A new approach for the verification of optical systems

Umair Siddique, Vincent Aravantinos and Sofiène Tahar

Department of Electrical and Computer Engineering,
Concordia University, Montreal, Canada

ABSTRACT

Optical systems are increasingly used in microsystems, telecommunication, aerospace and laser industry. Due to the complexity and sensitivity of optical systems, their verification poses many challenges to engineers. Traditionally, the analysis of such systems has been carried out by paper-and-pencil based proofs and numerical computations. However, these techniques cannot provide perfectly accurate results due to the risk of human error and inherent approximations of numerical algorithms. In order to overcome these limitations, we propose to use theorem proving (i.e., a computer-based technique that allows to express mathematical expressions and reason about them by taking into account all the details of mathematical reasoning) as an alternative to computational and numerical approaches to improve optical system analysis in a comprehensive framework. In particular, this paper provides a higher-order logic (a language used to express mathematical theories) formalization of ray optics in the HOL Light theorem prover. Based on the multivariate analysis library of HOL Light, we formalize the notion of light ray and optical system (by defining medium interfaces, mirrors, lenses, etc.), i.e., we express these notions mathematically in the software. This allows us to derive general theorems about the behavior of light in such optical systems. In order to demonstrate the practical effectiveness, we present the stability analysis of a Fabry-Pérot resonator.

Keywords: Ray Optics, Optical Resonators, Formal Methods, Theorem Proving, Higher-Order Logic

1. INTRODUCTION

Optical system development involves the physical modeling of optical components, analysis, and production. This process is always subject to time, safety and cost constraints. Due to their sophistication and physical nature, optical systems are considered to be more complicated than many other types of engineering systems. One of the most critical requirement is the validation of the models and verification of the system properties. Therefore, significant portion of time is spent in the analysis and verification to find the bugs in design process prior to the manufacturing of the actual system. The minor bugs in optical systems can lead to disastrous consequences such as the loss of human lives because of their use in surgeries and high precision biomedical devices, or financial loss because of their use in high budget space missions. For example, Hubble Telescope,[1] which is considered as one of NASA's largest projects with a budget of $1.6 billion, faced a historical system failure due to the misalignment of two mirrors of the telescope. Considering these facts, it is very important to build a framework for the analysis of optical systems which is both accurate and scalable. In general, the analysis of optical systems has been carried out by three techniques: paper-and pencil based proofs, computer simulation and computer algebra systems (CAS).

Paper-and-Pencil Proof: Traditionally, the analysis of optical system models has been done using paper-and-pencil proofs.[2–4] A mathematical model of the optical system is built using the underlying physical concepts. This model is then used to verify that the system exhibits the desired properties using mathematical reasoning on paper. However, considering the complexity of present age optical and laser systems, such an analysis is very difficult if not impossible, and thus quite error-prone. Many examples of erroneous paper-and-pencil proofs are available in the open literature, a recent one can be found in paper of Cheng[5] and its identification and correction is reported by Naqvi.[6]

Computer Simulation: The availability of high-speed computers attracted the attention of researchers working in the field of optics to perform simulation based analysis using numerical algorithms. The main idea of

Corresponding Author: Umair Siddique: E-mail:muh_sidd@ece.concordia.ca

Optical System Alignment, Tolerancing, and Verification VII, edited by José Sasián,
Richard N. Youngworth, Proc. of SPIE Vol. 8844, 88440G · © 2013 SPIE
CCC code: 0277-786X/13/$18 · doi: 10.1117/12.2024860

simulation-based methods is to construct a discretized model and then simulate the output of the system using different input patterns. One of the most commonly used computer-based analysis techniques for optical systems is the numerical computation of complex ray-transfer matrices.[7–9] The analysis of optical and laser systems involves complex and vector analysis along with transcendental functions, thus numerical computations cannot provide perfectly accurate results due to the heuristics and approximations of the underlying numerical algorithms.

Computer Algebra Systems: Computer algebra systems (CAS)[10] are becoming popular for the analysis of optical systems. In such systems, mathematical computations are done using symbolic algorithms, hence they are better than simulation-based analysis in terms of precision. But the simplification performed by computer algebra systems are not 100 % reliable[11] due to their inability to deal with side conditions. Another source of inaccuracy in computer algebra systems is the presence of unverified huge symbolic manipulation algorithms in their core, which are quite likely to contain bugs.

Formal methods[12] allow accurate and precise analysis and thus overcome the above mentioned limitations of traditional approaches. The main idea behind them is to develop a mathematical model for the given system and analyze this model using computer-based mathematical reasoning, which in turn increases the chances for catching subtle but critical design errors that are often ignored by traditional techniques. There are essentially two main formal methods techniques: model checking[13] and higher-order-logic theorem proving.[14] Model checking is an automated verification technique for systems that can be expressed as finite-state machines. On the other hand, higher-order logic theorem proving is an interactive verification technique, but it is more flexible and can handle variety of systems.

The above mentioned inaccuracy problems of traditional analysis techniques are impeding their usage in designing safety-critical engineering systems, where a small analysis error can result in heavy losses in terms of human life or finance (e.g., corneal surgery[15]). In order to raise the level of accuracy of analyzing optical systems, we propose in this paper a higher-order logic theorem proving based framework for ray optics analysis. Developing a higher-order logic model for a physical system and formally analyzing this model is a very challenging task since it requires expertise in both mathematics and physics. However, it provides an effective way for identifying critical design errors that are often ignored by traditional analysis techniques like simulation and computer algebra systems. We believe that higher-order logic theorem proving[14] offers a promising solution for conducting the formal analysis of such critical optical and laser systems.

The comparison of all the traditional techniques and theorem proving is given in Table 1. These are evaluated according to their expressiveness, scalability, accuracy, availability of optical system fundamentals (Ray Optics) and the possibility of automation of the analysis. It is important to note that the accuracy of paper-and-pencil proofs is uncertain for the larger systems (it is indicated by a question-mark (?) in Table 1). Theorem proving, which is the most accurate among all techniques, lacks the formalization of the underlying theories of ray optics for handling real-world optical systems but this downside can be overcome, hence the "× - (✓)" mark in Table 1.

