こうろせつだん！

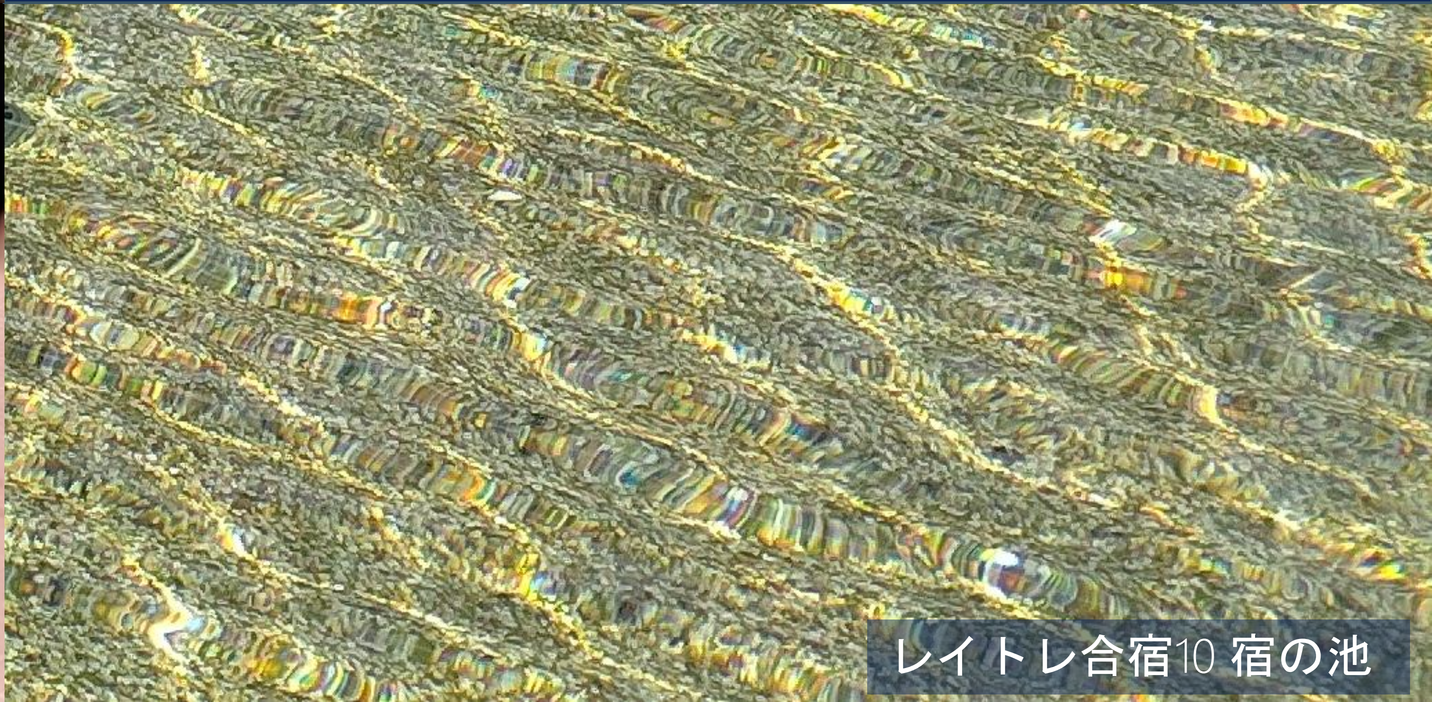
INTRODUCTION TO PATH CUTS

ushiostarfish

Caustics is an Endless Dream on Ray Tracing

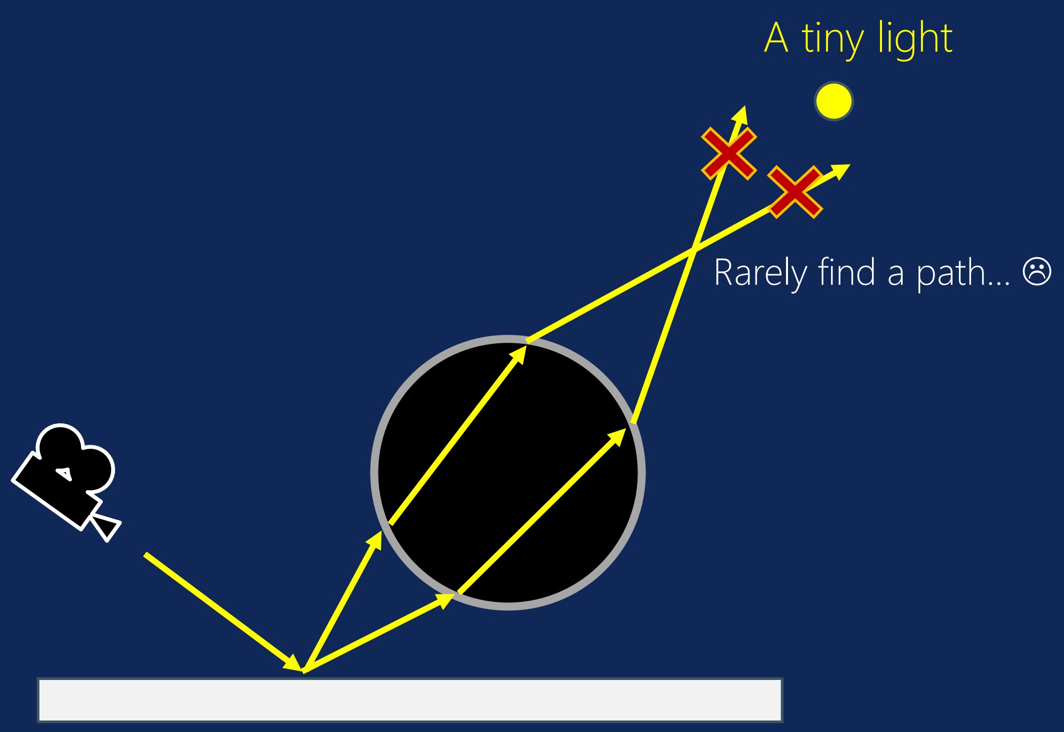
山梨・金精軒 水信玄餅 おいしかった

レイトレ合宿7 shockerさんからのもらいもの

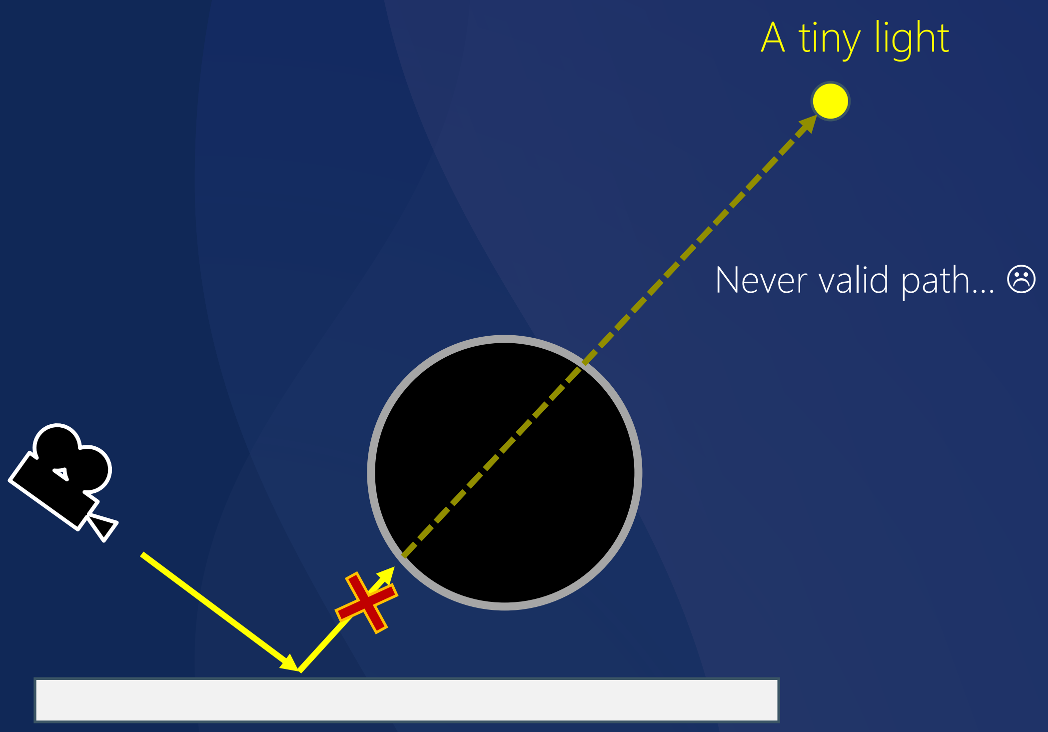


レイトレ合宿10 宿の池

DIFFICULTY OF CAUSTICS IN PATH TRACING



BRDF sampling



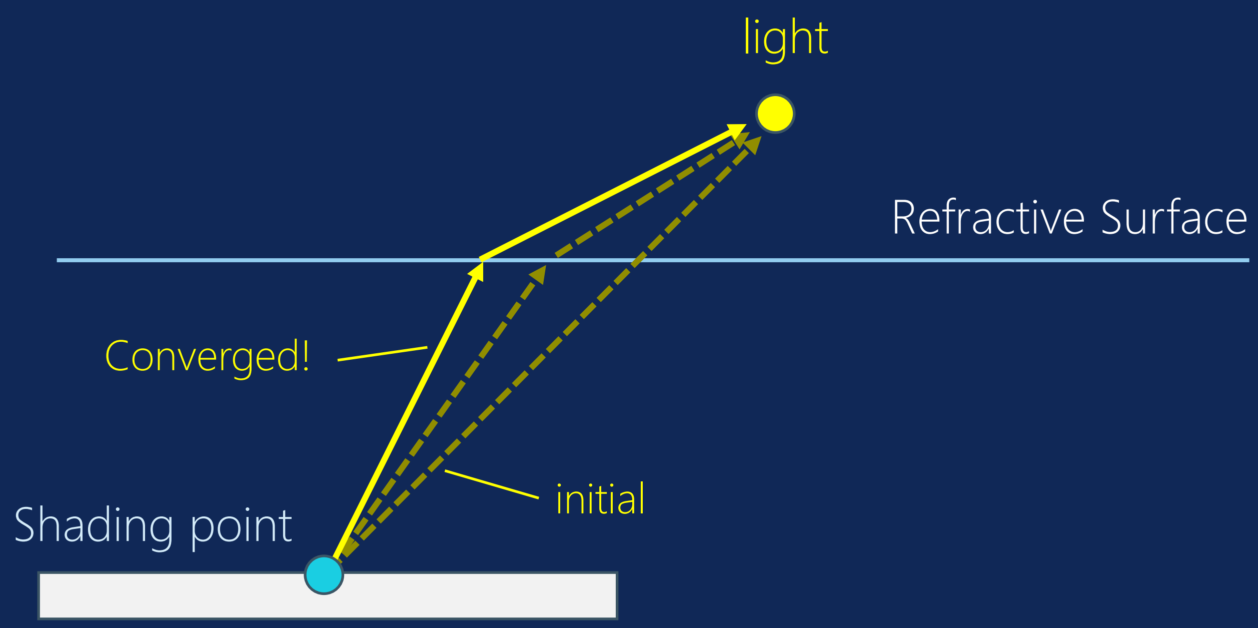
Light sampling

GENERAL LIGHT TRANSPORT ALGORITHMS SUITABLE FOR CAUSTICS

- Photon Mapping
- Bidirectional path tracing
- Vertex connection and merging
- Metropolis light transport

LIGHT TRANSPORT ALGORITHMS **DEDICATED** FOR CAUSTICS

- Walter, et al. "Single Scattering in Refractive Media with Triangle Mesh Boundaries" [2009]
  - Just 1 refraction
  - Newton-Raphson to find a contributable path
