Specular Polynomials

ZHIMIN FAN, State Key Lab for Novel Software Technology, Nanjing University, China
JIE GUO*, State Key Lab for Novel Software Technology, Nanjing University, China
YIMING WANG and TIANYU XIAO, State Key Lab for Novel Software Technology, Nanjing University, China
HAO ZHANG, Southeast University, China
CHENXI ZHOU and ZHENYU CHEN, State Key Lab for Novel Software Technology, Nanjing University, China
PENGPEI HONG, University of Utah, United States of America
YANWEN GUO, State Key Lab for Novel Software Technology, Nanjing University, China
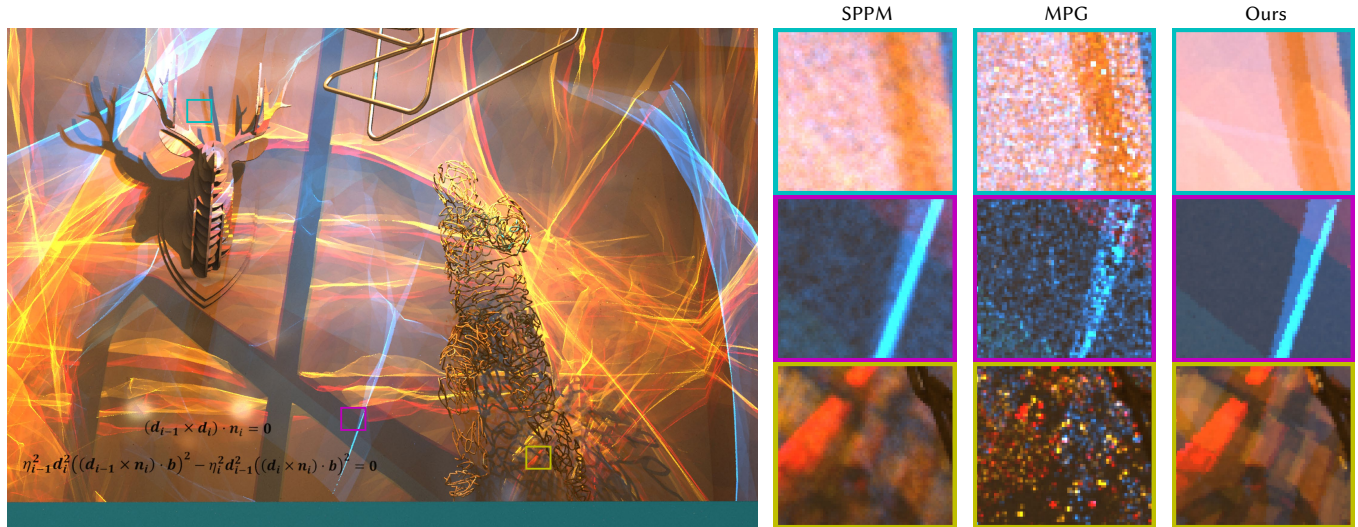LING-QI YAN, University of California, Santa Barbara, United States of America

Fig. 1. We render a shop window scene featuring challenging caustics and complex visibility, using our pipeline based on specular polynomials. The caustics stem from colored point light sources placed inside a dielectric object, and the whole scene is viewed through a transparent window. Such a configuration makes most existing rendering algorithms fail, while our method succeeds in reproducing the stunning light transport effect. The insets show equal-time (10 min) comparisons against Stochastic Progressive Photon Mapping (SPPM) [Hachisuka and Jensen 2009] and Manifold Path Guiding (MPG) [Fan et al. 2023].

Finding valid light paths that involve specular vertices in Monte Carlo rendering requires solving many non-linear, transcendental equations in

*Corresponding authors.

Authors' addresses: Zhimin Fan, zhiminfan2002@gmail.com, State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China; Jie Guo, guojie@nju.edu.cn, State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China; Yiming Wang, 02yimingwang@gmail.com; Tianyu Xiao, tianyuxiaoty@outlook.com, State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China; Hao Zhang, 213210060@seu.edu.cn, Southeast University, Nanjing, China; Chenxi Zhou, 502023320017@smail.nju.edu.cn; Zhenyu Chen, chenzy@smail.nju.edu.cn, State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China; Pengpei Hong, hpommpy@gmail.com, University of Utah, Salt Lake City, United States of America; Yanwen Guo, ywguo@nju.edu.cn, State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China; Ling-Qi Yan, lingqi@cs.ucsb.edu, University of California, Santa Barbara, Santa Barbara, United States of America.

high-dimensional space. Existing approaches heavily rely on Newton iterations in path space, which are limited to obtaining at most a single solution each time and easily diverge when initialized with improper seeds.

We propose *specular polynomials*, a Newton iteration-free methodology for finding a complete set of admissible specular paths connecting two arbitrary endpoints in a scene. The core is a reformulation of specular constraints into polynomial systems, which makes it possible to reduce the task to a univariate root-finding problem. We first derive bivariate systems utilizing rational coordinate mapping between the coordinates of consecutive vertices. Subsequently, we adopt the hidden variable resultant method for variable elimination, converting the problem into finding zeros of the determinant of univariate matrix polynomials. This can be effectively solved through Laplacian expansion for one bounce and a bisection solver for more bounces.

Our solution is generic, completely deterministic, accurate for the case of one bounce, and GPU-friendly. We develop efficient CPU and GPU implementations and apply them to challenging glints and caustic rendering. Experiments on various scenarios demonstrate the superiority of specular polynomial-based solutions compared to Newton iteration-based counterparts. Our implementation is available at https://github.com/mollnn/spoly.

CCS Concepts: • **Computing methodologies → Ray tracing**.

Additional Key Words and Phrases: Specular chain, Polynomial, Caustics, Glints

1 INTRODUCTION

Although Monte Carlo (MC) rendering algorithms have made enormous strides in reducing noise [Christensen and Jarosz 2016; Fascione et al. 2018; Keller et al. 2015], they are still inadequate for handling certain kinds of light paths. Paths containing specular chains (i.e., multiple consecutive specular scattering events) are extremely difficult to sample for existing MC rendering algorithms. The bottleneck is that a light path satisfying all physical constraints at specular reflective/refractive vertices has an infinitely small probability for sampling. To alleviate this issue, prior methods resort to some kind of multivariate Newton solver: seed paths are heuristically generated or randomly sampled first and then undergo Newton iterations in path space (e.g., manifold walk) to reach admissible paths [Jakob and Marschner 2012; Zeltner et al. 2020].

Unfortunately, Newton solvers do not always converge and are highly sensitive to the selection of initial seed paths. Improper seed paths can lead to divergence and hence introduce substantial bias or variance to the final rendering [Hanika et al. 2015; Wang et al. 2020; Zeltner et al. 2020]. Several recent works design dedicated strategies to carefully and intelligently select seed paths leveraging historical samples [Fan et al. 2023; Xu et al. 2023; Yu et al. 2023]. Even so, missing high-frequency optical details (e.g., glints and caustics) and producing severe outliers happen frequently in complex scenes. Moreover, Newton solvers that walk on the whole specular manifolds [Jakob and Marschner 2012] will also fail when facing scenes containing small specular geometries and complex visibility [Otsu et al. 2018], since they rely heavily on local continuity.

In this paper, we endeavor to remove the Newton solver from specular chain sampling, thereby fundamentally avoiding the aforementioned issues. Our key insight is that Newton's iteration-based methods rely solely on information about a single point at a time. While this aligns seamlessly with manifold-based techniques [Hanika et al. 2015; Jakob and Marschner 2012; Zeltner et al. 2020], there are also quite a few cases in which we possess comprehensive information about an entire region. Specifically, when considering paths passing through a tuple of triangles in Fig. 2, we can express the specular constraints in closed-form equations using vertex positions and normals [Walter et al. 2009; Wang et al. 2020]. Can we therefore obtain all of the solutions directly from these closed-form equations?

To this end, we derive **specular polynomials**, a polynomial formulation of specular constraints, and solve these polynomials directly, bypassing the use of multivariate Newton's method. Here, a significant challenge arises due to the high dimensionality involved. A specular chain with k vertices requires solving the problem in a space of $2k$ dimensions. To reduce the number of variables, we recursively apply rational mapping to vertex coordinates along a specular chain, resulting in **bivariate specular polynomials**.
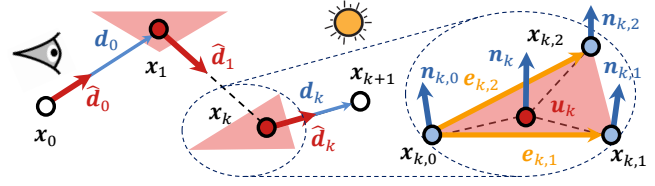
Fig. 2. Illustration of our problem setup and important symbols.

Leveraging the hidden variable resultant method tailored for polynomial equations [Nakatsukasa et al. 2015], we further convert them into **univariate specular polynomials**. This allows us to adopt a wide variety of methods in mathematics, such as root isolation [Collins and Loos 1976], bisection, and eigenvalue decomposition [Golub and Van Loan 2012], to effectively solve the simplified problem with only one variable.

When applied to the rendering of glints and caustics, our solver can be run on the triangle tuples selected using existing techniques [Wang et al. 2020] that prune non-contributing regions of the path space. This pipeline can simulate these challenging effects, consuming even less time than previous methods for one bounce and achieving comparable performance for more bounces. As a direct and deterministic method, it naturally handles discontinuities raised by small specular geometries and complex visibility, producing fewer artifacts such as strong outliers and energy loss.

In summary, our main contributions include:

- **A polynomial formulation of specular constraints**, derived by combining vertex constraint polynomials and rational coordinate mappings between barycentric coordinates.
- **A specular path solver** using hidden variable resultant method combined with direct or eigenvalue solvers, which is deterministic and free from multivariate Newton iterations.
- **Applications to glints and caustics rendering**, which achieves fast and almost noise-free rendering of specular light transport effects.

2 RELATED WORKS

Specular light transport. Since caustics and glints can not be robustly handled by (bidirectional) path tracing [Kajiya 1986; Veach and Guibas 1995] even when properly guided [Müller et al. 2017; Rath et al. 2023; Reibold et al. 2018; Ruppert et al. 2020; Vorba et al. 2019] or using Metropolis sampling [Veach and Guibas 1997], numerous specialized rendering methods have emerged. One line of approaches involves searching and root-finding techniques to identify all specular chains connecting two endpoints. Mitchell and Hanrahan [1992] utilize Fermat's principle with interval Newton's method to identify specular reflection paths including non-specular vertices. Chen and Arvo [2000] examine the reflection geometry by employing Fermat's principle to establish a path Jacobian and path Hessian, which addresses perturbations in the endpoint of a path and enhances the rendering of reflections on curved surfaces. Walter et al. [2009] introduce a pruning strategy and employ Newton's method to discover refractive specular paths. Loubet et al. [2020] analytically estimate the contribution of each specular triangle under far-field assumptions, enabling importance sampling of

specular vertices, though requiring recursive subdivision to reduce bias for pure specular cases. However, these methods are tailored for a single specular bounce. Wang et al. [2020] extend Walter et al. [2009]'s method to multi-bounce specular chains. To mitigate performance degradations, they resort to using the regular Newton's method instead of the interval variant, sacrificing some solutions for computational efficiency and practicality.