Criteria	Paper-and-Pencil Proof	Simulation	CAS	Theorem Proving
Expressiveness	✓	✓	✓	✓
Scalability	×	×	✓	✓
Accuracy	**✓-?**	×	×	✓
Ray Optics	✓	✓	✓	× - (✓)
Automation	×	✓	✓	×

Table 1. Comparison of analysis techniques for optical systems

The rest of the paper is organized as follows: Section 2 presents the concepts of higher-order logic theorem proving. Section 3 presents our HOL Light formalization of ray optics. We present a stability analysis procedure in Section 4. In order to demonstrate the practical effectiveness and the use of our work, we present in Section 5 the formal analysis of widely used Fabry-Pérot resonator. Finally, Section 6 concludes the paper and highlights some future directions.

2. PRELIMINARIES

In this section, we provide a brief introduction to the higher-order logic theorem proving and to the HOL Light theorem prover.

2.1 Theorem Proving

Theorem proving is concerned with the construction of mathematical theorems by a computer program using axioms and hypothesis. Theorem proving systems are widely used in software and hardware verification. For example hardware designer can prove the different properties of a digital circuit by using some predicates to model the circuits model. Similarly, a mathematician can prove the transitivity of real numbers using the axioms of real number theory. The language of these mathematical theorems or conjectures is logic i.e., propositional logic, first-order logic or higher-order logic, depending upon the expressibility requirement, for example, the use of higher-order logic is advantageous over first-order logic in terms of the availability of additional quantifiers and its high expressiveness.

A theorem prover is a software in which mathematical theories can be expressed with as much accuracy as pencil-and-paper, but with the precise control of the computer which ensures that no mathematical mistake is involved. Concretely, mathematical expressions (and not just equations) can be input in the system which is able to understand its precise semantics, and thus ascertain that no faulty reasoning is applied. As an example, a theorem prover will not allow to conclude that "$\frac{x}{x} = 1$" unless it is first proved that $x \neq 0$. No computer algebra system will take care of such a subtlety when simplifying some equation. This is achieved by defining a precise syntax of the mathematical sentences that can be input in the software. The designer of the theorem prover also provides some axioms and inference rules which are the only ways to prove a sentence correct. This purely deductive aspect provides the guarantee that every sentence proved in the system is true (in particular, there is no approximation like in computer algebra systems). When a mathematical or physical theory is expressed inside a theorem prover, we say that it is formalized.

There are two types of provers: automatic and interactive. In an interactive theorem prover, significant user-computer interaction is required while automatic theorem provers can perform different proof tasks automatically. The main downside of automatic theorem provers is their less expressiveness, which limits their usage in the domains where complicated mathematics is involved (e.g., multivariate calculus). Some prominent interactive higher-order logic based theorem provers are Isabelle, Coq, HOL, HOL Light, ProofPower, ACL2 and MIZAR.[16]

2.2 The HOL Light Theorem Prover

HOL Light[17] is an interactive theorem proving environment for the construction of mathematical proofs in higher-order logic. A theorem is a formalized statement that may be an axiom or could be deduced from already verified theorems by an inference rule. A theorem consists of a finite set Ω of Boolean terms called the assumptions and a Boolean term S called the conclusion. For example, "$\forall x.x \neq 0 \Rightarrow \frac{x}{x} = 1$" represents a theorem in HOL Light. A HOL Light theory consists of a set of types, constants, definitions, axioms and theorems. HOL theories are organized in a hierarchical fashion and theories can inherit the types, constants, definitions and theorems of other theories as their parents. In the development of the framework, presented in this paper, we make use of the HOL Light theories of Boolean variables, real numbers, transcendental functions and multivariate analysis. In fact, one of the primary motivations of selecting the HOL Light theorem prover for our work was to benefit from these built-in mathematical theories. The proofs in HOL Light are based on the concept of a tactic that breaks goals into simple subgoals. There are many automatic proof procedures and proof assistants[18] available in HOL Light which help the user in directing the proof to the end.

Table 1 provides the mathematical interpretations of some frequently used HOL Light symbols and functions in this paper.

3. FORMALIZATION OF RAY OPTICS THEORY

In this section, we present an overview of our higher-order logic formalization of ray optics. The formalization consists of three parts: 1) the formalization of optical system structure; 2) the modeling of ray behavior; 3) the optical resonators and their stability.

Table 2. HOL Light Symbols and Functions

HOL Symbol	Standard Symbol	Meaning
/\	and	Logical *and*
\/	or	Logical *or*
~	not	Logical *negation*
`T`	true	Logical true value
`F`	false	Logical false value
==>	$\longrightarrow$	Implication
<==>	$=$	Equality
`!x.t`	$\forall x.t$	for all $x : t$
`?x.t`	$\exists x.t$	for some $x : t$
`@x.t`	$\epsilon x.t$	an x such that : t
`λx.t`	$\lambda x.t$	Function that maps x to $t(x)$
`num`	$\{0, 1, 2, \ldots\}$	Positive Integers data type
`real`	All Real numbers	Real data type
`complex`	All complex numbers	Complex data type
`suc n`	$(n + 1)$	Successor of natural number
`abs x`	$\|x\|$	Absolute function
`&a`	$\mathbb{N} \to \mathbb{R}$	Typecasting from Integers to Reals
`A**B`	$[A][B]$	Matrix-Matrix or Matrix-Vector multiplication

3.1 Modeling of Optical System structure

We present the formalization of optical systems in the perspective of sequential ray tracing, where the order and behavior of each optical component is predefined.[19]

The theory of ray optics explains the behavior of light when it passes through the free space and interacts with different interfaces like spherical and plane. We can model free space by a pair of real numbers (n, d), which are essentially the refractive index and the total width, as shown in Figure 1 (a). For the sake of simplicity, we consider only two fundamental interfaces, i.e., plane and spherical which are further categorized as either transmitted or reflected (as shown in Figure 1 (b-e)). Furthermore, spherical interface can be described by its radius of curvature (R). We encode the above description in the language (i.e., higher-order logic) of the HOL Light by defining some new types as follows:

Definition 1 (Optical Interface and Free Space).