    - Minimize the cost function
  - Fully deterministic to find all of the contributable path☺



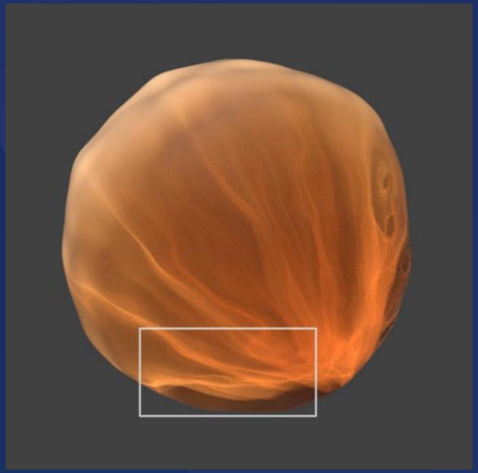


Figure 7 on the paper

An Example of Cost Function

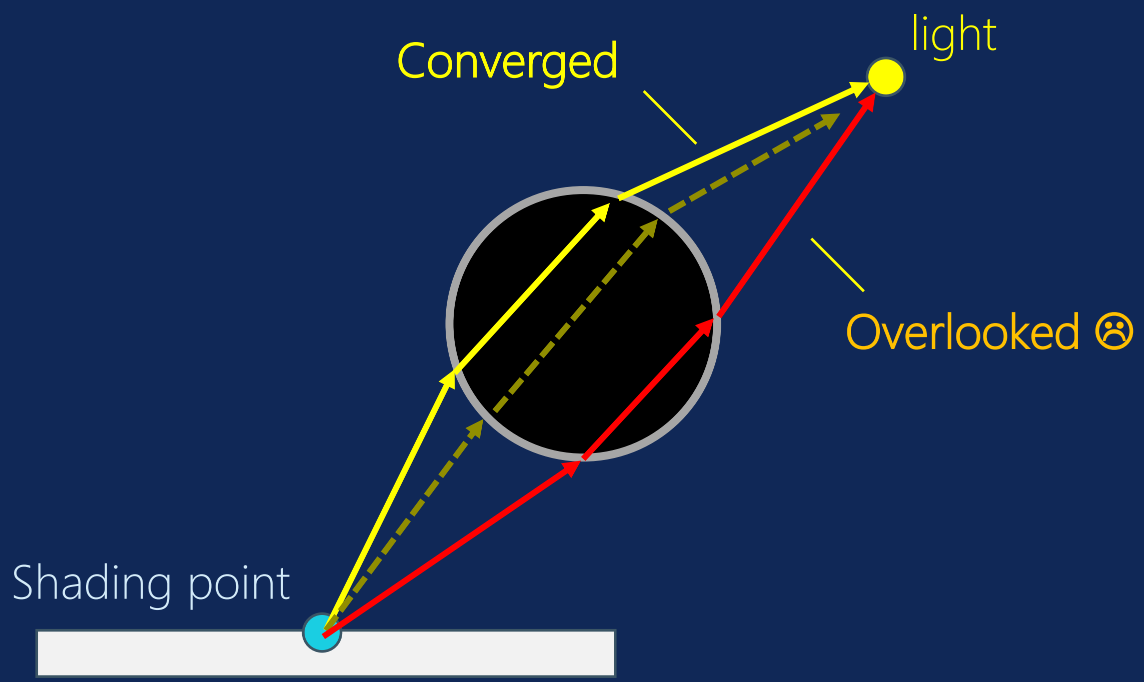
Super Simple !! ☺

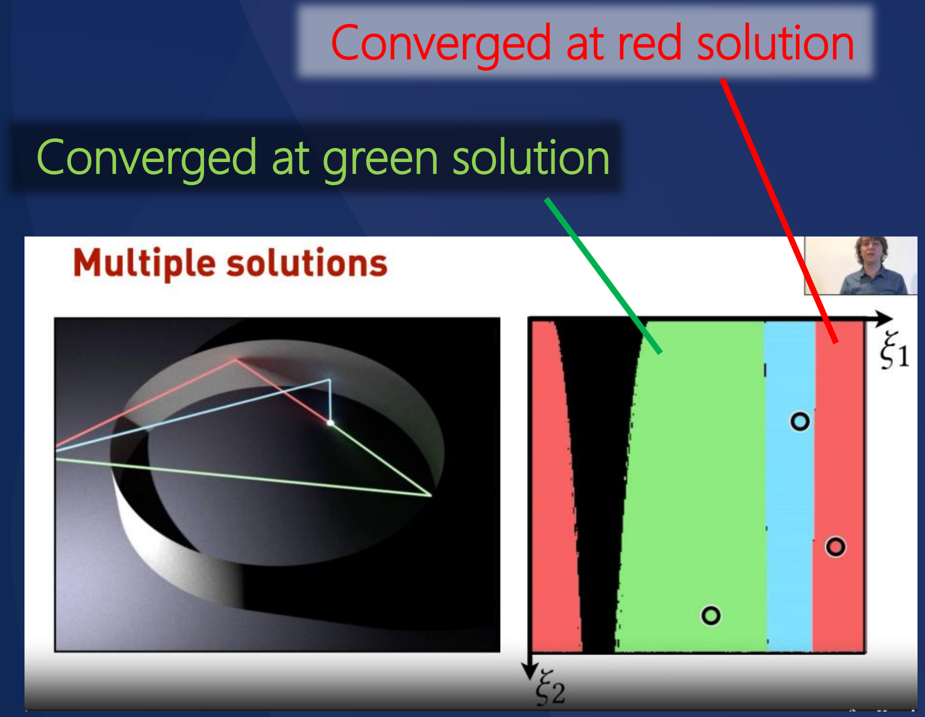
$$C_t = \hat{\boldsymbol{n}} - \frac{h_t}{|h_t|}$$

where $h_t = -(\eta_i \omega_i + \eta_o \omega_o)$

A FUNDAMENTAL ISSUE

- Newton-Raphson itself is for the local minimum for finding minimum
  - high sensitivity to initial conditions





Specular Manifold Sampling (Talk Video)

TWO TYPES OF APPROACHES FOR MULTIPLE SOLUTIONS

- Walter, et al. "Single Scattering in Refractive Media with Triangle Mesh Boundaries" [2009]

How do you handle multiple solutions?