Alternatively, other methods alleviate the computational burden through stochastic sampling. Manifold Exploration Metropolis Light Transport (MEMLT) [Jakob and Marschner 2012] facilitates random walks on a specular manifold using Newton's method, which then extends to the half-vector space [Kaplanyan et al. 2014] and integrates into regular Monte Carlo sampling as Manifold Next-Event Estimation (MNEE) [Hanika et al. 2015]. MNEE augments both unidirectional and bidirectional techniques [Speierer et al. 2018]. However, with fixed initialization, MNEE can only identify at most one specular chain connecting a given pair of endpoints, resulting in energy loss. Specular Manifold Sampling (SMS) [Zeltner et al. 2020] tackles this challenge by employing random initialization, coupled with an unbiased reciprocal probability estimator. The convergence rate of SMS is subsequently improved by introducing a large jump [Jhang and Chang 2022]. Additionally, for the volumetric counterparts, Pediredla et al. [2020] and Fraboni et al. [2023] apply gradient-based optimizations to the problem of refractive radiative transfer. Nevertheless, the variance remains unbounded, as the convergence basin for each solution can become uncontrollably small, even when properly guided by reconstructed importance seed distributions [Fan et al. 2023].

In stark contrast to the gradient-based iteration methods that operate in path space, we introduce the first generic framework for computing all solutions without reliance on multivariate Newton iterations. The proposed framework combines analytical derivation with numerical computation, robustly handling the intricacies of specular constraints.

Specialized methods for rendering glints and caustics. Industrial caustics rendering has conventionally employed photon-based approaches [Hachisuka and Jensen 2009; Hachisuka et al. 2008; Jensen and Christensen 1995] or regularization [Jendersie and Grosch 2019; Kaplanyan and Dachsbacher 2013; Weier et al. 2021]. However, the inherent bias caused by spatial relaxation in these methods frequently results in issues such as light leaking and unexpected blurring, while our method solves the original, unrelaxed problem.

Specialized glint-rendering methods are mostly limited to a single reflection event and are designed exclusively for surfaces with normal maps [Yan et al. 2014; Zhu et al. 2022]. Alternatively, some works concentrate on rendering procedurally generated glints [Jakob et al. 2014]. In contrast, our approach handles actual geometric primitives, accommodating multiple bounce reflections and refractions, which successfully addresses both glints and caustic rendering tasks.

Polynomial root-finding techniques. Many mathematical tools for elimination and root-finding are tailored for polynomials. For univariate polynomials, real root isolation [Collins and Loos 1976] by differentiation can be used to develop robust solvers [Yuksel 2022]. Another approach involves solving the eigenvalues of the companion matrices through QR decomposition [Golub and Van Loan 2012].

Table 1. List of important symbols. By default, a vector with a hat means it is normalized.

Symbol	Description
x_0, x_{k+1}	Position of non-specular separators
x_i	Position of specular vertices
P_i	Position matrix ($p_{i,0}, p_{i,1}, p_{i,2}$)
$e_{i,1}, e_{i,2}$	Vector of triangle edges
n_i	Un-normalized linearly interpolated normal of x_i
$\hat{n}_i$	Normal vector of x_i
N_i	Normal matrix ($n_{i,0}, n_{i,1}, n_{i,2}$)
h_i	Generalized half-vector of x_i
$t_{i,1}, t_{i,2}$	Tangent vectors of x_i, computing from n_i and $e_{i,1/2}$
d_i	Position difference of vertices x_{i+1} and x_i
$\hat{d}_i$	Direction from x_i to x_{i+1}
u_i	Barycentric coordinate of x_i

Elimination methods can be employed to handle polynomial systems involving two or more variables [Buchberger 1992; Hilton et al. 1996]. The resultant method is efficient for discovering all solutions of multivariate polynomials systems [Nakatsukasa et al. 2015], which is widely embraced in many fields [Kajiya 1982; Kapur et al. 1994; Sadeghimanesh and England 2022].

Substantial research has focused on the resultant. Various resultant matrices, such as Sylvester [Sylvester 1853] and Bézout [Bézout 1779] for bivariate polynomial systems and Dixon [Dixon 1908] for multivariate (three or more) polynomial systems, have been explored. Fast algorithms with low time complexity for computing the resultant matrix have been developed [Chionh et al. 2002; Emiris and Pan 2005; Grenet et al. 2013; Qin et al. 2017].

In this work, we formulate the constraints as polynomials for elimination. We employ the Bézout resultant for bivariate polynomial systems predominantly due to its numerical stability and low computational complexity [Chionh et al. 2002]. Although existing works also mention some polynomial forms of specular constraints [Glaeser and Schröcker 2000], they are in a very different context.

3 POLYNOMIAL FORMS OF SPECULAR CONSTRAINTS

In this section, we convert the specular constraints into polynomial systems, starting with the definition of our problem setup.

3.1 Problem definition

Formally, we denote two fixed separators as x_0 and x_{k+1}, where x_0 could be a vertex on the camera or a non-specular shading point, and x_{k+1} could be a vertex on the light source or another non-specular shading point in bidirectional techniques [Fan et al. 2023; Jakob and Marschner 2012; Speierer et al. 2018]. A specular chain connecting x_0 and x_{k+1} is represented by $\overline{x}$, which is comprised of specular vertices $x_1, x_2, \ldots, x_k$. Important symbols used in this paper are summarized in Table 1 and illustrated in Fig. 2.

Existing pruning techniques [Walter et al. 2009; Wang et al. 2020] can be used to select the triangle tuples that may contain a specular chain connecting the two separators. Thus, our problem begins with a given tuple of k triangles $\mathcal{T}_1, \ldots, \mathcal{T}_k$, each consisting

of three vertices with positions[1] $P_i = (p_{i,0}, p_{i,1}, p_{i,2})$ and normals $N_i = (n_{i,0}, n_{i,1}, n_{i,2})$, our objective is to determine the set of all possible tuples of specular vertices $(x_1, \ldots, x_k)$ that forms admissible specular chains, where each vertex x_i lies on a triangle $\mathcal{T}_i$ with interpolated normal vectors $\hat{n}_i$. To make the problem tractable, we resort to the barycentric coordinates $u_i = (1 - u_i - v_i, u_i, v_i)^\top$ to represent the positions of these vertices:

$$x_i = (1 - u_i - v_i)p_{i,0} + u_i p_{i,1} + v_i p_{i,2} = P_i u_i. \tag{1}$$

Similarly, the normal vector $\hat{n}_i$ of specular vertices is also determined by the barycentric interpolation from the vertex normals:

$$\hat{n}_i = \frac{n_i}{\|n_i\|}, \quad n_i = (1 - u_i - v_i)n_{i,0} + u_i n_{i,1} + v_i n_{i,2} = N_i u_i. \tag{2}$$

Here, n_i is un-normalized and is linear to u_i. Normalizing n_i results in $\hat{n}_i$ which is no longer linear to u_i. For simplicity, we introduce dummy barycentric coordinates u_0 and u_{k+1} for separators. Eq. (1) still holds for $p_{0,0} = p_{0,1} = p_{0,2} = x_0$ and $p_{k+1,0} = p_{k+1,1} = p_{k+1,2} = x_{k+1}$, and u_0 and u_{k+1} will not appear in the final equations.

Ideal specular reflection or refraction occurs at each vertex, imposing constraints on the positions of consecutive vertices and their normal vectors. Specifically, for each specular vertex x_i, the constraint can be characterized by [Hanika et al. 2015; Kaplanyan et al. 2014; Wang et al. 2020; Zeltner et al. 2020]

$$h_i \times n_i = 0, 1 \le i \le k. \tag{3}$$

Here, $h_i = \eta_i \hat{d}_i - \eta_{i-1} \hat{d}_{i-1}$ is the generalized half vector [Walter et al. 2007], where

$$\hat{d}_i = \frac{d_i}{\|d_i\|}, \quad d_i = x_{i+1} - x_i, \tag{4}$$

and η_i represents the index of refraction of the outgoing side of x_i. Both h_i and n_i vary with respect to u_i and v_i.

Our problem is to solve the above constraint equations for variables $u_i = (u_i, v_i)$, which forms a system of $2k$ independent equations on $2k$ variables u_i and v_i ($i = 1, \ldots, k$). This has been proven to have only a finite number of solutions [Wang et al. 2020]. Unfortunately, these equations are not polynomials due to the existence of square roots and fractions. Thus, the rest of this section is devoted to transforming them into specular polynomials. Furthermore, we will attempt to maintain the degree and number of variables as low as possible, as this is critical to the performance of the solving process.

3.2 Vertex constraints

To convert the aforementioned constraint equations into polynomials, we first consider expressing the specular constraint on vertex x_i as a polynomial in variables u_{i-1}, u_i, and u_{i+1}. Because these constraints solely deal with the reflection/refraction behavior on a single vertex, they are referred to as **vertex constraints**. To clarify the derivation, we treat vertex constraints as two separate conditions: the coplanarity constraint and the angularity constraint.

Coplanarity constraint. The reflection/Snell's law states that the normal vector $\hat{n}_i$, the outgoing direction $\hat{d}_i$, and the incident direction $\hat{d}_{i-1}$ all reside in the same plane, i.e.,

$$\boxed{(d_{i-1} \times d_i) \cdot n_i = 0.} \tag{5}$$

Note that we have excluded the normalization factors, so this is already a polynomial equation in u_{i-1}, u_i, and u_{i+1}. We can simplify it further using $d_{i-1} + d_i = x_{i+1} - x_{i-1}$ and $d_{i-1} \times d_{i-1} = 0$:

$$\boxed{(d_{i-1} \times (x_{i+1} - x_{i-1})) \cdot n_i = 0.} \tag{6}$$

Angularity constraint. The incident and reflected/refracted angle values must satisfy the reflection/Snell's law:

$$\eta_{i-1} \|\hat{d}_{i-1} \times n_i\| = \eta_i \|\hat{d}_i \times n_i\|. \tag{7}$$

Since the coplanarity condition implies that $\hat{d}_{i-1}$, $\hat{d}_i$, and n_i lies on the same plane, $\hat{d}_{i-1} \times n_i$ and $\hat{d}_i \times n_i$ are parallel to each other, i.e.,

$$\frac{\hat{d}_{i-1} \times n_i}{\|\hat{d}_{i-1} \times n_i\|} = \frac{\hat{d}_i \times n_i}{\|\hat{d}_i \times n_i\|}. \tag{8}$$

Therefore, Eq. (7) becomes

$$\eta_{i-1}\hat{d}_{i-1} \times n_i = \eta_i \hat{d}_i \times n_i \tag{9}$$

with $\eta_i = \eta_{i-1}$ handling the reflection case.

