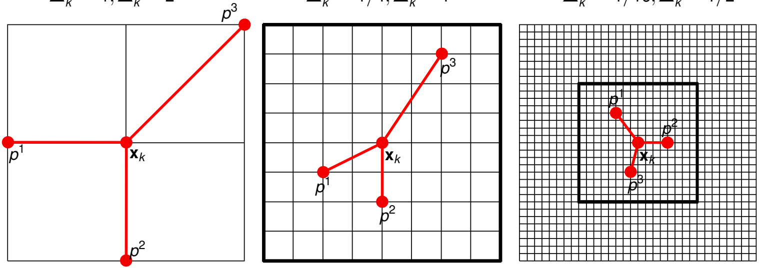
GPS Frames

$\Delta_k^m = \Delta_k^p = 1$ $\qquad$ $\Delta_k^m = \Delta_k^p = 1/2$ $\qquad$ $\Delta_k^m = \Delta_k^p = 1/4$



MADS Frames

$\Delta_k^m = 1, \Delta_k^p = 2$ $\Delta_k^m = 1/4, \Delta_k^p = 1$ $\Delta_k^m = 1/16, \Delta_k^p = 1/2$



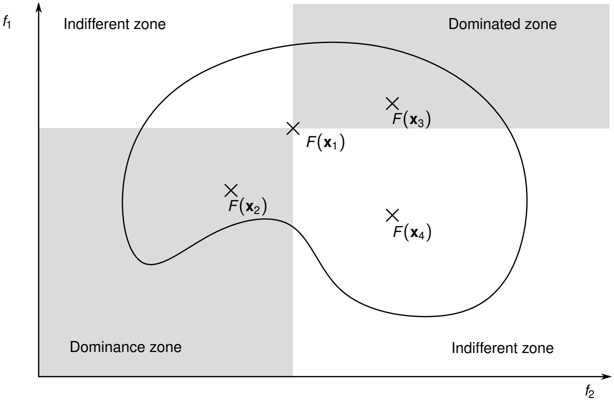
Pareto Optimality

Definition

Let $\mathbf{u}, \mathbf{v} \in X$ be two points of the multiobjective function $F : X \to Y$.

- $\mathbf{u} \preceq \mathbf{v}$ (**u** weakly dominates **v**) $\iff f_i(\mathbf{u}) \leq f_i(\mathbf{v}) \; \forall i \in \{1, \dots, p\}$
- $\mathbf{u} \prec \mathbf{v}$ (**u** dominates **v**)
  $\iff \mathbf{u} \preceq \mathbf{v}$ and $f_j(\mathbf{u}) < f_j(\mathbf{v})$ for at least one $j \in \{1, \dots, p\}$
- $\mathbf{u} \sim \mathbf{v}$ (**u** is indifferent to **v**)
  $\iff$ **u** does not dominate **v** and **v** does not dominate **u**

Pareto Optimality



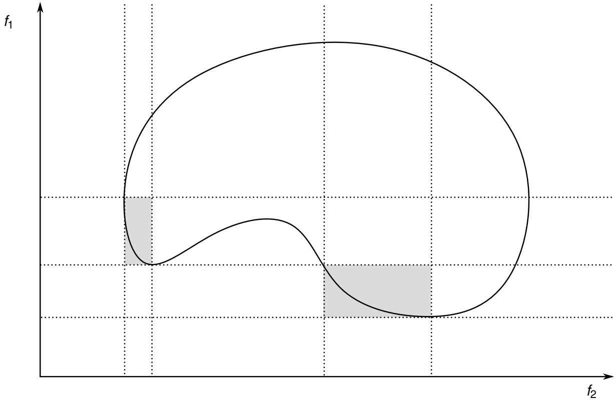
Pareto Optimality

Definition

Let $\mathbf{x} \in X$ be a point of the multiobjective function $F : X \to Y$.