Point Sampling

- Jakob and Marschner, "Manifold Exploration" [2012]
- Hanika, et al., "Manifold NEE" [2015]
- Hanika et al., "Specular Manifold Sampling" [2020]

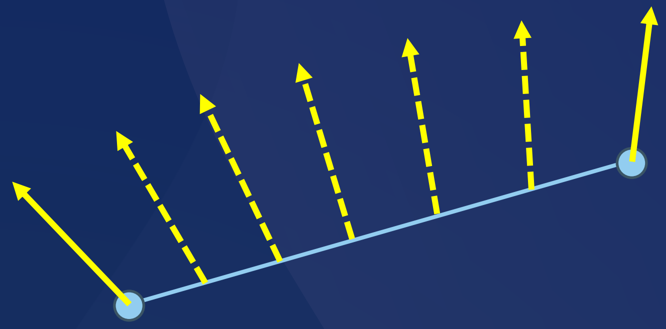
Primitive Search

Today's Focus

- Wang, at al, "Path Cuts" [2020]
  - Extending to multi dimensions
- Fan, at al, "Specular Polynomials" [2024]
- Fan, at al, "Bernstein Bounds for Caustics" [2025]
  - Stochastic Culling

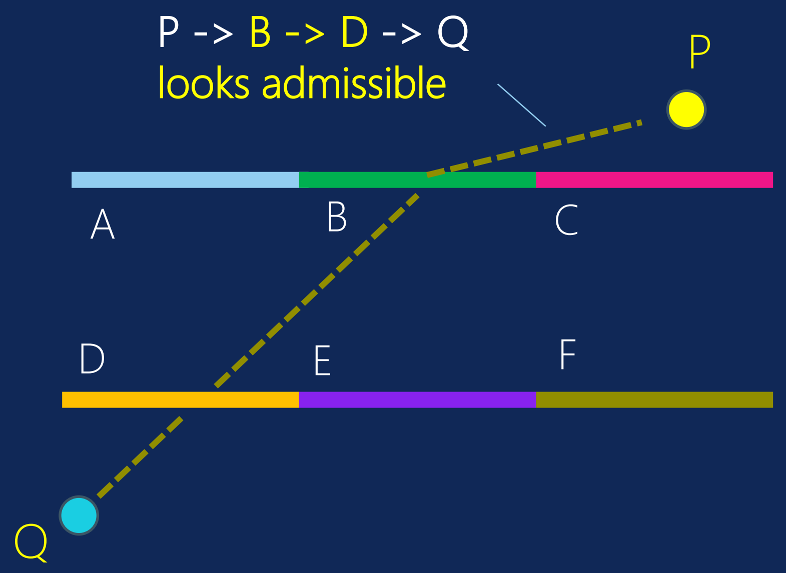
ASSUMPTIONS FOR PATH CUTS

- Triangle only
- Phong normal interpolation only
  - No normal mapping
  - A weak point of the current primitive approach

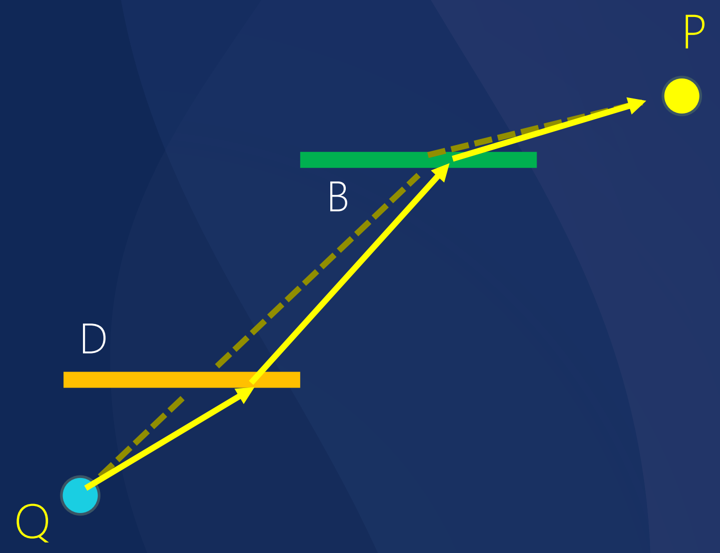


OVERVIEW OF PATHCUTS

- Two-phases approach
  - [Stage 1] Enumerate admissible tuples of triangles
  - [Stage 2] Newton-Raphson iterations to find a valid path



Stage 1. Finding Admissible Tuples

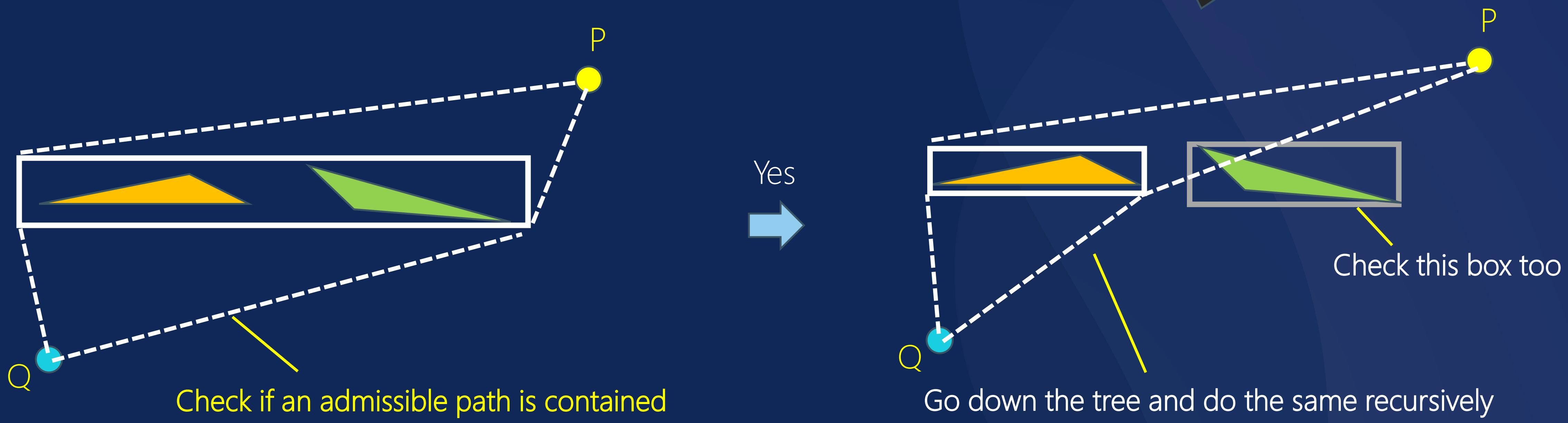


Stage 2. Newton-Raphson iterations

[STAGE1]FINDING ADMISSIBLE TUPLES

- Brute force search works, but prohibitively expensive
- Search using a bounding volume hierarchy for mesh

This can be seen as a cut of non-admissible path space.
➡ Path Cuts



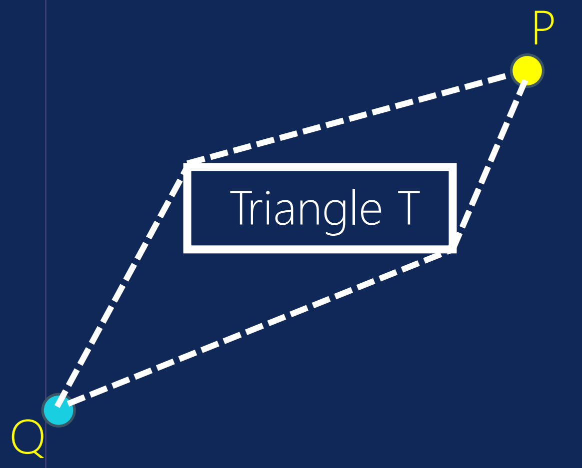
ADMISSIBLE CHECK

- Treat AABB as an interval of a point -> interval arithmetic is a good-fit tool
  - Use interval arithmetic for all values and vectors
  - Incident/outgoing direction, half vector, normal can be expressed with "interval"
- The cost function can be expressed with interval

```
struct intr
{
    float lower;
    float upper;
};
```

Keep track of minimum and maximum of the value.

@ykozw88
https://speakerdeck.com/ykozw/ahuinyan-suan
@Shocker_0x15
https://qiita.com/shocker-0x15/items/f2d7f6135c1bbfa16859



The cost function to interval arithmetic

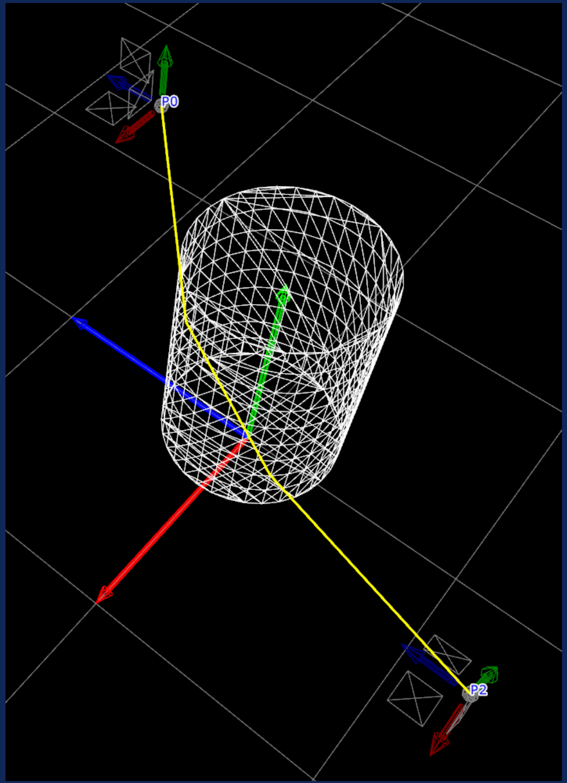
```
intr3 wi = normalize( P - make_intr3( Triangle T ) );
intr3 wo = normalize( Q - make_intr3( Triangle T ) );
intr3 ht = normalize( - eta_i * wi - eta_o * wo );
if ( zeroIncluded( normal - ht ) )
{
    // may contain admissible path
}
```