Unfortunately, the square roots in $\hat{d}_{i-1}$ and $\hat{d}_i$ indicate that Eq. (9) is not a polynomial. We propose two polynomialization techniques to resolve this issue.

3.3 Polynomialization

Making the aforementioned angularity constraint a polynomial requires removing the square roots in the denominators. To achieve this goal, we propose two ways by computing squares or constructing common factors, respectively: The **square form** handles both reflection and refraction, but the **product form**, which has a lower degree than the square form, only works for reflection. Therefore, these two forms are used jointly in our final pipeline.

Square form. A straightforward way to remove the square roots is by performing squaring. To convert it into a scalar equation, we project[2] Eq. (9) onto an arbitrary vector b. Then, we eliminate the square roots by squaring both sides and remove the denominators by multiplying the common denominator $d_i^2 d_{i-1}^2$, resulting in[3]:

$$\boxed{\eta_{i-1}^2 d_i^2 \left((d_{i-1} \times n_i) \cdot b\right)^2 = \eta_i^2 d_{i-1}^2 \left((d_i \times n_i) \cdot b\right)^2.} \tag{10}$$

This equation represents a polynomial in terms of u_{i-1}, u_i, and u_{i+1}. In practice, we choose $b = (1, 0, 0)^\top$ by default and switch to $(0, 0, 1)^\top$ if the equation degenerates. Note that the square operations may introduce additional solutions with positive and negative signs, so we have to check the original constraints, Eq. (3), to remove such superfluous solutions.

Product form. An alternative method is to remove the normalization operation directly. This requires constructing two equations

---

[1] Here, we simplify the representation by using matrices (position matrix P_i and normal matrix N_i) to express the positions of the three vertices of a triangle. Thus, interpolations in Eq. (1) and Eq. (2) can be written as matrix-vector multiplications.

[2] Projection means computing the dot product of Eq. (9) with the basis b. It should also be noted that Eq. (10) is merely a necessary condition of Eq. (9). Yet, even if some superfluous solutions are discovered, they will be immediately filtered out after checking the specular constraints in path space. The same goes for Eq. (13).

[3] Here, we use the notation x^2 to represent the dot product $x \cdot x$ of a vector x.
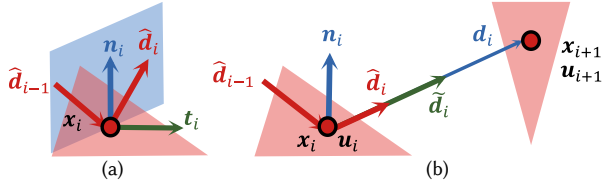
Fig. 3. (a) Product form of vertex constraints requires decomposing the ray in the local tangent frame of x_i. (b) Recursive rational mappings represent u_{i+1} using a rational expression of u_i.

both with factors $\|d_{i-1}\|$ and $\|d_i\|$. As illustrated in Fig. 3(a), for a reflective vertex x_i, we can exploit the symmetry property that the angle between $\hat{d}_{i-1}$ and n_i equals the angle between $-\hat{d}_i$ and n_i:

$$\hat{d}_{i-1} \cdot n_i = -\hat{d}_i \cdot n_i. \tag{11}$$

Supposing that an arbitrary direction t_i is perpendicular to n_i, the angle between $\hat{d}_{i-1}$ and t_i also equals the angle between $\hat{d}_i$ and t_i:

$$\hat{d}_i \cdot t_i = \hat{d}_{i-1} \cdot t_i. \tag{12}$$

These equations can be viewed as decomposing the incident and outgoing directions in the local tangent frame of x_i. Afterward, to eliminate the square root in the normalization factors of $\hat{d}_{i-1}$ and $\hat{d}_i$ in the above equations, we multiply them together. This results in a polynomial equation in terms of u_{i-1}, u_i, and u_{i+1}:

$$(d_{i-1} \cdot n_i)(d_i \cdot t_i) + (d_{i-1} \cdot t_i)(d_i \cdot n_i) = 0. \tag{13}$$

In practice, we choose $t_i = n_i \times e_{i,1}$ by default, where $e_{i,1} = p_{i,1} - p_{i,0}$. We switch to $t_i = n_i \times e_{i,2}$ if the equation degenerates.

Multivariate specular polynomials. Using the above two polynomialization algorithms, we can directly construct a multivariate polynomial system with $2k$ variables $u_1, \ldots, u_k$. We refer to this system as **multivariate specular polynomials**. Table 2 summarizes these polynomials. The maximum degree is 6 for refraction and 4 for reflection when using interpolated shading normals. Our proposed formulation still applies to specular triangles with face normals, where $n_i = n_{i,0}$ is a constant. In this case, the maximum degree is 4 for refraction and 2 for reflection.

Due to numerical instability and computational burden [Noferini and Townsend 2016; Qin et al. 2017], directly solving these multivariate formulations with many variables is not recommended. In mathematics, the most reliable solvers for polynomial systems are designed for two variables [Nakatsukasa et al. 2015]. Inspired by this, we further transform all multivariate formulations into bivariate ones, making the method much more practicable. Fortunately, this conversion can be accomplished through a change of variables leveraging the light transport behavior of specular paths.

3.4 Variable reduction

Our key idea for variable reduction lies in expressing the barycentric coordinates of all specular vertices using that of the first vertex in a given specular chain. This is achieved recursively: Each vertex is represented by its preceding neighbor, as illustrated in Fig. 3(b).

Specifically, we compute the intersection of the reflected/refracted ray with the $i+1$-th triangle $\mathcal{T}_{i+1}$ using the Möller–Trumbore algorithm [Akenine-Möller and Trumbore 1997], which represent u_{i+1} as a function of u_i and u_{i-1} in the following rational form named **rational coordinate mapping** in this paper:

$$u_{i+1}(u_i, u_{i-1}) = \frac{(\tilde{u}_{i+1}(u_i, u_{i-1}), \tilde{v}_{i+1}(u_i, u_{i-1}))^\top}{\kappa_{i+1}(u_i, u_{i-1})}, \tag{14}$$

where

$$\tilde{u}_{i+1}(u_i, u_{i-1}) = (\tilde{d}_i \times e_{i+1,2}) \cdot (x_i - p_{i+1,0}), \tag{15}$$

$$\tilde{v}_{i+1}(u_i, u_{i-1}) = ((x_i - p_{i+1,0}) \times e_{i+1,1}) \cdot \tilde{d}_i, \tag{16}$$

$$\kappa_{i+1}(u_i, u_{i-1}) = (\tilde{d}_i \times e_{i+1,2}) \cdot e_{i+1,1}. \tag{17}$$

Here, $\tilde{d}_i$ is a special and unnormalized version of $\hat{d}_i$, which is determined by the scattering type at x_i. To keep the expression of u_{i+1} rational, $\tilde{d}_i$ is also expected to be rational. These rational expressions are beneficial for polynomialization since rational equations can be easily converted into polynomial equations.

Reflection. The reflected direction at x_i can be written as

$$\hat{d}_i = -2(\hat{d}_{i-1} \cdot \hat{n}_i)\hat{n}_i + \hat{d}_{i-1}. \tag{18}$$

To eliminate the square roots in $\hat{n}_i$ and $\hat{d}_{i-1}$, we introduce a scaled direction vector $\tilde{d}_i$ through multiplying $\hat{d}_i$ by $n_i^2\sqrt{d_{i-1}^2}$:

$$\tilde{d}_i = -2(d_{i-1} \cdot n_i)n_i + d_{i-1}n_i^2, \tag{19}$$

which is a polynomial in u_{i-1} and u_i.

Refraction. The expression of the refracted direction at x_i is much more complicated:

$$\hat{d}_i = \eta_i'(\hat{d}_{i-1} - (\hat{d}_{i-1} \cdot \hat{n}_i)\hat{n}_i) - \sqrt{1 - \eta_i'^2(1 - (\hat{d}_{i-1} \cdot \hat{n}_i)^2)}\hat{n}_i, \tag{20}$$

where $\eta_i' = \eta_{i-1}/\eta_i$. Similarly, we also multiply the above equation by $n_i^2\sqrt{d_{i-1}^2}$, yielding

$$\tilde{d}_i = \eta_i'(d_{i-1}n_i^2 - (d_{i-1} \cdot n_i)n_i) - \sqrt{\beta_i}n_i, \tag{21}$$

where

$$\beta_i = n_i^2 d_{i-1}^2 - \eta_i'^2 \left(n_i^2 d_{i-1}^2 - (d_{i-1} \cdot n_i)^2 \right). \tag{22}$$

The square root operation in Eq. (21) forbids it to be rationalized accurately. Thus, we provide the following cheap piecewise rational approximation to $\sqrt{x}$ in the range of $[0, 1]$. We subdivide the the range $[0, 1]$ into 6 consecutive pieces. In each piece, $\sqrt{x}$ is fit by the following rational function:

$$\frac{c_{0,i} + c_{1,i}x}{d_{0,i} + d_{1,i}x}, \ i = 0, 1, 2 \ldots, 5. \tag{23}$$

The details of the coefficents $c_{0,i}, c_{1,i}, d_{i,0}, d_{1,i}$ are included in the supplemental material. The approximation provides an error less than 10^{-3}, which bounds the angular error within a threshold [Wang et al. 2020; Zeltner et al. 2020] and is sufficient for most cases.

Table 2. Summary of vertex constraint polynomials. We show the degree when using interpolated shading normals (I) and face normals (F).

Constraints	Formulation I Equation	Degree I/F	Formulation II Equation	Degree I/F
Coplanarity	Consecutive difference form $(\boldsymbol{d}_{i-1} \times \boldsymbol{d}_i) \cdot \boldsymbol{n}_i = 0$	3/2	Endpoint difference form $(\boldsymbol{d}_{i-1} \times (\boldsymbol{x}_{i+1} - \boldsymbol{x}_{i-1})) \cdot \boldsymbol{n}_i = 0$	2/1
Angularity	Square form (for reflection/refraction) $\eta_{i-1}^2 \boldsymbol{d}_i^2 \left((\boldsymbol{d}_{i-1} \times \boldsymbol{n}_i) \cdot \boldsymbol{b}\right)^2 - \eta_i^2 \boldsymbol{d}_{i-1}^2 \left((\boldsymbol{d}_i \times \boldsymbol{n}_i) \cdot \boldsymbol{b}\right)^2 = 0$	6/4	Product form (for reflection) $(\boldsymbol{d}_{i-1} \cdot \boldsymbol{n}_i)(\boldsymbol{d}_i \cdot \boldsymbol{t}_i) + (\boldsymbol{d}_{i-1} \cdot \boldsymbol{t}_i)(\boldsymbol{d}_i \cdot \boldsymbol{n}_i) = 0$	4/2

3.5 Bivariate specular polynomials

Until now, for any specular chain, we can use rational coordinate mapping (Sec. 4.4) to express all $\boldsymbol{u}_i$ in the chain using $\boldsymbol{u}_1$. By putting them into the vertex constraint (Sec. 4.3) on the last vertex $\boldsymbol{x}_k$, we arrive at a bivariate form of specular polynomials:

$$\begin{cases} a(\boldsymbol{u}_1) = a(u_1, v_1) = 0, \\ b(\boldsymbol{u}_1) = b(u_1, v_1) = 0, \end{cases} \tag{24}$$

with $\boldsymbol{u}_1 = (u_1, v_1)$ being the only variable[4]. This allows us to design practical solutions to simulate specular light transport with different types. We show three special cases in what follows. The notations follow Heckbert's tradition [Heckbert 1990]: R denotes reflection and T denotes refraction.