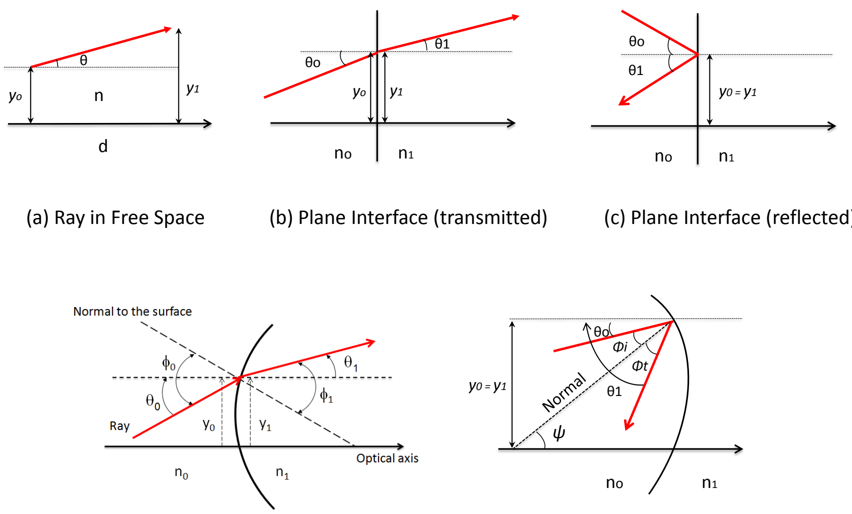
```
new_type_abbrev ("free_space",`:ℝ × ℝ`)
define_type "optical_interface = plane | spherical ℝ"
define_type "interface_kind = transmitted | reflected"
```

An optical component is made of a free space (`free_space`) and an optical interface (`optical_interface`) as defined above. Finally, an optical system is a list of optical components followed by a free space. When passing through an interface, the ray is either transmitted or reflected (it is because of the fact that we are only considering sequential ray tracing). In our formalization, this information is also provided in the type of optical components, as shown by the use of the type `interface_kind` as follows:

Definition 2 (Optical Component and System).

```
new_type_abbrev ("optical_component", `:free_space × optical_interface × interface_kind`)
new_type_abbrev ("optical_system",`:optical_component list × free_space`)
```

Note that this datatype can easily be extended to many other optical components if needed.



(a) Ray in Free Space (b) Plane Interface (transmitted) (c) Plane Interface (reflected)

(d) Spherical Interface (transmitted) (e) Spherical Interface (reflected)

Figure 1. Behavior of ray at different interfaces

The next step in our formalization is to define some predicates to ensure the validity of free space, optical components and systems. A value of type `free_space` does represent a real space only if the refractive index is greater than zero. In addition, in order to have a fixed order in the representation of an optical system, we impose that the distance of an optical interface relative to the previous interface is greater or equal to zero. We also need to assert the validity of a value of type `optical_interface` by ensuring that the radius of curvature of spherical interfaces is never equal to zero. This yields the following predicates:

Definition 3 (Valid Free Space and Valid Optical Interface).

```
⊢ is_valid_free_space ((n,d):free_space) ⇔ 0 < n ∧ 0 ≤ d
⊢ (is_valid_interface plane ⇔ T) ∧ (is_valid_interface (spherical R) ⇔ 0 <> R)
```

Then, by ensuring that this predicate holds for every component of an optical system, we can characterize valid optical systems as follows:

Definition 4 (Valid Optical Component).

```
⊢ ∀fs i ik.  is_valid_optical_component ((fs,i,ik):optical_component)
   ⇔ is_valid_free_space fs ∧ is_valid_interface i
```

Definition 5 (Valid Optical System).

```
⊢ ∀cs fs.  is_valid_optical_system ((cs,fs):optical_system) ⇔
  ALL is_valid_optical_component cs ∧ is_valid_free_space fs
```

where `ALL` is a HOL Light library function which checks that a predicate holds for all the elements of a list.

3.2 Modeling of Ray Behavior

We can now formalize the physical behavior of a ray when it passes through an optical system. We only model the points where it hits an optical interface (instead of modeling all the points constituting the ray). So it is sufficient to just provide the distance of every of these hitting points to the axis and the angle taken by the ray at these points. Consequently, we should have a list of such pairs $(distance, angle)$ for every component of a system. In addition, the same information should be provided for the source of the ray. For the sake of simplicity, we define a type for a pair $(distance, angle)$ as `ray_at_point`. This yields the following definition:

Definition 6 (Ray).

```
new_type_abbrev ("ray_at_point", `:ℝ × ℝ`)
new_type_abbrev ("ray", `:ray_at_point × ray_at_point × (ray_at_point × ray_at_point) list`)
```

The first `ray_at_point` is the pair (*distance*, *angle*) for the source of the ray, the second one is the one after the first free space, and the list of `ray_at_point` pairs represents the same information for the interfaces and free spaces at every hitting point of an optical system.

Once again, we specify what is a valid ray by using some predicates. First of all, we define what is the behavior of a ray when it is traveling through a free space. In paraxial limit, ray travels in a straight line in free space and thus its distance from the optical axis and angle can be related as $y_1 = y_0 + d * \theta_0$ and $\theta_1 = \theta_0$ (as shown in Figure 1), respectively.[20] In order to model this behavior, we require the position and orientation of the ray at the previous and current point of observation, and the free space itself.

Definition 7 (Behavior of a Ray in Free Space).

```
⊢ is_valid_ray_in_free_space (y₀,θ₀) (y₁,θ₁) ((n,d):free_space) ⇔
    y₁ = y₀ + d * θ₀ ∧ θ₀ = θ₁
```

where (y_0,θ_0), (y_1,θ_1) and (`(n,d):free_space`) represent the ray orientation at previous and current point, and free space, respectively.