$$C_t = \hat{\boldsymbol{n}} - \frac{h_t}{|h_t|}$$

The interval of the cost function

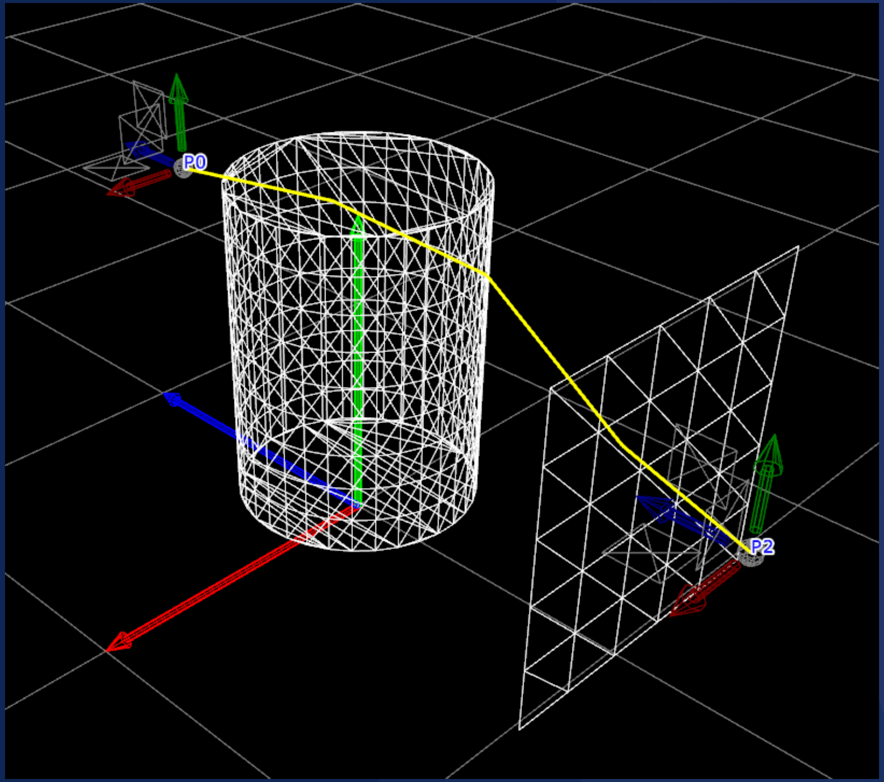
BAD NEWS 😭 😭 😭

TT (2 refractions)



1,368 tuples...

TTT (3 refractions)

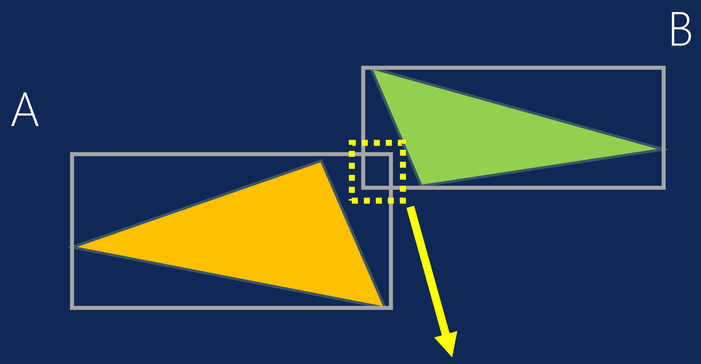


146,598 tuples...

BAD NEWS 😭 😭 😭

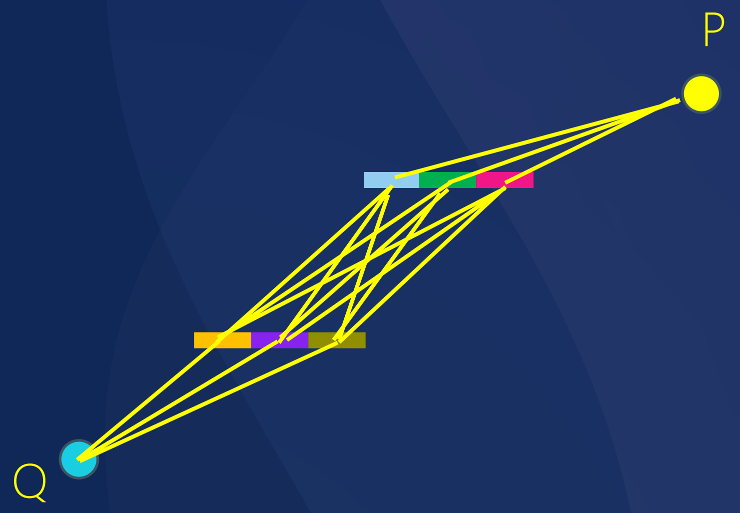
- The algorithm sounds reasonable; however, unfortunately **VERY EXPENSIVE**.
  - Too loose interval bounds
  - Combination explosion

An Evil Example:
overlapped bounds



A – B vector can be ANY DIRECTION!!!

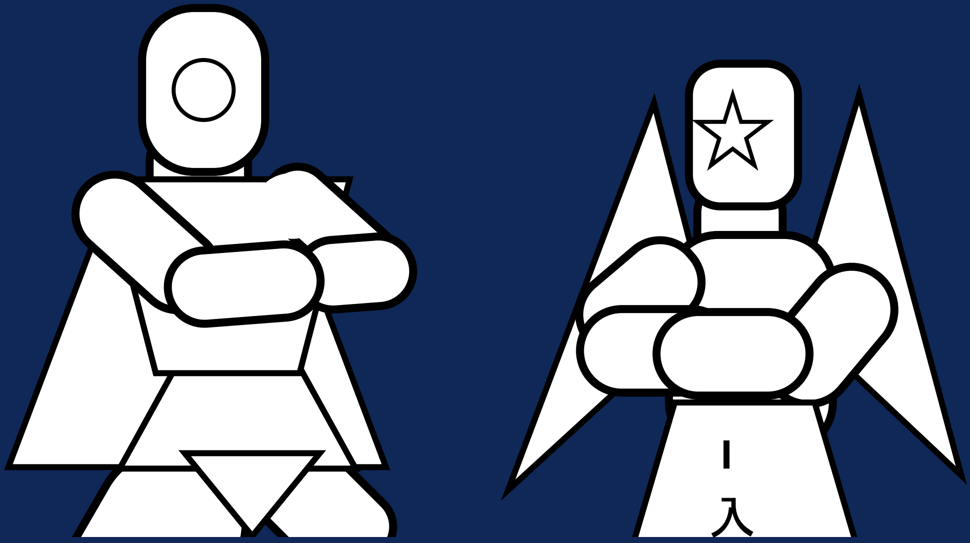
➡ The admissible check is totally useless…



M at the first, N at the second
= MN candidates!!

ANYTHING WE CAN DO?

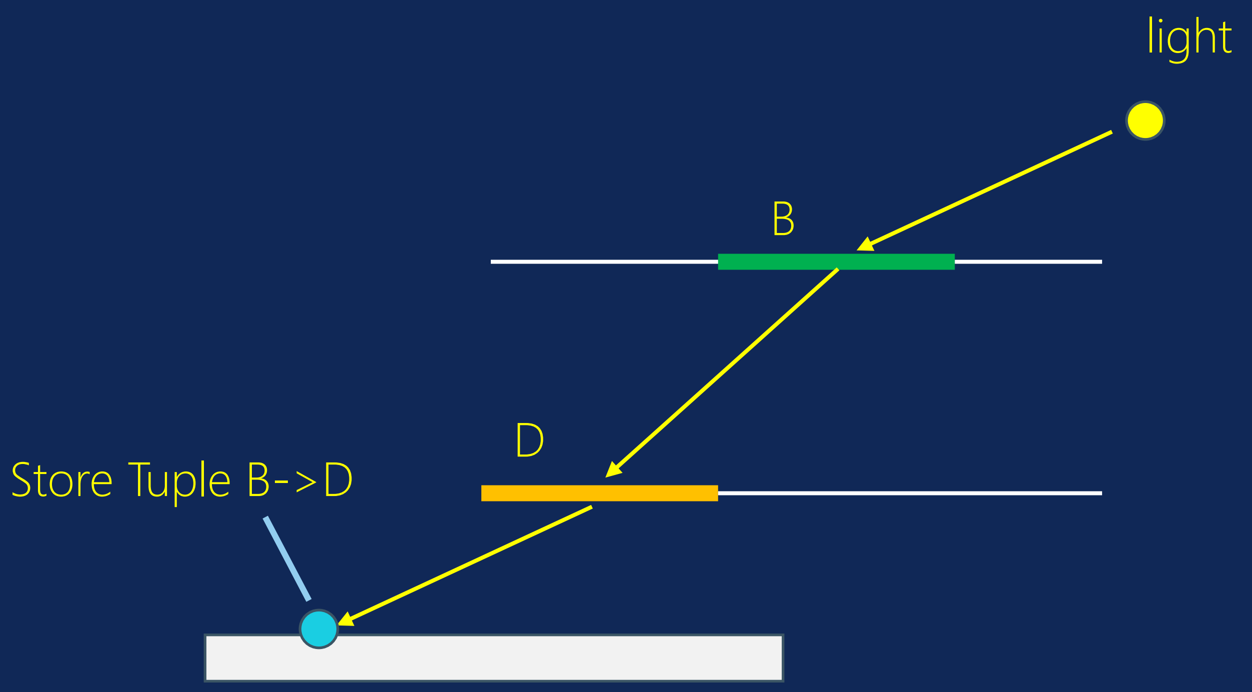
- Specialize some functions interval arithmetic
  - Square, normalize, reflection, etc..
- Affine Arithmetic
  - Combination explosion still there..