R. For specular reflection with a single bounce, Eq. (6) and Eq. (13) show that

$$\begin{cases} a(\boldsymbol{u}_1) = ((\boldsymbol{P}_1\boldsymbol{u}_1 - \boldsymbol{x}_0) \times (\boldsymbol{x}_2 - \boldsymbol{x}_0)) \cdot \boldsymbol{N}_1\boldsymbol{u}_1 = 0, \\ b(\boldsymbol{u}_1) = ((\boldsymbol{P}_1\boldsymbol{u}_1 - \boldsymbol{x}_0) \cdot \boldsymbol{N}_1\boldsymbol{u}_1)\left((\boldsymbol{x}_2 - \boldsymbol{P}_1\boldsymbol{u}_1) \cdot (\boldsymbol{N}_1\boldsymbol{u}_1 \times \boldsymbol{e}_{1,1})\right) + \\ \qquad ((\boldsymbol{x}_2 - \boldsymbol{P}_1\boldsymbol{u}_1) \cdot \boldsymbol{N}_1\boldsymbol{u}_1)\left((\boldsymbol{P}_1\boldsymbol{u}_1 - \boldsymbol{x}_0) \cdot (\boldsymbol{N}_1\boldsymbol{u}_1 \times \boldsymbol{e}_{1,1})\right) = 0. \end{cases} \tag{25}$$

The above two polynomials are of degree 2 and 4, respectively.

T. The derivation of the refractive case is analogous, using Eq. (6) and Eq. (10):

$$\begin{cases} a(\boldsymbol{u}_1) = ((\boldsymbol{P}_1\boldsymbol{u}_1 - \boldsymbol{x}_0) \times (\boldsymbol{x}_2 - \boldsymbol{x}_0)) \cdot \boldsymbol{N}_1\boldsymbol{u}_1 = 0, \\ b(\boldsymbol{u}_1) = \eta_0^2(\boldsymbol{x}_2 - \boldsymbol{P}_1\boldsymbol{u}_1)^2\left(((\boldsymbol{P}_1\boldsymbol{u}_1 - \boldsymbol{x}_0) \times \boldsymbol{N}_1\boldsymbol{u}_1) \cdot \boldsymbol{b}\right)^2 - \\ \qquad \eta_1^2(\boldsymbol{P}_1\boldsymbol{u}_1 - \boldsymbol{x}_0)^2\left(((\boldsymbol{x}_2 - \boldsymbol{P}_1\boldsymbol{u}_1) \times \boldsymbol{N}_1\boldsymbol{u}_1) \cdot \boldsymbol{b}\right)^2 = 0. \end{cases} \tag{26}$$

The above two polynomials are of degree 2 and 6, respectively.

RR. For the angularity relationship, we replace $\boldsymbol{d}_{i-1}$ in Eq. (13) with $\tilde{\boldsymbol{d}}_{i-1}$ defined in Eq. (19). Combined with the coplanarity constraint, we obtain

$$\begin{cases} a(\boldsymbol{u}_1) = ((\boldsymbol{P}_2\boldsymbol{u}_2 - \boldsymbol{P}_1\boldsymbol{u}_1) \times (\boldsymbol{x}_3 - \boldsymbol{P}_1\boldsymbol{u}_1)) \cdot \boldsymbol{N}_2\boldsymbol{u}_2 = 0, \\ b(\boldsymbol{u}_1) = (\tilde{\boldsymbol{d}}_1 \cdot \boldsymbol{n}_2)(\boldsymbol{d}_2 \cdot \boldsymbol{t}_2) + (\tilde{\boldsymbol{d}}_1 \cdot \boldsymbol{t}_2)(\boldsymbol{d}_2 \cdot \boldsymbol{n}_2) = 0. \end{cases} \tag{27}$$

Here, $\tilde{\boldsymbol{d}}_1$ is a polynomial in $\boldsymbol{u}_1$, while $\boldsymbol{n}_2$, $\boldsymbol{d}_2$, and $\boldsymbol{t}_2$ are all linear to $\boldsymbol{u}_2$. Recall that $\boldsymbol{u}_2$ is a rational expression of $\boldsymbol{u}_1$. Therefore, we can easily obtain polynomial systems in $\boldsymbol{u}_1$ from the above equations.

---

[4]Again, $\boldsymbol{u}_0$ and $\boldsymbol{u}_{k+1}$ are dummy variables since $\boldsymbol{x}_0$ and $\boldsymbol{x}_{k+1}$ are fixed.

Table 3. Summary of various forms of specular polynomials. Note that the time cost of solving a bivariate system is proportional to the cubic of the product degree of the two polynomials.

	Type	Equation	#Var.	Degree
Multivar.	R^k	Eqs. (6), (13)	$2k$	2, 4
	T^k	Eqs. (6), (10)	$2k$	2, 6
	$(R\|T)^k$	Eq. (3)	$3k+1$	2
Bivar.	R	Eqs. (6), (13)	2	2, 4
	T	Eqs. (6), (10)	2	2, 6
	RR	Eqs. (6), (13), (14)	2	10, 16
	RT, TR	Eqs. (6), (10), (14)	2	10, 24
	TT	Eqs. (6), (13), (14)	2	18, 48

Other cases. Table 3 summarizes the above theoretical results. The first three lines list the multivariate formulations of specular polynomials, while the other lines show the bivariate versions for one or two reflection/refraction bounces via rational coordinate mapping. The generalization to bounces over two is straightforward.

4 AN EFFICIENT POLYNOMIAL SOLVER

In this section, we develop a practical pipeline that effectively constructs and solves the aforementioned specular polynomials.

4.1 Variable elimination using resultants

Although we have converted specular constraints into simplified polynomials only containing two variables, finding the roots of the system is still challenging due to the high degree. However, many techniques exist in mathematical literature that can well solve the univariate polynomial with a high degree [Nakatsukasa et al. 2017; Yuksel 2022]. This inspires us to further eliminate one variable in our bivariate specular polynomials.

In this paper, we resort to the hidden variable resultant method [Nakatsukasa et al. 2015] due to its efficiency and robustness. Specifically, we eliminate one variable u_1, and convert the problem into finding the zeros of the Bézout resultant [Bézout 1779] $r(v_1)$, a univariate polynomial in terms of v_1. In this case, v_1 is called the hidden variable. The resultant $r(v_1)$ is defined as the determinant of the resultant matrix $\boldsymbol{R}(v_1)$ [Stiller 2004]:

$$r(v_1) = \det \boldsymbol{R}(v_1). \tag{28}$$

Here, for the Bézout resultant, the (i, j)-th element of the $n \times n$ matrix $\mathbf{R}(v_1)$ is given by[5] [Bézout 1779]

$$\sum_{k=0}^{\min(i, n-1-j)} \left(a_{i-k}(v_1)b_{j+1+k}(v_1) - b_{i-k}(v_1)a_{j+1+k}(v_1) \right), \quad (29)$$

where n is the degree of bivariate specular polynomials. We call $r(v_1)$ the **univariate specular polynomial**. It connects to bivariate specular polynomials via $a_i(v_1)$ and $b_i(v_1)$, which represent the coefficient of u_1^i in $a(u_1, v_1)$ and $b(u_1, v_1)$:

$$\begin{cases} a(u_1, v_1) = \sum_{i=0}^{n} a_i(v_1)u_1^i = 0, \\ b(u_1, v_1) = \sum_{i=0}^{n} b_i(v_1)u_1^i = 0. \end{cases} \quad (30)$$

This allows us to solve for one variable v_1 first and put it back into the original bivariate system to solve for the other variable u_1. We provide a running example to demonstrate how the resultant method works in the supplemental document.

4.2 Solving the univariate problem

The last building block of our whole pipeline is an efficient solver to the univariate equation:

$$r(v_1) = \det \mathbf{R}(v_1) = 0. \quad (31)$$

We perform Laplacian expansion for the determinant $\det \mathbf{R}(v_1)$ to obtain the coefficients of the resultant polynomial explicitly, and subsequently solve the univariate polynomial equations $r(v_1) = 0$. Then we find the roots of univariate polynomials in a recursive manner: The derivative of a polynomial of degree d is a polynomial of degree $d - 1$, and the zeros of the latter determine the monotonic pieces of the former. On each monotonic piece, only a single root exists and can be found through a bisection solver. Consistency is determined by checking whether the length of the bisection interval is smaller than a threshold, which is set to 10^{-9} in our implementation.

However, a significant drawback of Laplacian expansion is its exponential time complexity, making it only practical in the case of a single bounce. For large matrices produced by two or more bounces, we found it not practical to obtain the resultant coefficients explicitly. Instead, we opt to directly find the zeros by uniformly dividing $[0, 1]$ and running bisection on each piece of which the determinant on the two endpoints are with different signs. We evaluate the determinant using Gaussian elimination and find that 100 pieces and 10 bisection iterations work well in our test scenes.

Discussion. The piecewise bisection solver requires a large number of pieces in the case that solutions are clustered. However, our experiments show that the errors are negligible and have little influence on our test scenes. An alternative and theoretically comprehensive approach is by employing linearization [Golub and Van Loan 2012], which converts the above root-finding problem into a generalized eigenvalue problem. In mathematics, this can be well solved by QZ decomposition. Unfortunately, the computational burden of QZ decomposition is prohibitively high since the time complexity is $O(kn^6)$, where n is the degree of the bivariate polynomial system and k is the number of iterations. Developing a fast and accurate solver to Eq. (31) with an arbitrarily high degree is still an open problem in mathematics.
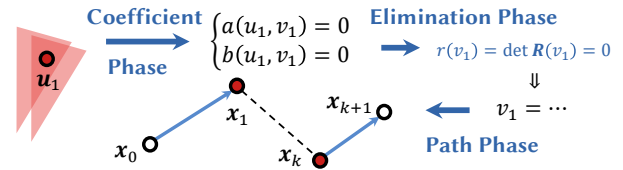

Fig. 4. Overview of the pipeline. Taking vertex positions and normals as inputs, our pipeline systematically constructs specular polynomials, converts the multivariate systems into a univariate problem via constructing resultant matrices, and subsequently solves the univariate problem. Finally, it validates the solutions and generates admissible paths.

4.3 The whole pipeline

In summary, the whole pipeline of our method consists of three phases shown in Fig. 4. Please refer to the supplemental document for more details and pseudo-code snippets.

Coefficient phase. We first convert the vertex positions and normals of triangles along a specular chain into bivariate polynomials on u_1 and v_1, according to the steps described in Sec. 4. The major task in this phase is to determine the coefficients of each specular polynomial.