Next, we define what is the valid behavior of a ray when hitting a particular interface. This requires the position and orientation of the ray at the previous and current interface, and the refractive index before and after the component. Then the predicate is defined by case analysis on the interface and its type as follows:

Definition 8 (Behavior of a Ray at Given Interface).

```
⊢ (is_valid_ray_at_interface (y₀,θ₀) (y₁,θ₁) n₀ n₁ plane transmitted
   ⇔ y₁ = y₀ ∧ n₀ * θ₀ = n₁ * θ₁) ∧
  (is_valid_ray_at_interface (y₀,θ₀) (y1,θ₁) n₀ n₁ (spherical R)
   transmitted ⇔ let φᵢ= θ₀ + y₁/R and φₜ = θ₁ + y₁/R in
   y₁ = y₀ ∧ n₀ * φᵢ = n₁ * φₜ) ∧
  (is_valid_ray_at_interface (y₀,θ₀) (y₁,θ₁) n₀ n₁ plane reflected
   ⇔ y₁ = y₀ ∧ n₀ * θ₀ = n₀ * θ₁) ∧
  (is_valid_ray_at_interface (y₀,θ₀) (y₁,θ₁) n₀ n₁ (spherical R)
   reflected ⇔ let φᵢ = y1/R - θ₀ in y₁ = y₀ ∧ θ₁ = -(θ₀ + 2 * φᵢ))
```

The above definition states some basic geometrical facts about the distance to the axis, and applies paraxial Snell's law[20] to the orientation of the ray as shown in Figure 1. Similarly, we can define the behavior of ray in the entire system by a predicate `is_valid_ray_in_system` (the definition of this predicate is straightforward, details can be seen here[21]).

3.3 Verification of Ray-Transfer Matrices

The main strength of the ray optics is its matrix formalism,[19] which provides an efficient way to model all the optical components in the form of a matrix. Indeed, matrix relates the input and the output ray by a linear relation. For example, in case of free space, the input and out put ray parameters are related by two linear equations, i.e., $y_1 = y_0 + d * \theta_0$ and $\theta_1 = \theta_0$, which further can be described as a matrix (also called ray-transfer matrix of free space) as follows:

$$\begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$$

Next, we verify the ray-transfer-matrix of free space as follows:

Theorem 1 (Ray-Transfer-Matrix for Free Space).

$\vdash \forall \texttt{n d } y_0\ \theta_0\ y_1\ \theta_1.\ \ \texttt{is\_valid\_free\_space (n,d)} \wedge$

$\texttt{is\_valid\_ray\_in\_free\_space } (y_0,\theta_0)\ (y_1,\theta_1)\ \texttt{(n,d))} \Longrightarrow \begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix} ** \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$

The first assumption ensures the validity of free space and the second assumption ensures the valid behavior of ray in free space. We use the traditional mathematical notation of matrices for the sake of clarity, whereas we define these matrices using the HOL Light Vectors library. For example, a simple `2 x 2` matrix can be defined as follows:

$\vdash \forall$ `A B C D. matrix_2x2 A B C D = vector[vector[A;B];vector[C;D]]`

Proof Sketch: We start the proof by rewriting the above statement of Theorem 1 with the definitions of `is_valid_free_space` (Definition 3) and `is_valid_ray_in_free_space` (Definition 7), which results in the following subgoal:

$$\forall \texttt{n d } y_0\ \theta_0\ y_1\ \theta_1.0 < n \wedge 0 \leq d \wedge y_1 = y_0 + d*\theta_0 \wedge \theta_0 = \theta_1 \Rightarrow \begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix} ** \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$$

This can easily be proved by simplifying with the matrix-vector multiplication and equating the two vectors.

Similarly, we proved the ray-transfer matrices of plane and spherical interfaces for the case of transmission and reflection, as listed in Table 2. The proof steps for these theorems are similar to the one for Theorem 1. In order to make the proof of these theorems automatic, we build a tactic `common_prove` which takes a theorem as its argument and automatically verifies it. For example, Theorem 1 can be automatically proved by the following HOL Light code:

```
let FS_MATRIX = common_prove
  `!n d y0 theta0 y1 theta1.
    is_valid_free_space ((n,d):free_space)
    /\ is_valid_ray_in_free_space (y0,theta0) (y1,theta1) (n,d)
    ==> vector[y1;theta1] = (matrix_2x2 &1 d &0 &1) ** vector[y0;theta0]`;;
```

Component	**HOL Light Formalization**
Plane Interface (Reflection)	$\vdash \forall \texttt{n d } y_0\ \theta_0\ y_1\ \theta_1.\ \ 0 < n_0 \wedge 0 < n_1 \wedge$ $\texttt{is\_valid\_ray\_at\_interface } (y_0,\theta_0)\ (y_1,\theta_1)\ n_0\ n_1 \texttt{ plane reflected} \Longrightarrow$ $\begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} ** \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$
Plane Interface (Transmission)	$\vdash \forall \texttt{n d } y_0\ \theta_0\ y_1\ \theta_1.\ \ 0 < n_0 \wedge 0 < n_1 \wedge$ $\texttt{is\_valid\_ray\_at\_interface } (y_0,\theta_0)\ (y_1,\theta_1)\ n_0\ n_1 \texttt{ plane transmitted} \Longrightarrow$ $\begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{n_0}{n_1} \end{bmatrix} ** \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$
Spherical Interface (Reflection)	$\vdash \forall \texttt{n d } y_0\ \theta_0\ y_1\ \theta_1 \texttt{ R. } 0 < n_0 \wedge 0 < n_1 \wedge \texttt{(is\_valid\_interface (spherical R)} \wedge$ $\texttt{is\_valid\_ray\_at\_interface } (y_0,\theta_0)\ (y_1,\theta_1)\ n_0\ n_1 \texttt{ (spherical R) reflected} \Longrightarrow$ $\begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{2}{R} & 1 \end{bmatrix} ** \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$
Spherical Interface (Transmission)	$\vdash \forall \texttt{n d } y_0\ \theta_0\ y_1\ \theta_1 \texttt{ R. } 0 < n_0 \wedge 0 < n_1 \wedge \texttt{(is\_valid\_interface (spherical R)} \wedge$ $\texttt{is\_valid\_ray\_at\_interface } (y_0,\theta_0)\ (y_1,\theta_1)\ n_0\ n_1 \texttt{ (spherical R) transmitted} \Longrightarrow$ $\begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{n_0 - n_1}{R*n_1} & \frac{n_0}{n_1} \end{bmatrix} ** \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$