良い子の諸君！
「あきらめたらそこで試合終了」という言葉があるが、
さっさと終了しないと次の試合が始まらない事もあるぞ！

ADMISSIBLE TUPLE FINDING - PHOTON TRACING

- Trace photons but store a path "tuples" instead of contributions
  - Look up them during eye path ray tracing similarly to photon mapping



Pros

- Simple and Fast!
- Less false-positive tuples
- Can be used as a good initial path for Newton-Raphson

Cons (open problems)

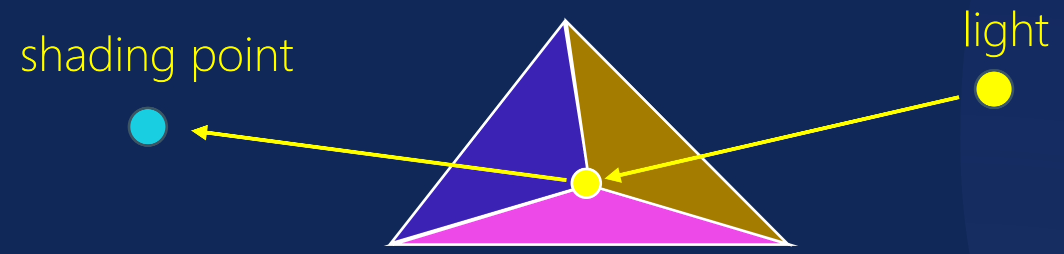
- Some tuples may be missed
  - Need infinite photons to enumerate all of admissible tuples
- Difficulty to support volume
  - Expensive to store tuples along rays

[STAGE2]ROOT FINDING

- Assumes there is only 1 solution for each tuple for simplicity
  - Assumes the scene geometry is subdivided enough
  - The resolution dependency is a weak point of primitive search algorithms
  - We can use subdivision but appropriate depth are not clear

THE COST FUNCTION ON PATH CUTS

- Path vertex can be parameterized by a barycentric coordinate



α_0, β_0

Find the value where the cost to be zero

2 parameters for each vertex

Cost function for Refraction

$$C_t = \hat{\boldsymbol{n}} - \frac{h_t}{|h_t|}$$

where $h_t = -(\eta_i \omega_i + \eta_o \omega_o)$

Cost function for Reflection

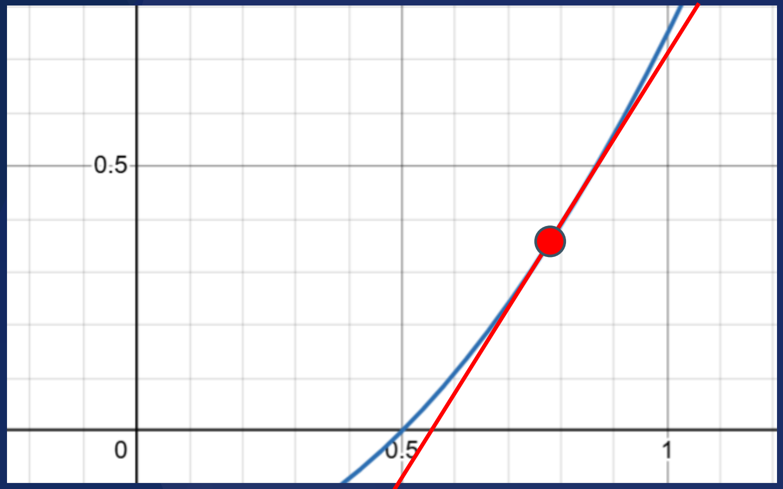
$$C_r = \hat{\boldsymbol{n}} - \frac{h_r}{|h_r|}$$

where $h_r = \dfrac{\omega_i + \omega_o}{|\omega_i + \omega_o|}$

NEWTON-RAPHSON

derivatives

- Let's think the local relationship between α_0, β_0 and C like 1d Newton's method



1st dim $\Delta C_0 = \frac{\partial C_0}{\partial \alpha_0}\Delta\alpha_0 + \frac{\partial C_0}{\partial \beta_0}\Delta\beta_0$

2st dim $\Delta C_1 = \frac{\partial C_1}{\partial \alpha_0}\Delta\alpha_0 + \frac{\partial C_1}{\partial \beta_0}\Delta\beta_0$

3st dim $\Delta C_2 = \frac{\partial C_2}{\partial \alpha_0}\Delta\alpha_0 + \frac{\partial C_2}{\partial \beta_0}\Delta\beta_0$

$$\begin{bmatrix}\Delta C_0\\ \Delta C_1\\ \Delta C_2\end{bmatrix} = \boxed{\begin{bmatrix}\frac{\partial C_0}{\partial \alpha_0} & \frac{\partial C_0}{\partial \beta_0}\\ \frac{\partial C_1}{\partial \alpha_0} & \frac{\partial C_1}{\partial \beta_0}\\ \frac{\partial C_2}{\partial \alpha_0} & \frac{\partial C_2}{\partial \beta_0}\end{bmatrix}}\begin{bmatrix}\Delta\alpha_0\\ \Delta\beta_0\end{bmatrix}$$

Note that the derivatives can be obtained by forward auto diff

NEWTON-RAPHSON

$$\begin{bmatrix} \Delta C_0 \\ \Delta C_1 \\ \Delta C_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial C_0}{\partial \alpha_0} & \frac{\partial C_0}{\partial \beta_0} \\ \frac{\partial C_1}{\partial \alpha_0} & \frac{\partial C_1}{\partial \beta_0} \\ \frac{\partial C_2}{\partial \alpha_0} & \frac{\partial C_2}{\partial \beta_0} \end{bmatrix} \begin{bmatrix} \Delta \alpha_0 \\ \Delta \beta_0 \end{bmatrix}$$

What we want to know

$$\begin{bmatrix} \frac{\partial C_0}{\partial \alpha_0} & \frac{\partial C_0}{\partial \beta_0} \\ \frac{\partial C_1}{\partial \alpha_0} & \frac{\partial C_1}{\partial \beta_0} \\ \frac{\partial C_2}{\partial \alpha_0} & \frac{\partial C_2}{\partial \beta_0} \end{bmatrix}^{-1} \begin{bmatrix} \Delta C_0 \\ \Delta C_1 \\ \Delta C_2 \end{bmatrix} = \begin{bmatrix} \Delta \alpha_0 \\ \Delta \beta_0 \end{bmatrix}$$

Is it solvable???

The inverse of a matrix is only defined for square matrices

$$\begin{bmatrix} \alpha_0^{x+1} \\ \beta_0^{x+1} \end{bmatrix} = \begin{bmatrix} \alpha_0^{x} \\ \beta_0^{x} \end{bmatrix} - \boxed{\begin{bmatrix} \frac{\partial C_0}{\partial \alpha_0} & \frac{\partial C_0}{\partial \beta_0} \\ \frac{\partial C_1}{\partial \alpha_0} & \frac{\partial C_1}{\partial \beta_0} \\ \frac{\partial C_2}{\partial \alpha_0} & \frac{\partial C_2}{\partial \beta_0} \end{bmatrix}^{-1}} \begin{bmatrix} C_0 \\ C_1 \\ C_2 \end{bmatrix}$$