Elimination phase. This phase converts the bivariate system on u_1 and v_1 into a univariate problem using the hidden variable resultant method introduced in Sec. 5.1. Here, each element of the resultant matrix $\mathbf{R}(v_1)$ is now a univariate polynomial in v_1. We only need to find all zeros of the determinant of the resultant matrix $\det \mathbf{R}(v_1)$. Then, we find all zeros of the matrix polynomial determinant using the method introduced in Sec. 5.2.

Path phase. As we have determined the solutions for v_1, we reintegrate them into the original polynomial $a(u_1, v_1)$ and solve for the other variable u_1, which only requires solving an univariate equation $a(u_1)|_{v_1}$ with explicitly known coefficients. Finally, we generate the path utilizing the solved barycentric coordinates, check the specular constraints in the path space [Wang et al. 2020; Zeltner et al. 2020], and conduct the visibility test. An admissible path will be rejected when the ray from any vertex $\mathbf{x}_i$ towards the next vertex $\mathbf{x}_{i+1}$ is blocked. If all the checks pass, our pipeline reports a valid solution and evaluates its contribution [Jakob and Marschner 2012; Wang et al. 2020]. Note that when there are multiple connections between two separators, we sum the contributions from all of them.

Complexity analysis. The time complexity for solving all chains given two separators is $O(t(n^4 + Cn^3))$. Here, C is the product of the number of intervals and bisection iterations. t is the number of triangle tuples, which depends on pruning techniques [Walter et al. 2009; Wang et al. 2020]. n is the degree of bivariate polynomials, which is approximately $4^k 2^r$ with r being the number of refractive
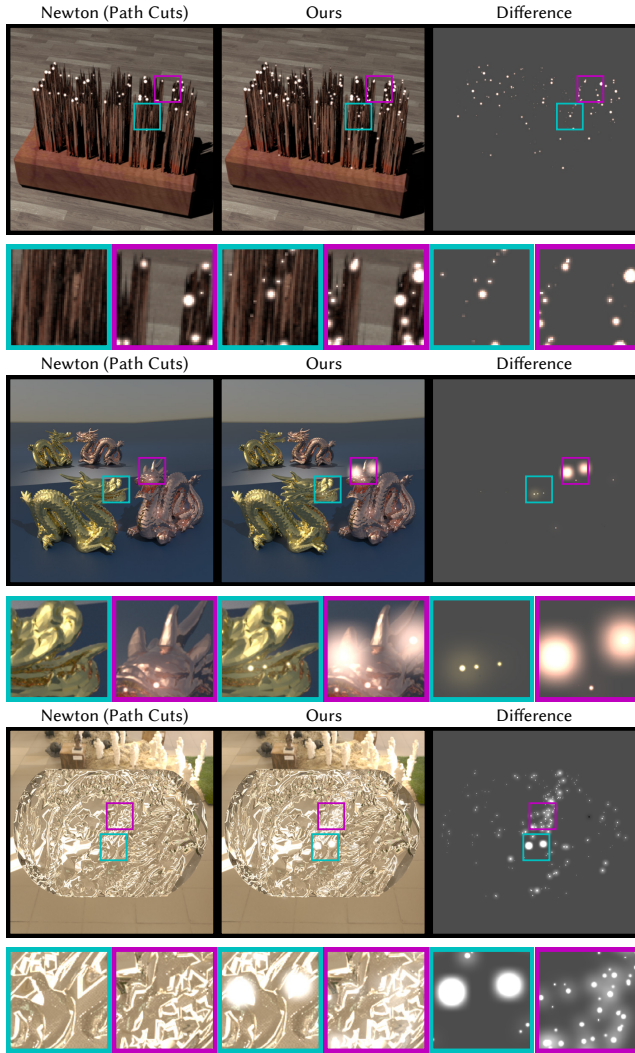
---

[5]Different algorithms can be used for constructing the Bézout resultant matrix. In our implementation, we adopt a fast computation method [Chionh et al. 2002]. The pseudo-code is presented in the supplemental document.

Fig. 5. Glints rendering featuring different types of specular chains. **Top**: *R* chains on a metal brush. **Middle**: *RR* chains in a scene with two specular dragons and a mirror. **Bottom**: *TT* chains passing through a relief, for which we also show glint-only images in the supplemental material.

vertices in the specular chain. The space complexity is $O(n^3 + C)$. Please refer to the supplemental material for a detailed discussion.

5 RESULTS

We have implemented our method on top of the Mitsuba renderer [Jakob 2010], and apply it on both glints rendering and caustics rendering. In the current implementation, we run our solver to find admissible specular paths for each tuple of triangles that passed the pruning of non-contributing tuples following Wang et al. [2020], which is based on a path space hierarchy and interval arithmetic bounds. For caustics rendering, we integrate our method into a conventional path tracer and perform our solver independently for each pair of separators. We sample the separators similarly

to Fan et al. [2023]; Zeltner et al. [2020]. The reference images of caustics rendering are generated by the unbiased variant of Specular Manifold Sampling (SMS) [Zeltner et al. 2020] with a very high sample rate. All timing measurements were conducted on a PC with a Core i9-13900KF processor and an RTX 4080 graphics card.

5.1 Glints rendering

In this task, we aim to find all admissible specular chains connecting a camera sample and a light sample in a deterministic way. We take Path Cuts [Wang et al. 2020], another deterministic method based on Newton iterations, as the competitor. The rendered images are post-processed by a bloom filter, to better show the high-frequency glints. As a state-of-the-art method in simulating pure specular paths, Path Cuts can find many challenging paths that are not affordable by conventional path tracers. However, it still runs the risk of missing some important admissible paths, since Newton's method involved in Path Cuts uses the center of triangles to construct seed paths and does not always converge, resulting in loss of glints as highlighted in the difference maps of Fig. 5. In these figures, we show glints stemming from different types of specular chains.

Instead, our method utilizes comprehensive information about each tuple of triangles along a specular path, converting the challenging light transport problem losslessly into a polynomial root-finding problem. Using the numerical tools discussed above, our method succeeds in producing rendering results that include significantly more glints than the existing method in our test scenes.

We do not compare our method to the interval Newton's method [Mitchell and Hanrahan 1992; Walter et al. 2009] since it requires several orders of magnitudes more time than the non-interval one, making it impractical for most scenes [Wang et al. 2020]. Instead, through polynomialization and variable reduction, we can solve the original problem very efficiently. The run-time cost of our method is generally in the same order of magnitude as the regular Newton's method, and is even faster than Newton's method in the case of a single bounce, while simultaneously finding more solutions.

5.2 Caustics rendering

For caustic rendering, we compare our method to conventional Path Tracing (PT) [Kajiya 1986], Practical Path Guiding (PPG) [Müller et al. 2017], the unbiased variant of Specular Manifold Sampling (SMS) [Zeltner et al. 2020] and Manifold Path Guiding (MPG) [Fan et al. 2023]. In particular, SMS and MPG are the state-of-the-art methods for caustics rendering. Unlike ours, both methods involve stochastic sampling to search specular chains. We show equal-time comparisons in Fig. 6.

Traditional guiding approaches face challenges when confronted with specular interactions due to the high-frequency variations inherent in the radiance distribution. Visual comparisons distinctly reveal the limitations of a typical guided path tracer, such as PPG, in handling complex light paths featuring tiny area light and specular vertices, resulting in outliers and energy loss. Similar issues will also happen with more advanced guiding approaches [Dodik et al. 2022; Rath et al. 2020, 2023; Ruppert et al. 2020].

SMS often exhibits noticeable noise, primarily due to its uniform sampling of seed chains. Moreover, an important issue with
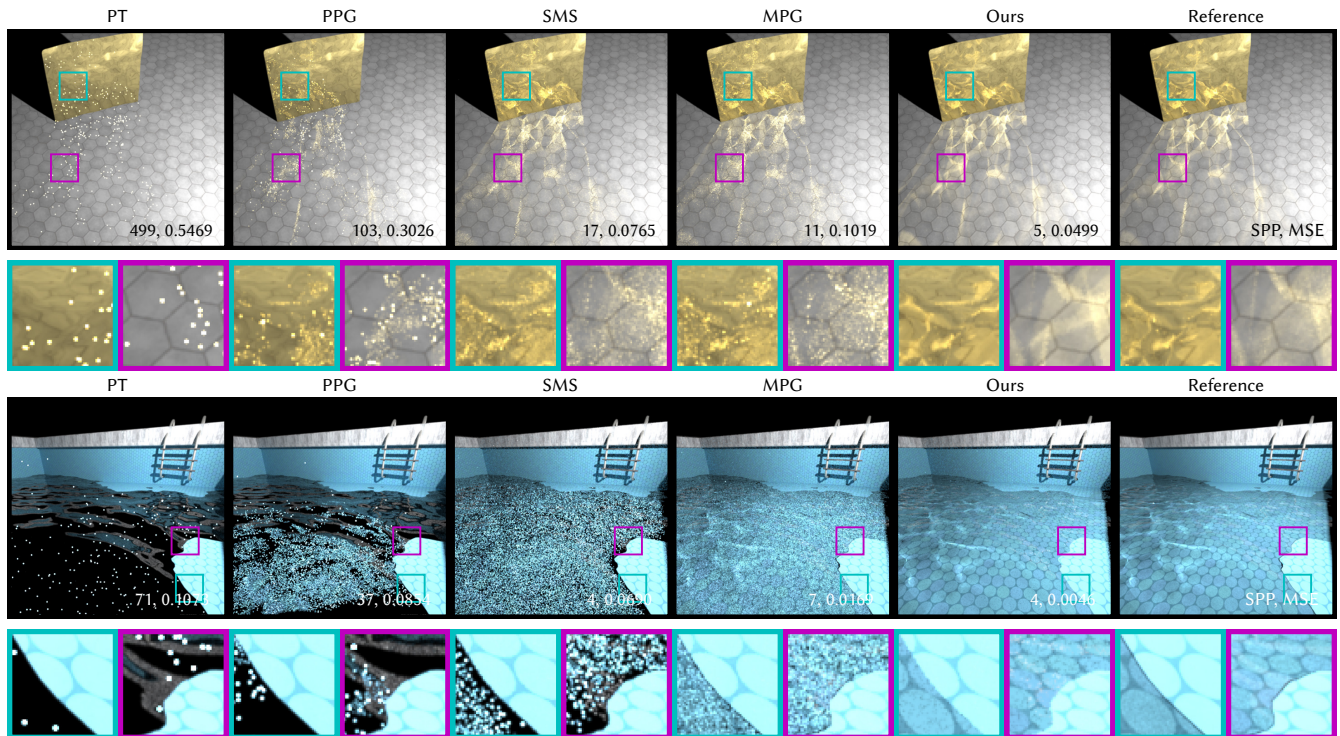
Fig. 6. Equal-time comparisons (10 sec) on caustics rendering with Path Tracing (PT) [Kajiya 1986], Practical Path Guiding (PPG) [Müller et al. 2017], the unbiased variant of Specular Manifold Sampling (SMS) [Zeltner et al. 2020] and Manifold Path Guiding (MPG) [Fan et al. 2023] on the Plane and Pool scenes. We report the samples per pixel (SPP) and quantitative errors in terms of mean square error (MSE).
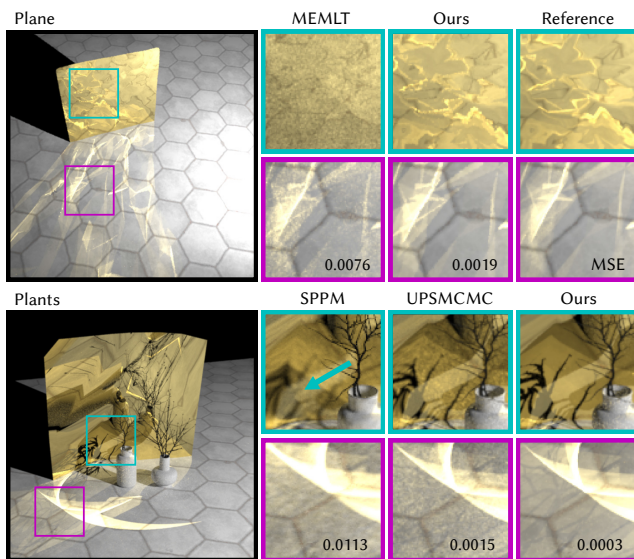

