OPOSSUM: An optics simulation framework for large-size laser systems

The goal of this project is the development of a common software platform for simulating various aspects of optical systems in a holistic approach. It should be particularly useful for simulating and designing large-size, high-energy / intensity laser systems. OPOSSUM stands for **Op**en **S**ource **O**ptics **S**imulation **S**ystem and **U**nified **M**odeler. This work is part of the task 3.4 (supporting calculations for system design) within the THRILL project and is led by GSI.

# The situation so far…

The design of complex high-energy / intensity laser systems requires a detailed simulation of optical (and sometimes mechanical) effects and aspects. Often, these aspects have to be simultaneously taken into account while optimizing a system design. Different aspects might even stand against each other, such that optimizing (e.g. maximizing/ minimizing) one effect degrades the performance of other system parameters. Therefore, a rather holistic approach would be desirable.

In the past, many tools were developed, often addressing very particular optical effects at several companies and research institutes. Often these tools are only used at the institutions which developed the software and even there only used by one or two people (e.g. in the frame of a master or PhD thesis). This of course sometimes leads to the situation that different institutes repeat the work and "reinvent the wheel". Hence, a common set of tools accompanied by proper knowledge exchange would significantly reduce this inefficiency.

Besides the solutions for modelling particular aspects of optical systems, there are many more general-purpose tools on the market, which are unfortunately commercial, closed-source solutions. Each software has its own underlying design strategy. Furthermore, many of these tools (e.g. ZEMAX, OSLO, etc.) are designed for simulating more "traditional" optical systems such as camera objectives or illumination setups. In contrast, laser (chain) systems often demand different features which are not always fully supported (or easy to model) by these software packages.

The usage of different tools during the design phase often requires repeatedly modelling the optical system in the particular software and providing a bunch of input parameters. A common platform would allow for modelling the desired system once and analyzing it with the above-mentioned tools and providing the input data in the particular format.

The first task of this work is the development of a concept, which allows for modeling the most common (if not all) optical systems. This effort leads to the development of the OPOSSUM framework. The entire software and its documentation is accessible at

<https://git.gsi.de/phelix/rust/opossum>

# The idea…

In general, optical systems consist of light sources which provide a more or less complex light field (time invariant or time dependent) and optical components, which modify this light field. Furthermore, there are light sinks such as simple beam dumps, targets or detectors. These are the elements, which produce a "result" (e.g. measurable signal) and thus make a system "productive". The components - light sources (such as a laser) or optical elements (e.g. Faraday isolators) - might itself consist of sub components. In principle, these components again might consist of sub components with an unlimited nesting level.

Of course, for a full system description, it would be sufficient simply placing the mechanical model of the optical components in a 3D space. For certain tasks, such as illumination or stray light analysis this would be an appropriate approach (and thus will be supported by our model). However, typical systems mostly cast optical rays or light fields in a directed way from one component to the next one. Optical systems can thus be rather described in network- or most often in tree-like structures.

Following the above idea, well-established structures could be used which already exist for a long time: [directed graphs](https://en.wikipedia.org/wiki/Directed\_graph). A directed graph consists of so-called nodes and edges. For our purposes, nodes represent the optical components, while edges represent the information about the light (energy, wavelength, wave front, nearfield distribution, etc.) to be handed from one node to the next one.

A node has one or more \*ports\* where edges can be connected to. We thereby strictly distinguish between incoming and outgoing ports. A node with no input ports represents a light source. A simple (ideal) propagation node would have one input and one output port. Furthermore, an ideal beam splitter has one input port and two or more output ports. More realistic components, such as a real lenses could also have more than one input and output ports e.g. for simulating ghost reflections from lens surfaces.

There will be different node types representing various optical components (ideal / real lenses, beam splitters, wave plates, etc.). Each node has, depending on its node type, various attributes, which describe component parameters such as length (e.g. for propagation nodes), focal length (ideal lenses), radii of curvature (real lenses) etc.

An optical model can now be investigated using different analyzer modules. Each module can concentrate on a specific aspect such as energy transmission, geometric raytracing, Fourier optics propagation or even finding all possible ghost foci in a system. The important point is the use of an model, which has to be developed only once without any data conversion etc…

# A common platform…

Using the above approach, nodes can include arbitrary analysis algorithms and simulate even complex effects. They might also use already existing simulation codes. As an example, a code for the simulation of second harmonic generation (SHG) using non-linear crystals has been developed as a separate project in our group. The OPOSSUM framework will allow for creating an SHG node incorporating this code. This way, arbitrary optical setups can make use of such a node.

# The OPOSSUM ecosystem…

materialdb, opticdb