Newton-Raphson step

THE PROBLEM

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$$

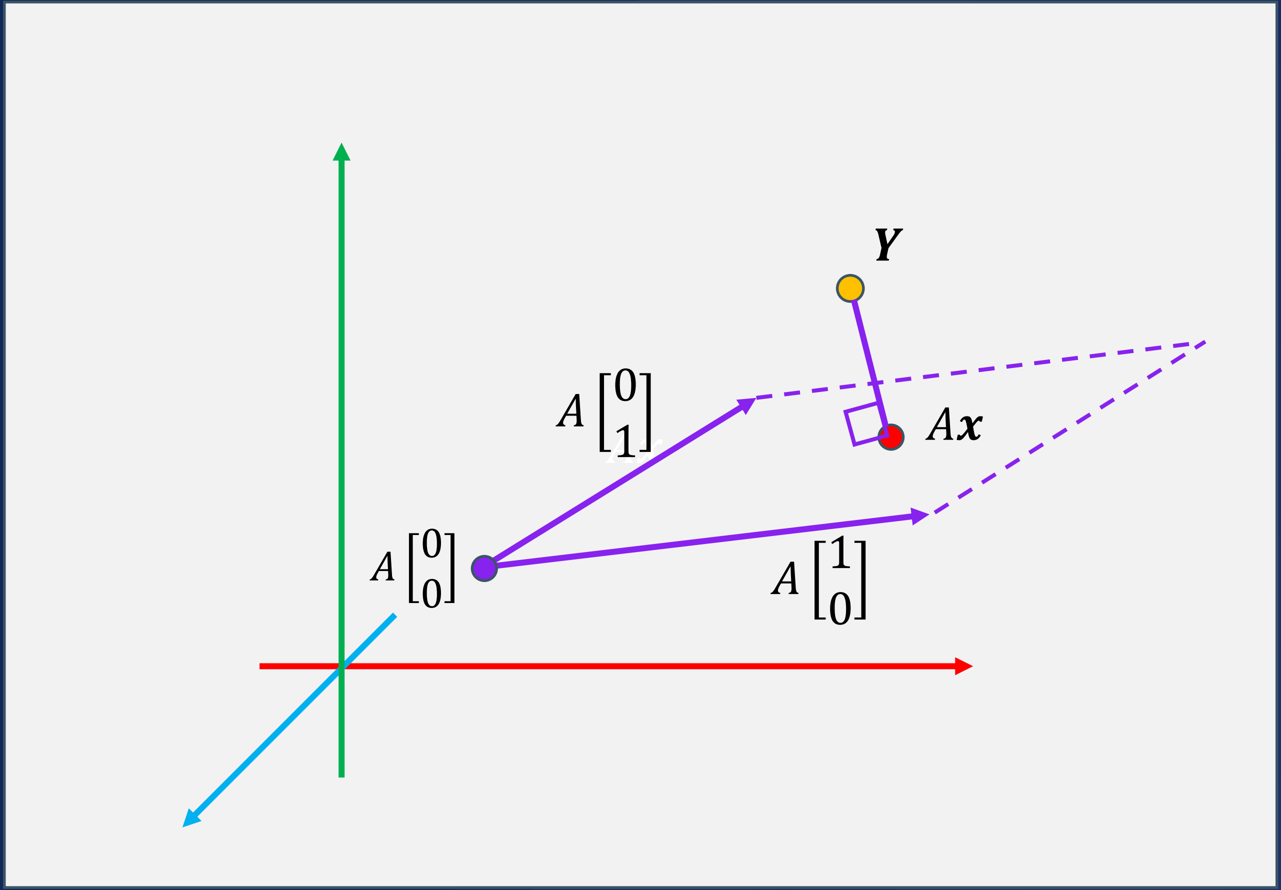
$\boldsymbol{y} = A\boldsymbol{x}$ —— We want this x

Observation:

Rarely $\boldsymbol{y} = A\boldsymbol{x}$ is satisfied… ☹

> called overdetermined system

However, the closest point can be defined



The Condition:

$$(\boldsymbol{y} - A\boldsymbol{x}) \cdot A\begin{bmatrix}1\\0\end{bmatrix} = 0 \qquad \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}^T (\boldsymbol{y} - A\boldsymbol{x}) = 0$$

$$(\boldsymbol{y} - A\boldsymbol{x}) \cdot A\begin{bmatrix}0\\1\end{bmatrix} = 0 \qquad \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}^T (\boldsymbol{y} - A\boldsymbol{x}) = 0$$

$$\Leftrightarrow \quad A^T(\boldsymbol{y} - A\boldsymbol{x}) = 0 \quad \Leftrightarrow \quad \boxed{\underline{A^T A}\boldsymbol{x} = A^T\boldsymbol{y}}$$

Normal Equation

Square matrix

SOLVING NORMAL EQUATION

- $x = (A^T A)^{-1} A^T y$ is a straight-forward way
  - Path Cuts paper uses this approach
  - Requires inverse of large matrix ☹ (6x6 for 3 bounces)
  - Calculation of $A^T A$ is generally not recommended
    - Numerically unstable (increase of conditional number)
- QR decomposition can be used
  - Good balance of stability and cost
  - Used in Pytorch
- Alternatives
  - SVD (Jacobi SVD)
  - Cholesky decomposition (need $A^T A$)

v2.9.0 (stable) Home Install PyTorch User Guide Reference API Developer Notes Community Tutorials Search the docs ...

torch.linalg.eigvalsh
torch.linalg.svd
torch.linalg.svdvals
torch.linalg.solve
torch.linalg.solve_triangular
torch.linalg.lu_solve
torch.linalg.lstsq
torch.linalg.inv
torch.linalg.pinv
torch.linalg.matrix_exp

torch.linalg.lstsq

```
torch.linalg.lstsq(A, B, rcond=None, *, driver=None)
```

Computes a solution to the least squares problem of a system of linear equations.

Letting $\mathbb{K}$ be $\mathbb{R}$ or $\mathbb{C}$, the **least squares problem** for a linear system $AX = B$ with $A \in \mathbb{K}^{m \times n}, B \in \mathbb{K}^{m \times k}$ is defined as

$$\min_{X \in \mathbb{K}^{n \times k}} \|AX - B\|_F$$

where $\|-\|_F$ denotes the Frobenius norm.

- If A is well-conditioned (its condition number is not too large), or you do not mind some precision loss.
  - For a general matrix: *'gelsy'* (QR with pivoting) (default)
  - If A is full-rank: *'gels'* (QR)

For CUDA input, the only valid driver is *'gels'*, which assumes that A is full-rank.

SOLVING NORMAL EQUATION VIA QR

- Decompose matrix $\boldsymbol{A}$ into Orthogonal matrix $\boldsymbol{Q}$ and upper triangular matrix $\mathbf{R}$
  - I used householder QR decomposition

Orthogonal

$$\boldsymbol{A} = \boldsymbol{QR} = \boxed{\begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix}} \begin{bmatrix} r_{11} & r_{12} \\ 0 & r_{22} \\ 0 & 0 \end{bmatrix}$$

Normal Equation

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{y}$$

$$(\boldsymbol{QR})^T(\boldsymbol{QR})\boldsymbol{x} = (\boldsymbol{QR})^T\boldsymbol{y}$$

$$\dots$$

$$\boldsymbol{Rx} = \boldsymbol{Q}^T\boldsymbol{y}$$

$$\boldsymbol{Rx} = \boldsymbol{y}'$$

Very easy to solve ☺

$$\boxed{\begin{bmatrix} r_{11} & r_{12} \\ 0 & r_{22} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} y_0{}' \\ y_1{}' \\ y_2{}' \end{bmatrix}}$$

$$r_{22}x_1 = y_1{}'$$

$$\boxed{x_1 = \frac{y_1'}{r_{22}}}$$

$$r_{11}x_0 + r_{12}x_1 = y_0{}'$$

$$\boxed{x_0 = \frac{y_0' - r_{12}x_1}{r_{11}}}$$

BAD NEWS 😭 😭 😭

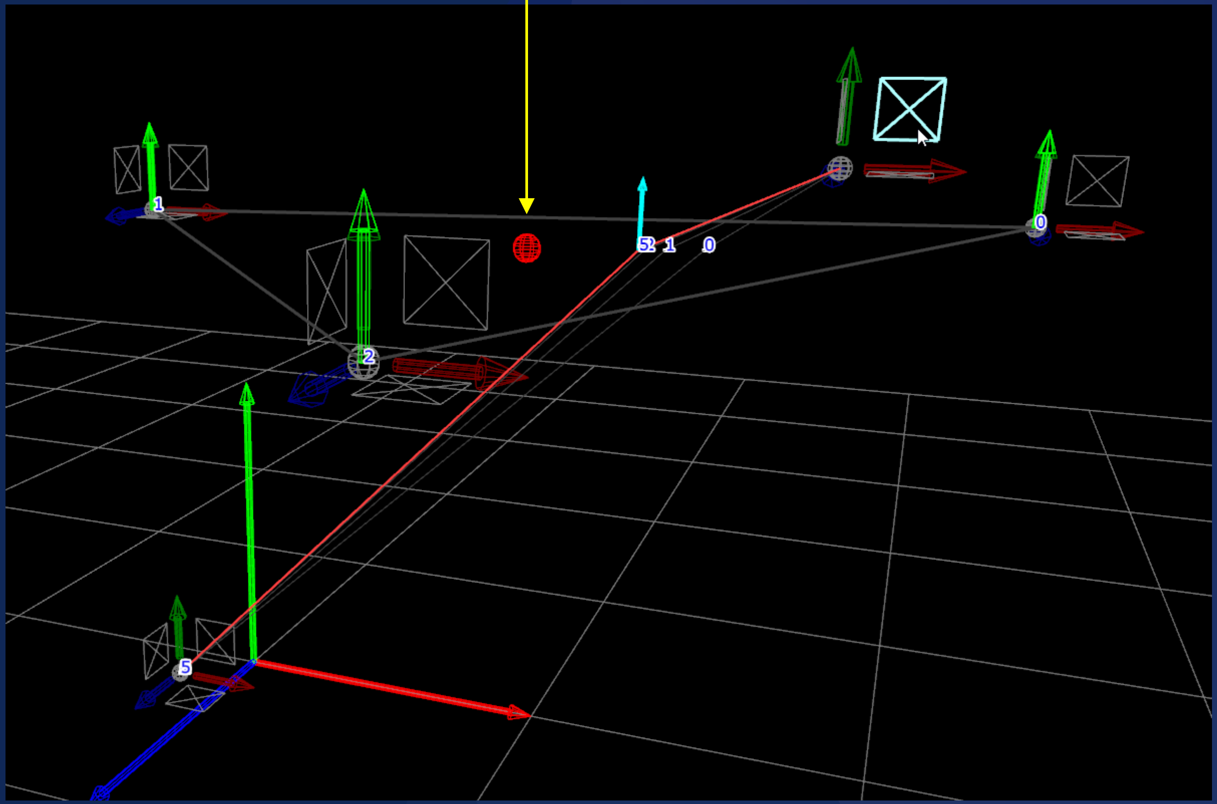
- Really Unstable!!!!!!
  - It's not too bad if the initial condition is good to be fair
- Some techniques are proposed