Fig. 7. **Top**: Equal-time (30 sec) comparison with Manifold Exploration Metropolis Light Transport (MEMLT) [Jakob and Marschner 2012]. **Bottom**: Equal-time comparison (10 sec) with Stochastic Progressive Photon Mapping (SPPM) [Hachisuka and Jensen 2009] and Metropolised Bidirectional Estimator (UPSMCMC) [Šik et al. 2016].

manifold-based methods is that they have an unbounded probability of finding a solution, which may be very small and lead to significantly high variance. This becomes particularly apparent in regions associated with admissible chains featuring extremely small convergence basins, such as the boundary of the water in the pool.

While MPG mitigates this issue by employing importance sampling of seed chains, the learned distribution requires a fairly long time to become accurate enough to perfectly fit the target function. As a result, some regions in the image rendered using MPG still exhibit high variance and outliers. Additionally, for manifold-based methods, the introduction of reciprocal probability estimation contributes to variance. Consequently, even in simple scenes (e.g., Plane) where the importance distributions are easy to fit, the rendering result is still with visible noise.

Our method is free from stochastic sampling, thus avoiding the noise and outliers faced by manifold-based methods. Consequently, our method generates results with very low variance and succeeds in finding all solutions in both two scenes.

In Fig. 7, we compare our method with Manifold Exploration Metropolis Light Transport (MEMLT) [Jakob and Marschner 2012; Veach and Guibas 1997]. Admissible chains are required by MEMLT as their seed paths. For SDS paths, it is inefficient to just rely on PT or BDPT for finding seed paths. This leads to either over-bright artifacts when Markov chains become trapped in small regions, or energy loss if no seed path exists for a given region. Working
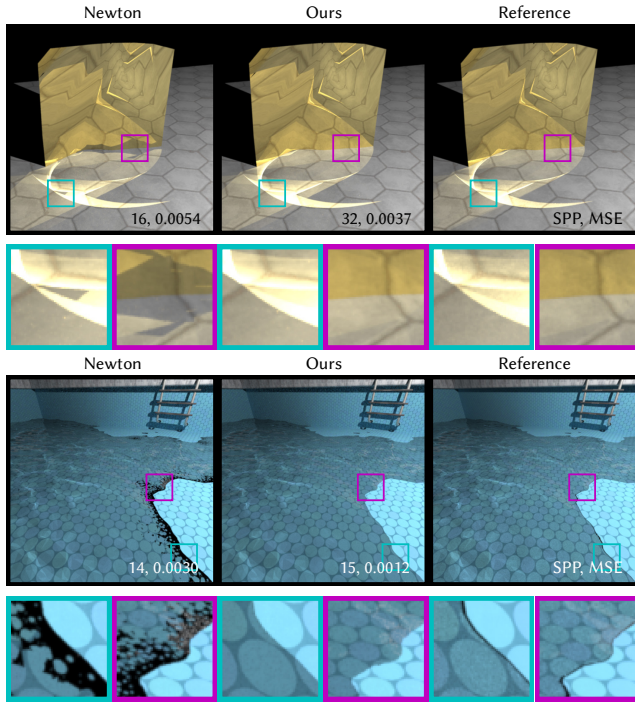
Fig. 8. Equal-time (30 sec) comparison between Newton's method and our specular polynomials on caustics rendering.

Fig. 9. Equal-sample (1 spp) comparison of caustics rendering featuring specular chains of *RR* type.

Fig. 10. Comparison of the bisection and eigenvalue solver. The eigenvalue solver uses 275× more time than our bisection solver.

as a deterministic method integrated into a standard Monte Carlo sampling framework such as path tracing, our approach eliminates blotchy energy loss in the images.

Additionally, we compare our method with photon-based approaches, in particular, Stochastic Progressive Photon Mapping (SPPM) [Hachisuka and Jensen 2009] and Metropolised Bidirectional Estimator (UPSMCMC) [Šik et al. 2016]. For SPPM, we reduce the photon lookup radius to achieve a balance between bias and variance. Besides visible noise, SPPM will cause over-blurriness (e.g., the cyan inset) due to the nature of density estimation. This problem also exists in UPSMCMC, which struggles to preserve the sharp edge of shadows in the cyan inset and simultaneously produces noticeable noise. In contrast, our deterministic approach solves for admissible paths directly and accurately, thus avoiding the issues of density estimation and yields almost noise-free rendering with detailed patterns of caustics.

5.3 Validations

Newton's method vs. specular polynomials on caustics rendering. To further verify that our deterministic method works well on caustics rendering, we replace our solver based on specular polynomials with the regular Newton's method. We adopt the Path Cuts framework to ensure that no stochastic sampling is involved, and the seed chain of Newton's method is heuristically generated by connecting the center of each triangle [Wang et al. 2020]. Therefore, both methods work in a deterministic way. From the comparison in Fig. 8, we see that regular Newton's method with deterministic heuristic
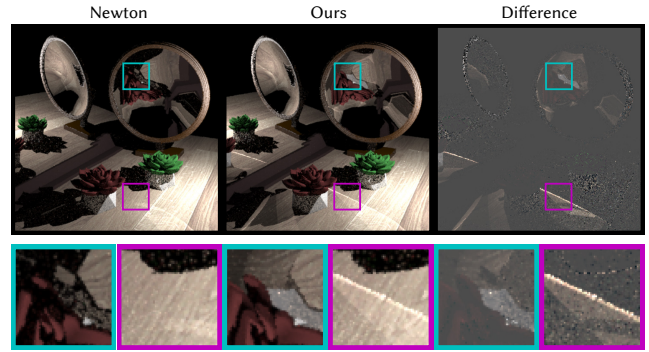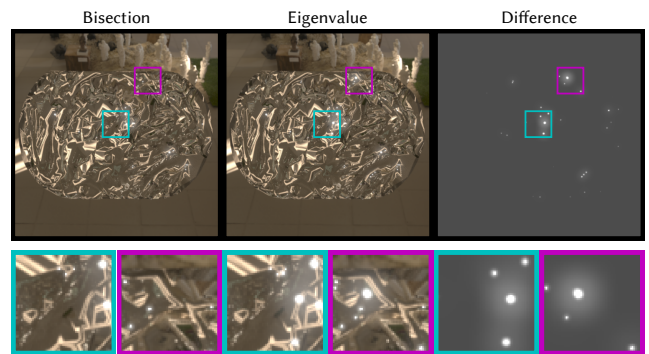
initialization is inadequate in identifying all solutions when a tuple of triangles encompasses multiple admissible paths. If the seed significantly deviates from the solution, Newton's method will diverge. Consequently, the rendering results suffer from energy loss and visual artifacts (e.g., the bottom of the plane), as shown in Fig. 8. In comparison, our method finds all the admissible paths in our test scenes, producing results that are nearly identical to the reference images.

We also apply our method to render caustics featuring specular chains of length two in Fig. 9. Since the dimensionality is high, the convergence basin of each solution for Newton's method may become extremely small. Simply relying on a heuristic initialization hardly finds the admissible paths in some regions, missing many caustics. Again, our method, using the bisection-based solver, works well, yielding caustics with complete shape and fine details.

Bisection solver vs. eigenvalue solver. In Fig. 10, we compare our direct solver using bisection with an eigenvalue-based solver using QZ decompositions. Eigenvalue solver in theory can ensure global convergence of the polynomial eigenvalue problem. However, its computational burden is extremely unaffordable. Our bisection-based solver finds almost all the solutions, generating rendering results of the same quality as QZ, consuming significantly less time.
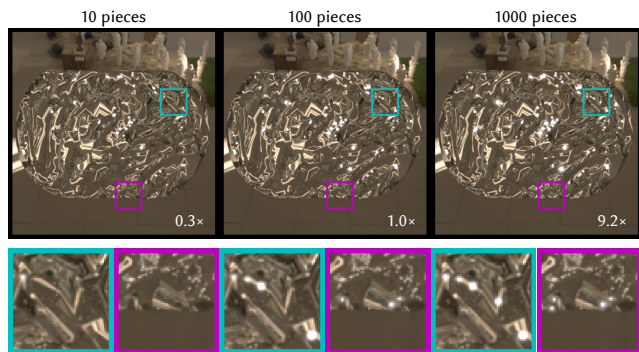
Fig. 11. Impact of the number of pieces in the bisection solver. The number shows the relative running time of our solver. Our choice of 100 pieces achieves an appropriate balance between accuracy and speed.

Impact of the number of pieces for bisection. Additionally, we verify the influence of the pieces on the bisection solver in Fig. 11. Using 10 pieces misses many solutions, whereas 1000 pieces greatly reduces performance. Our choice of 100 pieces works well in this complex scene of glints rendering.

5.4 Performance

In Table 4, we report the rendering statistics of glints rendering. For specular chains with type R, our CPU-based solver is 3.3× faster than regular Newton iterations. For type T, ours is also 2.5× faster. It is important to note that in both cases, the bottleneck, which takes nearly half of the time, is spent on finding the root of the univariate polynomial, which is dependent on the choice of the underlying solver. The construction of our bivariate specular polynomials and the conversion to univariate polynomials only takes a quite small amount of time.

For two specular bounces, we develop an efficient GPU implementation. Most parts of our method are polynomial arithmetics and determinant evaluations, which is GPU-friendly since the computation steps of each thread are the same. Thanks to the strong parallelization capability of GPU, our method achieves comparable performance with regular Newton's method, while finding significantly more solutions.

6 DISCUSSIONS AND LIMITATIONS

Accurate rational coordinate mapping for refraction. When dealing with refraction, polynomials naturally have limitations. It is challenging to polynomialize the square roots in the refracted direction expression. To address this problem, we employ first-order rational approximation. Nonetheless, this may lead to inaccuracy as shown in Fig. 12. Future works are required to find better approaches.