Table 3. Ray-Transfer Matrices of Optical Components

We can trace the input ray R_i through an optical system consisting of k optical components by the composition of ray-transfer matrices of each optical component as follows:

$$R_o = (M_k.M_{k-1}....M_1).R_i \tag{1}$$

Simply, we can write $R_o = M_s R_i$ where $M_s = \prod_{i=k}^{1} M_i$. Here, R_o is the output ray and R_i is the input ray. It is important to note that in this equation, individual matrices of optical components are composed in reverse order. We formalize this fact with the following recursive definition:

Definition 9 (System Composition).

```
⊢ system_composition ([],n,d) ⇔ free_space_matrix d ∧
  system_composition (CONS ((nt,dt),i,ik) cs,n,d) ⇔
  (system_composition (cs,n,d) ** interface_matrix nt (head_index (cs,n,d)) i ik) **
  free_space_matrix dt
```

The general ray-transfer-matrix relation is then given by the following theorem:

Theorem 2 (Ray-Transfer-Matrix for Optical System).

```
⊢ ∀sys ray.  is_valid_optical_system sys ∧
  is_valid_ray_in_system ray sys ⟹
  let (y0,θ0),(y1,θ1),rs = ray in
  let yn,θn = last_ray_at_point ray in
```

$$\begin{bmatrix} y_n \\ \theta_n \end{bmatrix} = \texttt{system\_composition sys **} \begin{bmatrix} y_0 \\ \theta_0 \end{bmatrix}$$

Here, the parameters `sys` and `ray` represent the optical system and the ray, respectively. `last_ray_at_point` returns the last `ray_at_point` of the ray in the system. Both assumptions in the above theorem ensure the validity of the optical system and the good behavior of the ray in the system. The theorem is easily proved by induction on the length of the system and by using previous results and definitions.

This concludes our formalization of optical system structure and ray along with the verification of important properties of optical components and optical systems. We provide the highlights of formal modeling and stability analysis of optical resonators in the next section.

3.4 Optical Resonators and their Stability

The stability analysis of optical resonators identifies geometric constraints of the optical components which ensure that light remains inside the resonator (see Figure 2). Both stable and unstable resonators have diverse applications, e.g., stable resonators are used in the measurement of refractive index of cancer cells,[22] whereas unstable resonators are used in the laser oscillators for high energy applications.[23]

The stability analysis of resonators requires the consideration of infinite rays or an infinite set of finite rays. In order to consider infinite set of rays, we adopt the conventional method, i,e., "unfolding" the resonator as many times as needed, depending on the considered ray.

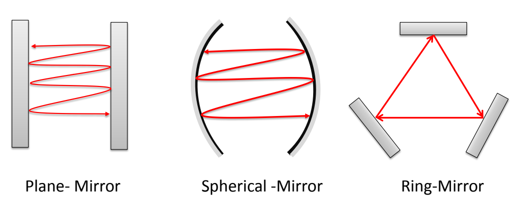


Figure 2. Different types of optical resonators

In our formalization, we want the user to provide only the minimum information so that HOL Light generates automatically the unfolded systems. Therefore, we do not define resonators as just optical systems but define a dedicated type for them: generally, resonators are made of two highly reflecting interfaces and a list of components in between. We thus define the following type:

Definition 10 (Optical Resonator).

```
define_type "resonator = :interface × optical_component list × free_space × interface"
```

Note that the additional free space in the type definition is required because the `optical_component` type only contains one free space (the one before the interface, not the one after).

As discussed earlier, we introduce a predicate to ensure that a value of type `resonator` indeed models a real resonator:

Definition 11 (Valid Optical Resonator).

```
⊢ ∀i₁ cs fs i₂.  is_valid_resonator ((i₁,cs,fs,i₂):resonator)⇔
  is_valid_interface i₁ ∧ ALL is_valid_optical_component cs ∧ is_valid_free_space fs ∧
  is_valid_interface i₁
```

Next, We define a function in order to unfold the resonator as follows:

Definition 12 (Unfold Resonator).

```
⊢ unfold_resonator ((i₁,cs,fs,i₂):resonator) N =
  list_pow (round_trip (i₁,cs,fs,i₂)) N,(head_index (cs,fs),0)
```

where `round_trip` is a function which takes resonator as an input and returns the one complete round trip. The function `list_pow l n` concatenates `n` copies of the list `l`. The function `head_index` provides the refractive index of the next optical component. The argument `N` represents the number of times we want to unfold the resonator. Note that the output type is `optical_system`, therefore all the previous predicates and theorems can be used on an unfolded resonator.