$\textsc{WalkManifold}(\mathbf{x}_1, \ldots, \mathbf{x}_n \rightsquigarrow \mathbf{x}'_n)$

1 Set $i = 0$ and $\beta = 1$
2 **while** $\|\mathbf{x}_n - \mathbf{x}'_n\| > \varepsilon L$
3 $\quad \mathbf{p} = \mathbf{x}_2 - \boxed{\beta}\, T(\mathbf{x}_2) P_2 A^{-1} B_k T(\mathbf{x}_n)^T (\mathbf{x}'_n - \mathbf{x}_n)$
4 $\quad$ Propagate the ray $\mathbf{x}_1 \to \mathbf{p}$ through all specular interactions, producing $\mathbf{x}_2^+, \ldots, \mathbf{x}_n^+$.
5 $\quad$ **if** step 4 succeeded and $\|\mathbf{x}_n^+ - \mathbf{x}'_n\| < \|\mathbf{x}_n - \mathbf{x}'_n\|$
6 $\quad\quad \mathbf{x}_2, \ldots, \mathbf{x}_n = \mathbf{x}_2^+, \ldots, \mathbf{x}_n^+$
7 $\quad\quad \boxed{\beta = \min\{1, 2\beta\}}$
8 $\quad$ **else**
9 $\quad\quad \boxed{\beta = \frac{1}{2}\beta}$
10 $\quad$ Set $i = i + 1$, and fail if $i > N$.
11 **return** $\mathbf{x}_2, \ldots, \mathbf{x}_{n-1}$

Manifold Exploration

The Initial condition



Angle based

$$\mathbf{c} = \begin{pmatrix} \theta(\mathrm{S}(\boldsymbol{\omega}_\mathrm{i})) - \theta(\boldsymbol{\omega}_\mathrm{o}) \\ \phi(\mathrm{S}(\boldsymbol{\omega}_\mathrm{i})) - \phi(\boldsymbol{\omega}_\mathrm{o}) \end{pmatrix}$$

Specular Manifold Sampling

They are not game changer..

CATCH

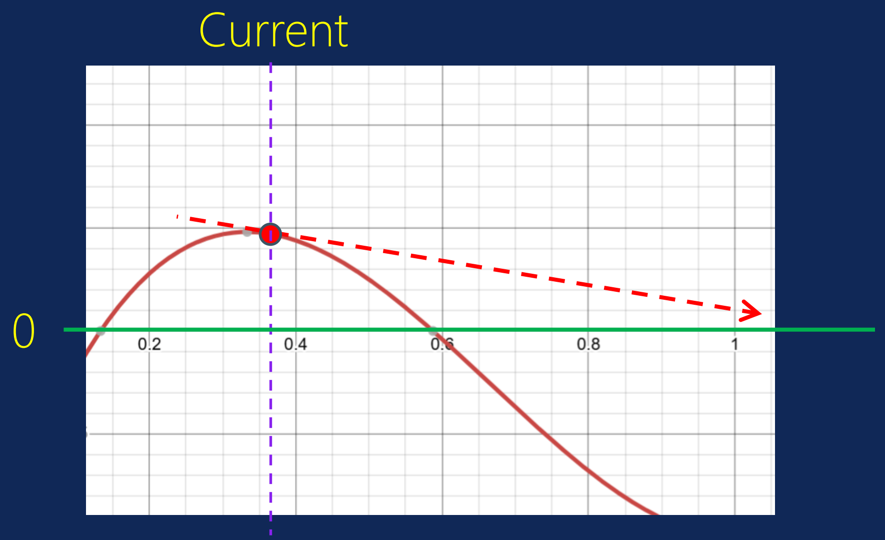
$$\begin{bmatrix}\Delta C_0\\ \Delta C_1\\ \Delta C_2\end{bmatrix} = \begin{bmatrix}\frac{\partial C_0}{\partial \alpha_0} & \frac{\partial C_0}{\partial \beta_0}\\ \frac{\partial C_1}{\partial \alpha_0} & \frac{\partial C_1}{\partial \beta_0}\\ \frac{\partial C_2}{\partial \alpha_0} & \frac{\partial C_2}{\partial \beta_0}\end{bmatrix}\begin{bmatrix}\Delta \alpha_0\\ \Delta \beta_0\end{bmatrix}$$

- Since we use the local relationship between parameters and the cost function
- The strong nonlinearity looks SUS
  - More specifically, normalizations of vectors

Cost function for Refraction in Path Cuts

$$C_t = \hat{\boldsymbol{n}} - \boxed{\frac{h_t}{|h_t|}}$$

where $h_t = -(\eta_i \omega_i + \eta_o \omega_o)$

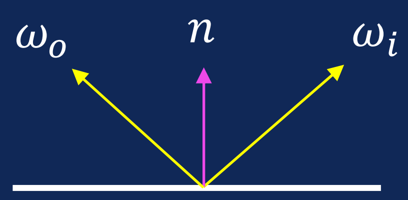


NORMALIZATION FREE FORM

- reflection

$$\boxed{\omega_r = n\frac{2\omega_i \cdot n}{n \cdot n} - \omega_i}$$

✓ normalization free



- refraction

$$\eta = \frac{\eta_o}{\eta_i}, \widehat{\omega}_i = \frac{\omega_i}{|\omega_i|}, \hat{n} = \frac{n}{|n|}$$

$$\omega_{t,PBRT} = -\frac{\widehat{\omega}_i}{\eta} + \left(\frac{\cos\theta_i}{\eta} - \cos\theta_t\right)\hat{n}$$

Required normalize vectors…☹

https://www.pbr-book.org/4ed/Reflection_Models/Specular_Reflection_and_Transmission#eq:refracted-direction

$\omega_{t,PBRT}, \omega_i, n$ are not normalized

MAKE IT NORMALIZATION FREE

$$\eta = \frac{\eta_o}{\eta_i}, \widehat{\omega}_i = \frac{\omega_i}{|\omega_i|}, \hat{n} = \frac{n}{|n|}$$

$$\omega_{t,PBRT} = -\frac{\widehat{\omega}_i}{\eta} + \left(\frac{\cos\theta_i}{\eta} - \cos\theta_t\right)\hat{n}$$

$$|\omega_i||n|^2\eta\omega_{t,PBRT} = |\omega_i||n|^2\eta\left\{-\frac{\widehat{\omega}_i}{\eta} + \left(\frac{\cos\theta_i}{\eta} - \cos\theta_t\right)\hat{n}\right\}$$

$$\boxed{\omega_o = -(n \cdot n)\omega_i + \left((\omega_i \cdot n) - \sqrt{(\omega_i \cdot \omega_i)(n \cdot n)(\eta^2 - 1) + (\omega_i \cdot n)^2}\right)n}$$

✓ normalization free

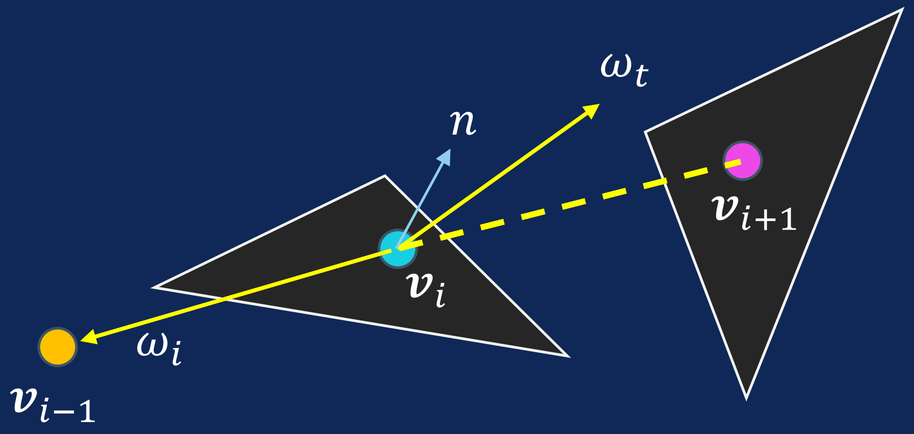
NEW COST FUNCTION FOR REFRACTION

$$\omega_i = \boldsymbol{v}_{i-1} - \boldsymbol{v}_i$$

$$\omega_t = -(n \cdot n)\omega_i + \left((\omega_i \cdot n) - \sqrt{(\omega_i \cdot \omega_i)(n \cdot n)(\eta^2 - 1) + (\omega_i \cdot n)^2}\right) n$$

$$C_{t,new} = \omega_t \underline{\times} (\boldsymbol{v}_{i+1} - \boldsymbol{v}_i)$$