Better numerical methods for root-finding. Our primary focus in this work revolves around the derivation and simplification of specular polynomials. The bisection solver we used may find fewer solutions than the slow but comprehensive eigenvalue solver. Nevertheless, finding accurate and comprehensive solutions to high-order polynomial equations poses challenges due to their high time

Table 4. Time usage (μs) per triangle of Newton's method and our solver measured on glints rendering scenes. We show the time usage of constructing bivariate polynomials and the matrix R (Poly.), expanding determinants (Det.), and solving for v_1 and u_1 (Sol.). The timing for the bisection solver is divided into choosing intervals (Det.) and performing bisections (Sol. v_1).

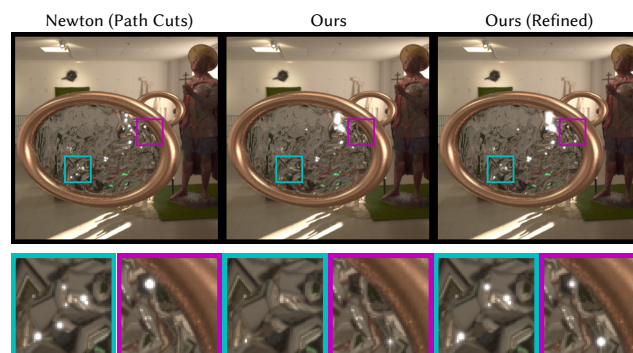| Type	Ours					Newton
	Poly.	Det.	Sol. v_1	Sol. u_1	Total	Total
R (CPU)	0.039	0.155	0.234	0.121	0.549	1.817
T (CPU)	0.221	0.329	0.283	0.157	0.990	2.427
RR (GPU)	0.649	1.617	1.931	0.739	4.936	0.306

Fig. 12. Limitations of the rational approximation for refraction. Small differences in the refracted direction amplify when the specular surfaces are far away from each other. Yet, simply refining our solutions with only one iteration of Newton's method leads to much better accuracy.

complexity and numerical instability. Notably, our polynomial systems exhibit specific characteristics. For instance, the coefficients are predominantly large only for low-order terms, as shown in Fig. 13. Employing better numerical tools tailored for such tasks remains a prospect for future exploration.

Long specular chains. While we primarily investigate small specular chains, our approach can also be applied to longer ones. However, because of the combinatorial explosion issue, it is not feasible to find all the specular chains in these situations. Combining our method with some stochastic method would be useful since unbiased specular transport involving three or more specular vertices can currently only be resolved through stochastic sampling utilizing reconstructed importance distributions.

Theoretical unbiasedness. Our specular polynomials are accurate except for the approximated rational expansion of refraction. However, comprehensive solving for two or more bounces requires a computationally expensive eigenvalue solver or a bisection solver with the number of pieces that tend to infinity. In our test scenes, our finite piece bisection solution performs well. Nonetheless, integrating a stochastic approach can be helpful if one wishes to guarantee unbiasedness [Misso et al. 2022; Zeltner et al. 2020].

Superfluous solutions. The use of the resultant method introduces superfluous solutions, i.e., solutions of the univariate polynomials
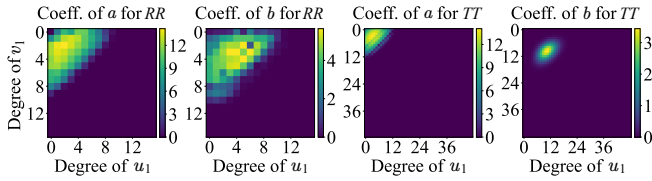
Fig. 13. Examples of the coefficients of bivariate specular polynomials. We show the absolute value of coefficients.

may not correspond to a valid specular path. For example, in the brush scene of Fig. 5, 52285 solutions of resultants are found, but only 6710 (13%) passes the check of path space constraints. However, this only influences the performance of solving for v_1, which is a relatively small part of the total computation time.

Handling near-specular vertices. Extending our method to support glossy chains is straightforward. After sampling the normal offset for glossy vertices, the admissible chains corresponding to the offset remain finite, and the problem reduces to pure specular situations. Thus, we sample the offset normal from the microfacet distribution before solving the specular chains connecting two separators, as in previous work [Fan et al. 2023; Hanika et al. 2015; Kaplanyan et al. 2014; Zeltner et al. 2020].

Surface representations. We have derived specular polynomials for triangles with interpolated normals, which are commonly utilized in existing rendering pipelines. Supporting other surface representations is a good avenue for future works.

7 CONCLUSION

Specular light transport, with the underlying multivariate root-finding problem, is a long-standing challenge for physically-based rendering. Existing methods simply perform Newton's method in the high-dimensional path space, which heavily relies on a proper seed sampling strategy and easily introduces high variance or bias.

As a fundamentally new methodology, specular polynomials proposed in this paper reformulate this problem into a univariate polynomial root-finding problem on a given interval, by applying polynomialization of vertex constraints, rational coordinate mapping, and the hidden variable resultant method. This makes the challenging light transport problem tractable and offers many benefits when integrated into existing rendering pipelines. Evaluations on the rendering of glints and caustics demonstrate our superiority in terms of accuracy and performance when compared to prior Newton iteration-based solutions.

We believe our work has made an important advance in dealing with unbounded convergence of Monte Carlo rendering and stochastic sampling for specular light transport, which may open up new research avenues for rendering intricate optical effects.

ACKNOWLEDGMENTS

REFERENCES

Tomas Akenine-Möller and Ben Trumbore. 1997. Fast, Minimum Storage Ray-Triangle Intersection. *J. Graphics, GPU, & Game Tools* 2 (1997), 21–28. https://api.semanticscholar.org/CorpusID:25931248

Bruno Buchberger. 1992. Gröbner Bases: An Introduction. In *Automata, Languages and Programming*, G. Goos, J. Hartmanis, and W. Kuich (Eds.), Vol. 623. Springer Berlin Heidelberg, Berlin, Heidelberg, 378–379. https://doi.org/10.1007/3-540-55719-9_89

Étienne Bézout. 1779. *Théorie Générale des Équations Algébriques*. Ph. D. Dissertation. Pierres, Paris.

Min Chen and James Arvo. 2000. Theory and application of specular path perturbation. *ACM Trans. Graph.* 19, 4 (oct 2000), 246–278. https://doi.org/10.1145/380666.380670

Eng-Wee Chionh, Ming Zhang, and Ronald N. Goldman. 2002. Fast Computation of the Bezout and Dixon Resultant Matrices. *Journal of Symbolic Computation* 33, 1 (Jan. 2002), 13–29. https://doi.org/10.1006/jsco.2001.0462

Per H. Christensen and Wojciech Jarosz. 2016. The Path to Path-Traced Movies. *Found. Trends. Comput. Graph. Vis.* 10, 2 (oct 2016), 103–175.

George E. Collins and Rüdiger Loos. 1976. Polynomial real root isolation by differentiation. In *Proceedings of the Third ACM Symposium on Symbolic and Algebraic Computation* (Yorktown Heights, New York, USA) *(SYMSAC '76)*. Association for Computing Machinery, New York, NY, USA, 15–25. https://doi.org/10.1145/800205.806319

A. L. Dixon. 1908. The Eliminant of Three Quantics in two Independent Variables: (Second Paper.). *Proceedings of The London Mathematical Society* (1908), 473–492.

Ana Dodik, Marios Papas, Cengiz Öztireli, and Thomas Müller. 2022. Path Guiding Using Spatio-Directional Mixture Models. *Computer Graphics Forum* 41, 1 (Feb. 2022), 172–189. https://doi.org/10.1111/cgf.14428

Ioannis Z. Emiris and Victor Y. Pan. 2005. Improved Algorithms for Computing Determinants and Resultants. *Journal of Complexity* 21, 1 (Feb. 2005), 43–71. https://doi.org/10.1016/j.jco.2004.03.003

Zhimin Fan, Pengpei Hong, Jie Guo, Changqing Zou, Yanwen Guo, and Ling-Qi Yan. 2023. Manifold Path Guiding for Importance Sampling Specular Chains. *ACM Trans. Graph.* 42, 6, Article 257 (Dec 2023), 14 pages.

Luca Fascione, Johannes Hanika, Rob Pieké, Ryusuke Villemin, Christophe Hery, Manuel Gamito, Luke Emrose, and André Mazzone. 2018. Path Tracing in Production. In *ACM SIGGRAPH 2018 Courses* (Vancouver, British Columbia, Canada) *(SIGGRAPH '18)*. Association for Computing Machinery, Article 15, 79 pages.

Basile Fraboni, Tsz Kin Chan, Thibault Vergne, and Jakub Jeziorski. 2023. Can You See the Heat? A Null-Scattering Approach for Refractive Volume Rendering. In *ACM SIGGRAPH 2023 Talks*. ACM, Los Angeles CA USA, 1–2. https://doi.org/10.1145/3587421.3595427

Georg Glaeser and Hans-Peter Schröcker. 2000. Reflections on refractions. *Journal for Geometry and Graphics* 4 (01 2000).

Gene H. Golub and Charles F. Van Loan. 2012. *Matrix Computations* (4th ed.). Johns Hopkins University Press.

Bruno Grenet, Pascal Koiran, and Natacha Portier. 2013. On the Complexity of the Multivariate Resultant. *Journal of Complexity* 29, 2 (April 2013), 142–157. https://doi.org/10.1016/j.jco.2012.10.001 arXiv:1210.1451 [cs]

Toshiya Hachisuka and Henrik Wann Jensen. 2009. Stochastic Progressive Photon Mapping. *ACM Trans. Graph.* 28, 4, Article 141 (2009), 8 pages. https://doi.org/10.1145/1661412.1618487

Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. 2008. Progressive photon mapping. *ACM Trans. Graph.* 27 (2008), 130.

Johannes Hanika, Marc Droske, and Luca Fascione. 2015. Manifold Next Event Estimation. *Computer Graphics Forum* 34, 4 (2015), 87–97. https://doi.org/10.1111/cgf.12681

Paul S. Heckbert. 1990. Adaptive Radiosity Textures for Bidirectional Ray Tracing. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques* (Dallas, TX, USA) *(SIGGRAPH '90)*. Association for Computing Machinery, New York, NY, USA, 145–154. https://doi.org/10.1145/97879.97895

A. Hilton, A.J. Stoddart, J. Illingworth, and T. Windeatt. 1996. Marching Triangles: Range Image Fusion for Complex Object Modelling. *Proceedings of 3rd IEEE International Conference on Image Processing* 1 (1996), 381–384. https://doi.org/10.1109/ICIP.1996.560840

Wenzel Jakob. 2010. Mitsuba renderer. http://www.mitsuba-renderer.org.