- $\mathbf{x}$ is globally Pareto optimal (just called Pareto optimal) $\iff$ There exists no $\mathbf{y}$ s.t. $\mathbf{y} \prec \mathbf{x}$. If $\mathbf{x}$ is Pareto optimal then $F(\mathbf{x})$ is called Pareto efficient.
- $\mathbf{x}$ is locally Pareto optimal $\iff$ There exists an $\varepsilon, \sigma > 0$ for which the set $\{\mathbf{y} \in B_\varepsilon(\mathbf{x}) \cap X \,|\, \mathbf{y} \prec \mathbf{x},\ F(\mathbf{y}) \in B_\sigma(F(\mathbf{x}))\}$ is empty. If $\mathbf{x}$ is locally Pareto optimal then $F(\mathbf{x})$ is called locally Pareto efficient.

Pareto Optimality



BiMADS

Produces an approximation of the Pareto front!

Main Idea:

1. Initially run MADS and save all Pareto optimal points
2. Generate a single objective formulation by a reference point approach
3. Run MADS and repeat step 2
4. Pareto optimal points along the way are returned as Pareto front

$\Rightarrow$ Implementation used from the NOMAD library

BiMADS

Produces an approximation of the Pareto front!

Main Idea:

1. Initially run MADS and save all Pareto optimal points
2. Generate a single objective formulation by a reference point approach
3. Run MADS and repeat step 2
4. Pareto optimal points along the way are returned as Pareto front

$\Rightarrow$ Implementation used from the NOMAD library

BiMADS

Produces an approximation of the Pareto front!

Main Idea:

1. Initially run MADS and save all Pareto optimal points
2. Generate a single objective formulation by a reference point approach
3. Run MADS and repeat step 2
4. Pareto optimal points along the way are returned as Pareto front

$\Rightarrow$ Implementation used from the NOMAD library

BiMADS

Produces an approximation of the Pareto front!

Main Idea:

1. Initially run MADS and save all Pareto optimal points
2. Generate a single objective formulation by a reference point approach
3. Run MADS and repeat step 2
4. Pareto optimal points along the way are returned as Pareto front

$\Rightarrow$ Implementation used from the NOMAD library

BiMADS

Produces an approximation of the Pareto front!

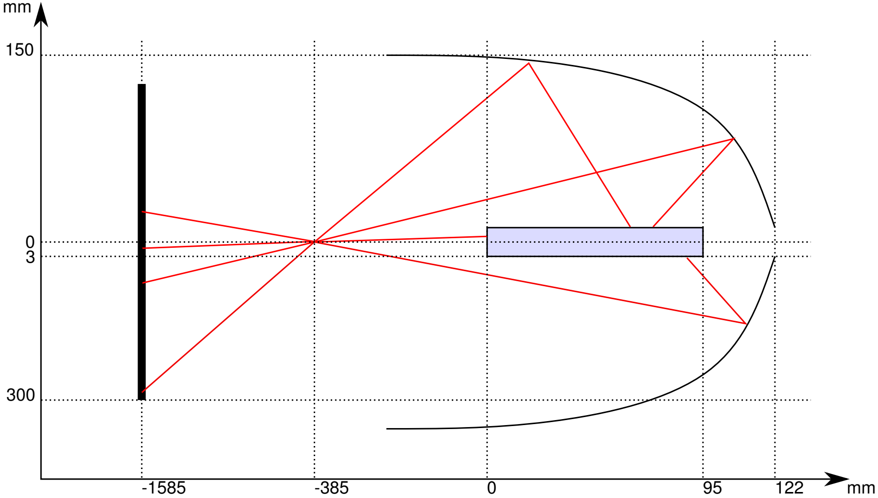
Main Idea:

1. Initially run MADS and save all Pareto optimal points
2. Generate a single objective formulation by a reference point approach
3. Run MADS and repeat step 2
4. Pareto optimal points along the way are returned as Pareto front