We can now define formally the notion of *stability*. For an optical resonator to be stable, the distance of the ray from the optical axis and its orientation should remain bounded whatever is the value of N. This is formalized as follows:

Definition 13 (Resonator Stability).

```
⊢ ∀res.  is_stable_resonator res ⇔ (∀r.  ∃y θ.  ∀N.
  is_valid_ray_in_system r (unfold_resonator res N) ⟹
  (let yₙ,θₙ = last_single_ray r in abs(yₙ) ≤ y ∧ abs(θₙ) < θ))
```

Proving that a resonator satisfies the abstract condition of Definition 13 does not seem trivial at first. However, if the determinant of a resonator matrix M is 1 (which is the case in practice), physicists have known for a long time that having $-1 < \frac{M_{11}+M_{22}}{2} < 1$ is sufficient to ensure that the stability condition holds. The obvious advantage of this criterion is that it is immediate to check. This can actually be proved by using Sylvester's Theorem[24] which we verified in HOL Light.[25]

Next, we derive now the generalized stability theorem for any resonator as follows:

Theorem 3 (Stability Theorem).

$$\vdash \forall \texttt{res}.\ \texttt{is\_valid\_resonator res} \land \\ (\forall \texttt{N. let M = system\_composition (unfold\_resonator res 1) in} \\ \texttt{det M = 1} \land \texttt{-1} < \frac{\texttt{M}_{1,1}+\texttt{M}_{2,2}}{2} \land \frac{\texttt{M}_{1,1}+\texttt{M}_{2,2}}{2} < \texttt{1}) \Longrightarrow \texttt{is\_stable\_resonator res}$$

where $\texttt{M}_{i,j}$ represents the element at column `i` and row `j` of the matrix. The formal verification of Theorem 3 requires the definition of stability (Definition 13) and Sylvester's theorem.[25]

4. PROCEDURE FOR THE STABILITY ANALYSIS

We present a general procedure for the modeling and analysis of optical resonators in order to demonstrate the flow and utilization of the theorem proving based framework. The overall procedure consists of four steps as shown in Figure 3. The inputs to the proposed framework are the description of optical resonator and the constraints of the underlying parameters, such as radius of curvature and free space width. The first step is to formally model the given resonators description in a theorem prover, as shown in Figure 3. Essentially, it requires the corresponding type definitions of optical resonators, which we have formalized in Section 3.1. In order to be sure that the formal model constructed in the previous step is indeed a real resonator, the next step is to check its validity. This requires the formalizing of the valid optical components and systems, presented in Section 3.1. If the model does not satisfy the validity conditions, the analysis cannot proceed further and initial design requires the revision. The third step in our stability analysis procedure is to formalize the stability theorem using the formal model and the parameter constraints. The ray behavior and resonator formalization (section 3.2 and 3.3) is a prerequisite for this step. The verification of the stability of resonator is the final step which is heavily based on stability theorem, verified in section 3.3 along with the formalization presented in section 3.2 and 3.3. Finally the output is the formal proof of resonator stability if the model satisfies all the constraints, otherwise the initial description and parameters constraints need revision.

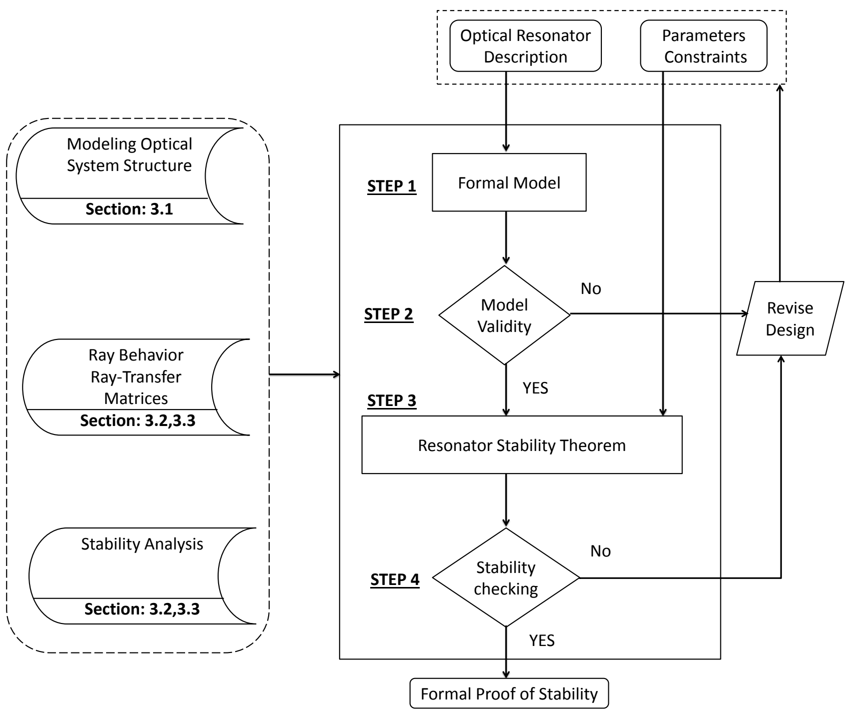


Figure 3. Procedure for the Stability Analysis

5. APPLICATION: STABILITY ANALYSIS OF FABRY PÉROT RESONATOR

Fabry Pérot (FP) resonator has been widely used in many applications such as reconfigurable Wavelength Division Multiplexing,[26] measurement of refractive index of cancer cells[22] and optical bio-sensing devices.[27] As a direct application of the framework developed in the previous sections, we present the step-by-step modeling and stability analysis of the Fabry Pérot (FP) resonator with spherical mirrors as shown in Figure 4.

STEP 1: The architecture of the given FP resonator is composed of two spherical mirrors with radius of curvature R, separated by a distance d and refractive index n. We can formally model this description by using the type definition of optical resonators (Definition 10). We formally model this resonator as follows:

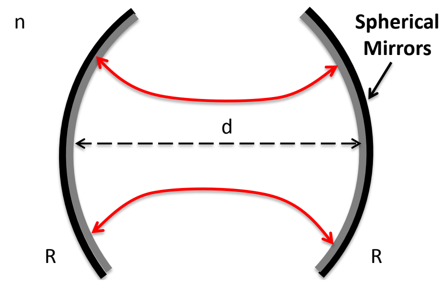


Figure 4. Fabry Pérot Resonator

Definition 14 (FP Resonator).
$\vdash$ ∀R d n. (fp_resonator R d n :resonator) = (spherical R,[],(n,d),spherical R)

Here the function fp_resonator takes three parameters i.e., Radius of curvature R, width of free space d and refraction index n. Note that [] represents an empty list of components because the given structure has no component between spherical interfaces but only a free space (n,d). The following is the corresponding HOL Light code:

```
let fp_resonator = new_definition
  ` fp_resonator   R d n :resonator = ((spherical R),[],(n,d),(spherical R)) `;;
```