Wenzel Jakob, Miloš Hašan, Ling-Qi Yan, Ravi Ramamoorthi, and Steve Marschner. 2014. Discrete Stochastic Microfacet Models. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 33, 4 (July 2014), 115:1–115:10. https://doi.org/10.1145/2601097.2601186

Wenzel Jakob and Steve Marschner. 2012. Manifold Exploration: A Markov Chain Monte Carlo Technique for Rendering Scenes with Difficult Specular Transport. *ACM Trans. Graph.* 31, 4, Article 58 (jul 2012), 13 pages. https://doi.org/10.1145/2185520.2185554

Johannes Jendersie and Thorsten Grosch. 2019. Microfacet Model Regularization for Robust Light Transport. *Computer Graphics Forum* 38, 4 (2019), 39–47. https://doi.org/10.1111/cgf.13768

Henrik Wann Jensen and Niels Jørgen Christensen. 1995. Photon maps in bidirectional Monte Carlo ray tracing of complex objects. *Comput. Graph.* 19 (1995), 215–224.

Jia-Wun Jhang and Chun-Fa Chang. 2022. Specular Manifold Bisection Sampling for Caustics Rendering. *Computer Graphics Forum* (2022).

James T. Kajiya. 1982. Ray Tracing Parametric Patches. In *Proceedings of the 9th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '82)*. Association for Computing Machinery, New York, NY, USA, 245–254. https://doi.org/10.1145/800064.801287

James T. Kajiya. 1986. The Rendering Equation. *SIGGRAPH Comput. Graph.* 20, 4 (aug 1986), 143–150. https://doi.org/10.1145/15886.15902

Anton S. Kaplanyan and Carsten Dachsbacher. 2013. Path Space Regularization for Holistic and Robust Light Transport. *Computer Graphics Forum* 32, 2pt1 (2013), 63–72. https://doi.org/10.1111/cgf.12026

Anton S. Kaplanyan, Johannes Hanika, and Carsten Dachsbacher. 2014. The Natural-Constraint Representation of the Path Space for Efficient Light Transport Simulation. *ACM Trans. Graph.* 33, 4, Article 102 (jul 2014), 13 pages. https://doi.org/10.1145/2601097.2601108

Deepak Kapur, Tushar Saxena, and Lu Yang. 1994. Algebraic and Geometric Reasoning Using Dixon Resultants. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation - ISSAC '94*. ACM Press, Oxford, United Kingdom, 99–107. https://doi.org/10.1145/190347.190372

A. Keller, L. Fascione, M. Fajardo, I. Georgiev, P. Christensen, J. Hanika, C. Eisenacher, and G. Nichols. 2015. The Path Tracing Revolution in the Movie Industry. In *ACM SIGGRAPH 2015 Courses* (Los Angeles, California) *(SIGGRAPH '15)*. Association for Computing Machinery, New York, NY, USA, Article 24, 7 pages.

Guillaume Loubet, Tizian Zeltner, Nicolas Holzschuch, and Wenzel Jakob. 2020. Slope-Space Integrals for Specular next Event Estimation. *ACM Trans. Graph.* 39, 6, Article 239 (nov 2020), 13 pages. https://doi.org/10.1145/3414685.3417811

Zackary Misso, Benedikt Bitterli, Iliyan Georgiev, and Wojciech Jarosz. 2022. Unbiased and consistent rendering using biased estimators. *ACM Trans. Graph.* 41, 4, Article 48 (jul 2022), 13 pages. https://doi.org/10.1145/3528223.3530160

Don P. Mitchell and Pat Hanrahan. 1992. Illumination from curved reflectors. *Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (1992).

Thomas Müller, Markus Gross, and Jan Novák. 2017. Practical Path Guiding for Efficient Light-Transport Simulation. *Computer Graphics Forum* 36 (07 2017), 91–100. https://doi.org/10.1111/cgf.13227

Yuji Nakatsukasa, Vanni Noferini, and Alex Townsend. 2015. Computing the Common Zeros of Two Bivariate Functions via Bézout Resultants. *Numer. Math.* 129, 1 (Jan. 2015), 181–209. https://doi.org/10.1007/s00211-014-0635-z

Yuji Nakatsukasa, Vanni Noferini, and Alex Townsend. 2017. Vector Spaces of Linearizations for Matrix Polynomials: A Bivariate Polynomial Approach. *SIAM J. Matrix Anal. Appl.* 38, 1 (Jan. 2017), 1–29. https://doi.org/10.1137/15M1013286

Vanni Noferini and Alex Townsend. 2016. Numerical Instability of Resultant Methods for Multidimensional Rootfinding. *SIAM J. Numer. Anal.* 54, 2 (Jan. 2016), 719–743. https://doi.org/10.1137/15M1022513

Hisanari Otsu, Johannes Hanika, Toshiya Hachisuka, and Carsten Dachsbacher. 2018. Geometry-Aware Metropolis Light Transport. *ACM Trans. Graph.* 37, 6, Article 278 (dec 2018), 11 pages. https://doi.org/10.1145/3272127.3275106

Adithya Pediredla, Yasin Karimi Chalmiani, Matteo Giuseppe Scopelliti, Maysamreza Chamanzar, Srinivasa Narasimhan, and Ioannis Gkioulekas. 2020. Path Tracing Estimators for Refractive Radiative Transfer. *ACM Transactions on Graphics* 39, 6 (Dec. 2020), 1–15. https://doi.org/10.1145/3414685.3417793

Xiaolin Qin, Dingxiong Wu, Lin Tang, and Zhenyi Ji. 2017. Complexity of Constructing Dixon Resultant Matrix. *International Journal of Computer Mathematics* 94, 10 (Oct. 2017), 2074–2088. https://doi.org/10.1080/00207160.2016.1276572

Alexander Rath, Pascal Grittmann, Sebastian Herholz, Petr Vévoda, Philipp Slusallek, and Jaroslav Křivánek. 2020. Variance-Aware Path Guiding. *ACM Trans. Graph.* 39, 4, Article 151 (aug 2020), 12 pages. https://doi.org/10.1145/3386569.3392441

Alexander Rath, Ömercan Yazici, and Philipp Slusallek. 2023. Focal Path Guiding for Light Transport Simulation. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 30, 10 pages. https://doi.org/10.1145/3588432.3591543

Florian Reibold, Johannes Hanika, Alisa Jung, and Carsten Dachsbacher. 2018. Selective Guided Sampling with Complete Light Transport Paths. *ACM Trans. Graph.* 37, 6, Article 223 (dec 2018), 14 pages. https://doi.org/10.1145/3272127.3275030

Lukas Ruppert, Sebastian Herholz, and Hendrik P. A. Lensch. 2020. Robust Fitting of Parallax-Aware Mixtures for Path Guiding. *ACM Trans. Graph.* 39, 4, Article 147 (aug 2020), 15 pages. https://doi.org/10.1145/3386569.3392421

AmirHosein Sadeghimanesh and Matthew England. 2022. Resultant Tools for Parametric Polynomial Systems with Application to Population Models. arXiv:2201.13189 [cs, q-bio]

Sébastien Speierer, Christophe Hery, Ryusuke Villemin, and Wenzel Jakob Pixar. 2018. Caustic Connection Strategies for Bidirectional Path Tracing.

Peter F. Stiller. 2004. An Introduction to the Theory of Resultants. https://api.semanticscholar.org/CorpusID:31644195

James Joseph Sylvester. 1853. On a Theory of the Syzygetic Relations of Two Rational Integral Functions, Comprising an Application to the Theory of Sturm's Functions, and That of the Greatest Algebraical Common Measure. *Philosophical Transactions of the Royal Society of London* 143 (1853), 407–548. https://doi.org/10.1098/rstl.1853.0018

Eric Veach and Leonidas Guibas. 1995. Bidirectional Estimators for Light Transport. In *Photorealistic Rendering Techniques*, Georgios Sakas, Stefan Müller, and Peter Shirley (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 145–167.

Eric Veach and Leonidas J. Guibas. 1997. Metropolis Light Transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*. ACM Press/Addison-Wesley Publishing Co., USA, 65–76. https://doi.org/10.1145/258734.258775

Jiří Vorba, Johannes Hanika, Sebastian Herholz, Thomas Müller, Jaroslav Křivánek, and Alexander Keller. 2019. Path Guiding in Production. , Article 18 (2019), 77 pages. https://doi.org/10.1145/3305366.3328091

Martin Šik, Hisanari Otsu, Toshiya Hachisuka, and Jaroslav Křivánek. 2016. Robust Light Transport Simulation via Metropolised Bidirectional Estimators. *ACM Trans. Graph.* 35, 6, Article 245 (dec 2016), 12 pages. https://doi.org/10.1145/2980179.2982411

Bruce Walter, Steve Marschner, Hongsong Li, and Kenneth E. Torrance. 2007. Microfacet Models for Refraction through Rough Surfaces. In *Rendering Techniques*.

Bruce Walter, Shuang Zhao, Nicolas Holzschuch, and Kavita Bala. 2009. Single Scattering in Refractive Media with Triangle Mesh Boundaries. *ACM Trans. Graph.* 28, 3, Article 92 (jul 2009), 8 pages. https://doi.org/10.1145/1531326.1531398

Beibei Wang, Miloš Hašan, and Ling-Qi Yan. 2020. Path Cuts: Efficient Rendering of Pure Specular Light Transport. *ACM Trans. Graph.* 39, 6, Article 238 (nov 2020), 12 pages. https://doi.org/10.1145/3414685.3417792

Philippe Weier, Marc Droske, Johannes Hanika, Andrea Weidlich, and Jiří Vorba. 2021. Optimised Path Space Regularisation. *Computer Graphics Forum* 40, 4 (2021), 139–151. https://doi.org/10.1111/cgf.14347

Xiaofeng Xu, Lu Wang, and Beibei Wang. 2023. Efficient Caustics Rendering via Spatial and Temporal Path Reuse. In *Computer Graphics Forum*. Wiley Online Library, e14975.

Ling-Qi Yan, Miloš Hašan, Wenzel Jakob, Jason Lawrence, Steve Marschner, and Ravi Ramamoorthi. 2014. Rendering glints on high-resolution normal-mapped specular surfaces. *ACM Trans. Graph.* 33, 4, Article 116 (jul 2014), 9 pages. https://doi.org/10.1145/2601097.2601155

Rui Yu, Yue Dong, Youkang Kong, and Xin Tong. 2023. Neural Path Sampling for Rendering Pure Specular Light Transport. *Computer Graphics Forum* (2023). https://api.semanticscholar.org/CorpusID:266443113

Cem Yuksel. 2022. High-Performance Polynomial Root Finding for Graphics. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 5, 3 (July 2022), 1–15. https://doi.org/10.1145/3543865

Tizian Zeltner, Iliyan Georgiev, and Wenzel Jakob. 2020. Specular Manifold Sampling for Rendering High-Frequency Caustics and Glints. *ACM Trans. Graph.* 39, 4, Article 149 (jul 2020), 15 pages. https://doi.org/10.1145/3386569.3392408

Junqiu Zhu, Sizhe Zhao, Yanning Xu, Xiangxu Meng, Lu Wang, and Ling-Qi Yan. 2022. Recent advances in glinty appearance rendering. *Computational Visual Media* 8 (06 2022), 535–552. https://doi.org/10.1007/s41095-022-0280-x