$\Rightarrow$ Implementation used from the NOMAD library

Results

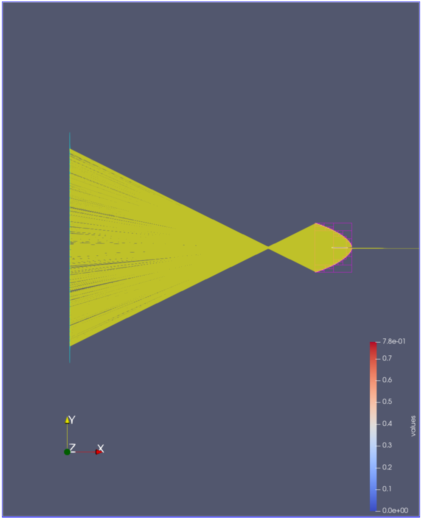
Setup



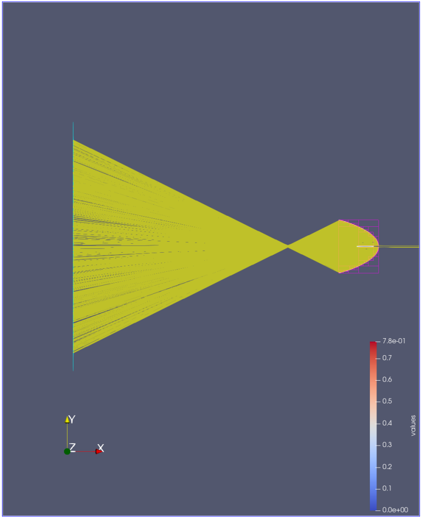
Results

Setup	Pump power	Absorbed Power[W]	Variance[W]	cw-Output[W]	opt.-to-opt. Eff. [%]	Beam quality x	Beam quality y
Reference (algebraic)	720	-	-	30.52	4.24	4.82	4.30
Fixed mirror and crystal distance	720	155.46	13.49	35.14	4.88	4.25	3.22
Open mirror and crytsal distance (initial random search)	690	186.76	2.50	38.22	5.54	4.03	1.96
Open mirror and crytsal distance (pipe configuration)	665	191.13	4.08	38.05	5.72	3.90	1.66