STEP 2: Next, we verify that the FP resonator is indeed a valid resonator using predicate is_valid_resonator (Definition 11) as follows:

Theorem 4 (Valid FP resonator).
$\vdash$ ∀R d n. $R \neq 0 \wedge 0 \leq d \wedge 0 < n \Longrightarrow$ is_valid_resonator (fp_resonator R d n)

The proof of above theorem can be done automatically by a tactic VALID_RESONATOR_TAC. The corresponding HOL Light Code is given below:

```
let VALID_FP_CAVITY = prove
  (`!R d n.
    ~(R = &0) /\ &0 < d /\ &0 < n  ==> is_valid_resonator (fp_cavity R d n)`,

    VALID_RESONATOR_TAC fp_cavity);;;
```

STEP 3: In this step, we need to construct the stability theorem for FP resonator based on the parameters constraints as follows:

Theorem 5 (Stability of FP Resonator).
$\vdash$ ∀R d n. $R \neq 0 \wedge 0 < n \wedge 0 < \frac{d}{2} \wedge \frac{d}{2} < 2 \Longrightarrow$ is_stable_resonator (fp_resonator R d n)

The first two assumptions just ensure the validity of the model description. The two following ones provide the intended stability criteria.

STEP 4: This step deals with the formal verification of the above theorem which indeed requires Theorem 3 along with some fundamental properties of the matrices and arithmetic reasoning. The complete proof of this theorem is given in Appendix A.

The formal stability analysis of the FP resonators demonstrates the effectiveness of the proposed theorem proving based approach to reason about ray optics models. Due to the formal nature of the model and inherent soundness of higher-order logic theorem proving, we have been able to formalize some foundations of ray optics along with the verification of some useful theorems about optical components and systems with an unrivaled accuracy. It is important to note that all the theorems in our formalization are verified under universal quantification of systems parameter (e.g., radius of curvature and width of free space) unlike the other numerical approaches where the results hold only for specific values of these parameters. The main benefit of formal proofs is that all the underlying assumptions can be seen explicitly and proof-steps can be verified mechanically using a theorem prover. This improved accuracy comes at the cost of the time and effort spent, while formalizing the underlying theory of ray optics and optical resonators. The formalization took around 1500 lines of HOL Light code and 250 man-hours. But the availability of such a formalized infrastructure significantly reduces the time required for the modeling and stability analysis of an FP resonators. In fact, the formal stability analysis of the FP resonator application took just around 100 lines of HOL Light code and a couple of man-hours.

Note that theorem proving based analysis has not been applied in optical systems industry so far due to the limited research in this area and unfamiliarity of formal methods in optics community. In spite of the fact that our approach requires significant time to formalize the underlying theories of optics, we believe that our formal development can replace some time consuming simulations. For example, verification of the optical resonator stability is very time consuming because of the involvement of infinite set of rays. On the other hand, resonator stability can be verified in a very short time using the infrastructure developed in this research.

We have also formally analyzed the stability of an FP resonator with fiber-rod-lens and a Z-resonator. The source code developed in this research is available for download[28] and thus can be used by other researchers and engineers for the analysis of their optical systems.

6. CONCLUSION

In this paper, we proposed a novel approach for the analysis of optical systems based on theorem proving. We provided a brief introduction of the current state-of-the-art and highlighted their limitations. Next, we presented a brief overview of higher-order logic formalization of ray optics. We also presented the formalization of optical resonators and their stability. In order to show the practical effectiveness of our approach, we presented the stability analysis of the widely used Fabry Pérot resonator. The main strengths of our approach as compared to the other traditional techniques (such as simulation and paper-and-pencil proofs) are the soundness and accuracy.

We plan to extend this work to obtain an extensive library of verified optical components. We also plan to verify the sequential ray tracing algorithm[19] which is one of the most important part of many optical systems analysis tools.[10,29,30] In this paper, we only presented the ray optics formalization which is not sufficient to analyze all kind of optical systems. Thus it will be very interesting to make a formal connection of our work with the other formalizations such as electromagnetics[31] and quantum optics.[32] One of our long term goal is to package our HOL Light formalization in a GUI, so that it can be used by the non-formal methods community in industry for the analysis of practical resonators, and in academia for teaching and research purposes.

REFERENCES

[1] "The Hubble Space Telescope Optical System Failure Report," tech. rep., NASA (Novemeber 1990).

[2] Kogelnik, H. and Li, T., "Laser Beams and Resonators," *Appl. Opt.* **5**(10), 1550–1567 (1966).

[3] Nakazawa, M., Kubota, H., Sahara, A., and Tamura, K., "Time-domain ABCD Matrix Formalism for Laser Mode-Locking and Optical Pulse Transmission," *IEEE Journal of Quantum Electronics* **34**(7), 1075 –1081 (1998).

[4] Mookherjea, S., "Analysis of Optical Pulse Propagation with Two-by-Two (ABCD) Matrices," *Physical Review E* **64**(016611), 1–10 (2001).

[5] Cui, Q. C. . T. J. and Zhang, C., "Waves in Planar Waveguide Containing Chiral Nihility Metamaterial," *Optics and Communication* **274**, 317–321 (2007).

[6] Naqvi, A., "Comments on Waves in Planar Waveguide Containing Chiral Nihility Metamaterial," *Optics and Communication* **284**, 215–216 (Elsevier, 2011).

[7] reZonator. `http://www.rezonator.orion-project.org/` (2013).

[8] Su, B., Xue, J., Sun, L., Zhao, H., and Pei, X., "Generalised ABCD Matrix Treatment for Laser Resonators and Beam Propagation," *Optics & Laser Technology* **43**(7), 1318 – 1320 (2011).

[9] LASCAD. `http://www.las-cad.com/` (2013).

[10] OpticaSoftware. `http://www.opticasoftware.com/` (2013).