Fixed setup

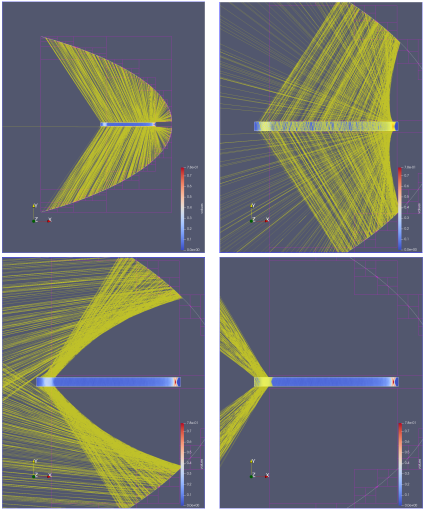


7.8e-01
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0e+00
values
Y
Z
X

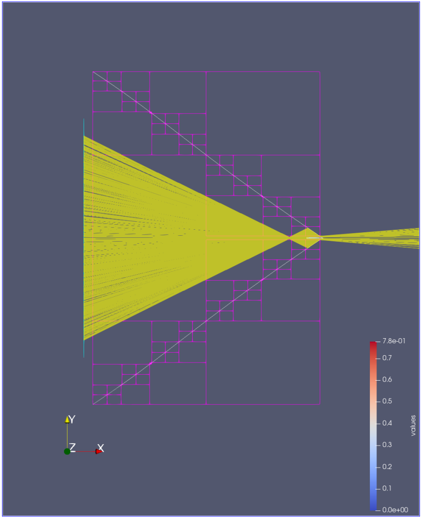


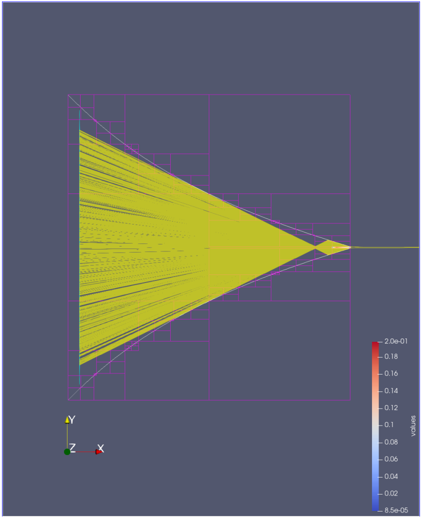
7.8e-01
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0e+00
values
Y
Z
X

Fixed Setup

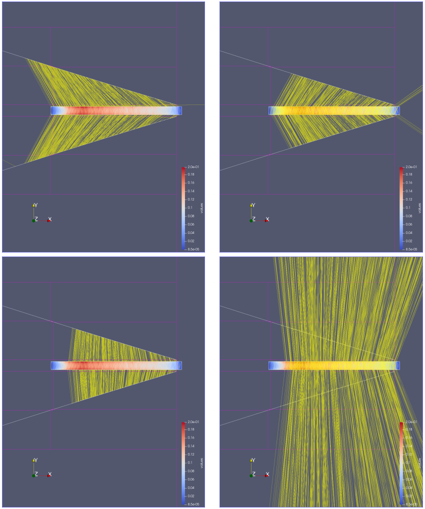


Open setup

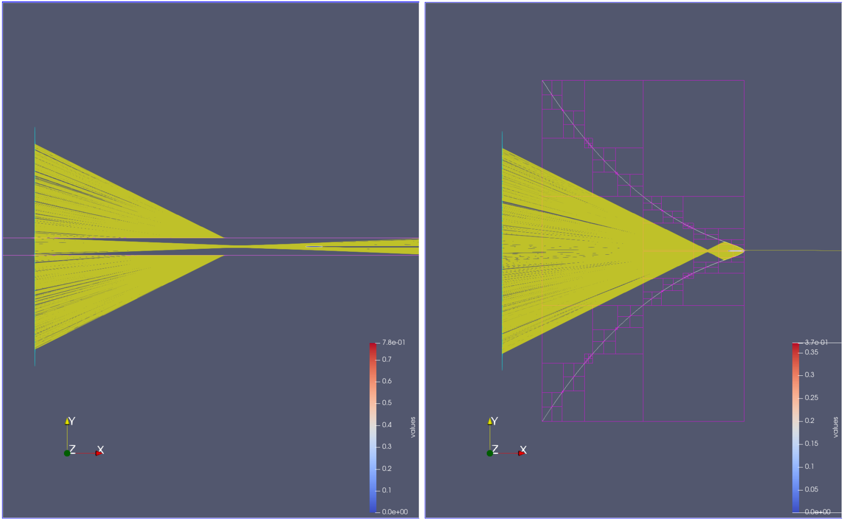

Y
Z
X
7.8e-01
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0e+00
values


Y
Z
X
2.0e-01
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
8.5e-05
values

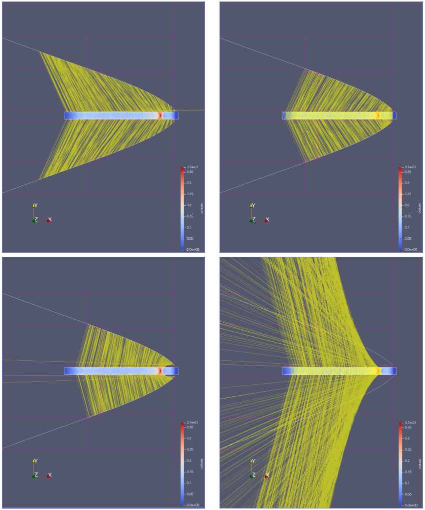
Open setup



Open setup - pipe configuration



Open setup - pipe configuration



FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

Thanks for listening.
Any questions/suggestions?