[11] Harrison, J., [*Theorem Proving with the Real Numbers*], Springer-Verlag (1998).

[12] Harrison, J., [*Handbook of Practical Logic and Automated Reasoning*], Cambridge University Press (2009).

[13] Jr., E. M. C., Grumberg, O., and Peled, D. A., [*Model Checking*], The MIT Press (1999).

[14] Gordon, M. J. C. and Melham, T. F., [*Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic*], Cambridge University Press (1993).

[15] T. Juhasz, G. D., Loesel, F. H., Kurtz, R. M., Horvath, C., Bille, J. F., and Mourou, G., "Applications of Femtosecond Lasers in Corneal Surgery," *Laser Physics* **10**(2), 495 – 500 (2011).

[16] Harrison, J., "A List of Theorem Provers." `http://www.cl.cam.ac.uk/users/jrh/ar.html` (2011).

[17] Harrison, J., "HOL Light: A Tutorial Introduction," in [*Proceedings of the First International Conference on Formal Methods in Computer-Aided Design (FMCAD'96)*], Srivas, M. and Camilleri, A., eds., *Lecture Notes in Computer Science* **1166**, 265–269, Springer-Verlag (1996).

[18] Harrison, J., "Formalized Mathematics," Tech. Rep. 36, Turku Centre for Computer Science (1996).

[19] Trger, F., ed., [*Springer Handbook of Lasers and Optics*], Springer, 2nd ed. (2012).

[20] Saleh, B. E. A. and Teich, M. C., [*Fundamentals of Photonics*], John Wiley & Sons, Inc. (1991).

[21] Siddique, U., Aravantinos, V., and Tahar, S., "Higher-Order Logic Formalization of Geometrical Optics," in [*International Workshop on Automated Deduction in Geometry*], 185–196, Informatics Research Report, School of Informatics, University of Edinburgh, Edinburgh, UK (2012).

[22] Song, W. Z., Zhang, X. M., Liu, A. Q., Lim, C. S., Yap, P. H., and Hosseini, H. M. M., "Refractive Index Measurement of Single Living Cells Using On-Chip Fabry-Perot Cavity," *Applied Physics Letters* **89**(20), 203901 (2006).

[23] Siegman, A. E., [*Lasers*], University Science Books, 1st ed. (1986).

[24] Sylvester, J. J., "The Collected Mathematical Papers of James Joseph Sylvester," **4**, Cambridge U. Press (1912).

[25] Siddique, U., Aravantinos, V., and Tahar, S., "Formal Stability Analysis of Optical Resonators," in [*NASA Formal Methods*], *Lecture Notes in Computer Science (LNCS)*, 368–382 (2013).

[26] Saadany, B., Malak, M., Kubota, M., Marty, F. M., Mita, Y., Khalil, D., and Bourouina, T., "Free-Space Tunable and Drop Optical Filters Using Vertical Bragg Mirrors on Silicon," *IEEE Journal of Selected Topics in Quantum Electronics* **12**(6), 1480 –1488 (2006).

[27] Baaske, M. and Vollmer, F., "Optical Resonator Biosensors: Molecular Diagnostic and Nanoparticle Detection on an Integrated Platform," *ChemPhysChem* **13**(2), 427–436 (2012).

[28] Siddique, U. and Aravantinos, V., "On the Formal Analysis of Geometrical Optics in HOL - HOL Light Script." `http://hvg.ece.concordia.ca/code/hol-light/goptics/` (2013).

[29] Synopsys-CODE-V. `http://optics.synopsys.com/codev/index.html` (2013).

[30] Radiant-Zemax. `http://radiantzemax.com/` (2013).

[31] Hasan, O., Afshar, S. K., and Tahar, S., "Formal Analysis of Optical Waveguides in HOL," in [*TPHOLs*], *Lecture Notes in Computer Science* **5674**, 228–243, Springer (2009).

[32] Mahmoud, M. Y., Aravantinos, V., and Tahar, S., "Formalization of Infinite Dimension Linear Spaces with Application to Quantum Theory," in [*NASA Formal Methods*], *Lecture Notes in Computer Science (LNCS)*, 413–427 (2013).

APPENDIX A. HOL LIGHT CODE OF STABILITY OF FP RESONATOR

```
(*------------------Stability Analysis of FP Resonator--------------------*)

let STABLE_FP =  prove
(`!R d n. ~(R = &0) /\ &0 < d/R /\ d/R < &2 /\  &0 < n /\ &0 < d
                   ==> is_stable_resonator (fp_resonator R d n)`,

REPEAT STRIP_TAC THEN MP_REWRITE_TAC STABILITY_THEOREM THEN
EXISTS_TAC(`(fp_mat_half R d):real^2^2`) THEN
REWRITE_TAC[MAT_POW2;fp_mat_half] THEN CONJ_TAC THENL
[ASM_SIMP_TAC[VALID_FP_CAVITY ];ALL_TAC] THEN CONJ_TAC THENL
[REWRITE_TAC[system_composition;unfold_resonator;LIST_POW;ONE;
            fp_cavity;round_trip;APPEND;optical_components_shift;
            LET_DEF;LET_END_DEF;REVERSE;interface_matrix;head_index;
            free_space_matrix] THEN
REPEAT(POP_ASSUM MP_TAC) THEN REWRITE_TAC[GSYM ONE] THEN
SIMP_TAC[GSYM MATRIX_MUL_ASSOC] THEN
SIMP_TAC[MAT2X2_MUL;REAL_MUL_RZERO;REAL_MUL_LZERO;REAL_ADD_LID;
         REAL_ADD_RID;REAL_MUL_LID;REAL_MUL_RID ] THEN
SIMP_TAC[MAT2X2_EQ ] THEN CONV_TAC REAL_FIELD; ALL_TAC] THEN
REPEAT(POP_ASSUM MP_TAC) THEN
REWRITE_TAC[DET_MAT2X2] THEN REWRITE_TAC[mat2x2;VECTOR_2] THEN
CONV_TAC REAL_FIELD);;
```