Cross product can check directional equality without normalization ☺

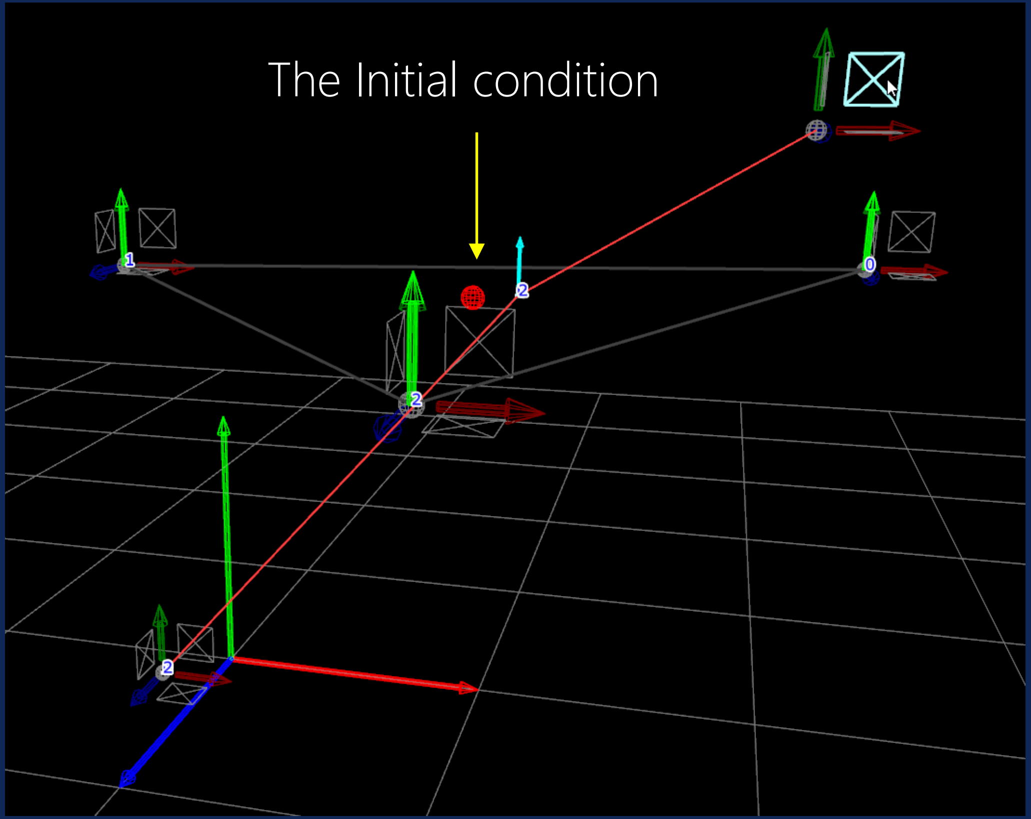


Current Triangle

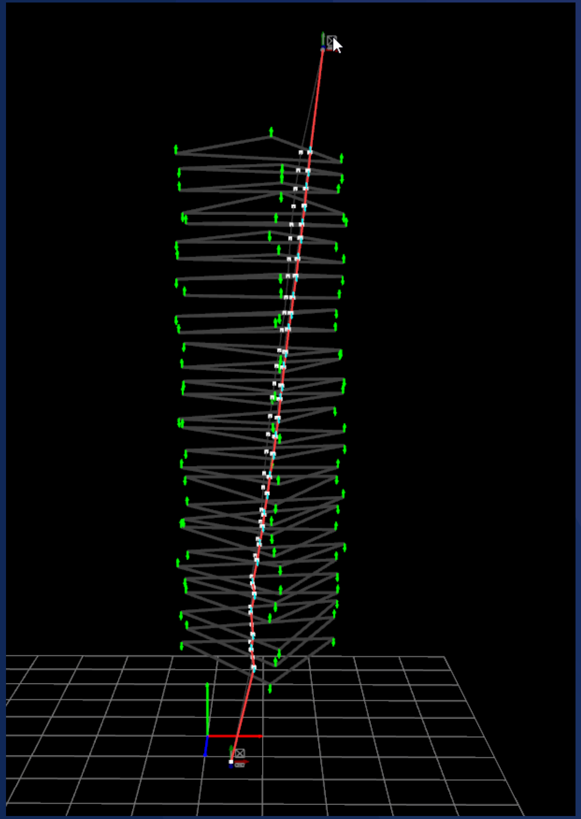
Next Triangle

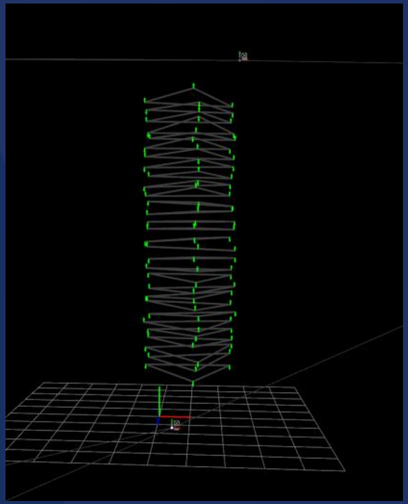
REALLY STABLE AND QUICK TO CONVERGE!!!

T



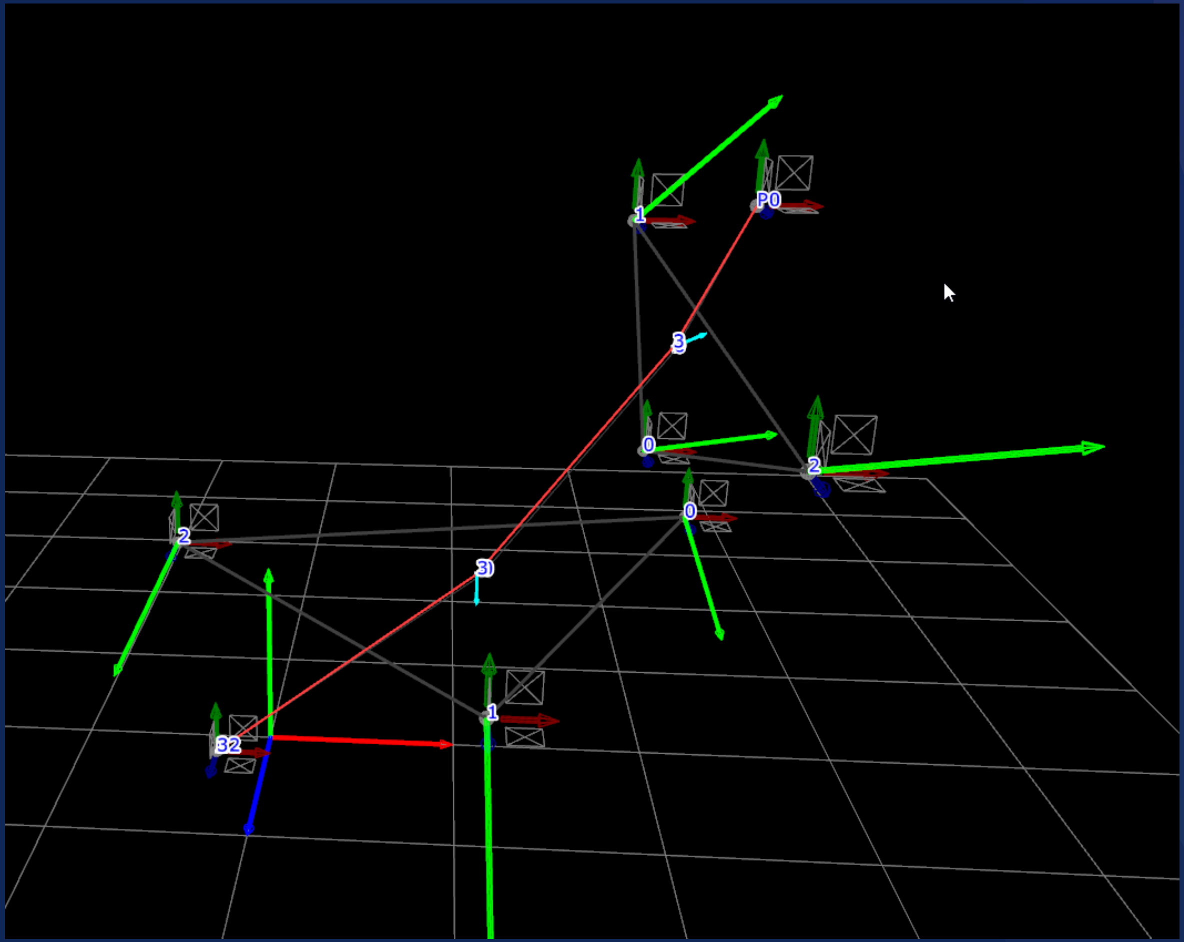
TTTTTTTTTTTTTTTTTTTTTTTTTTTTTT (30 refractions)



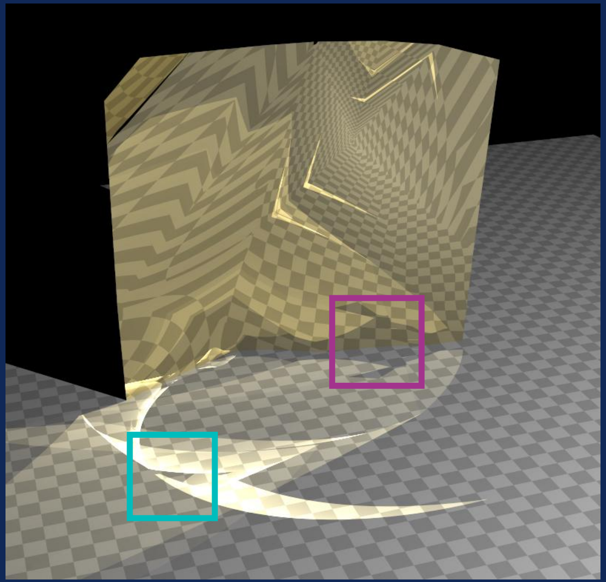


Never converged with cost function on path cuts

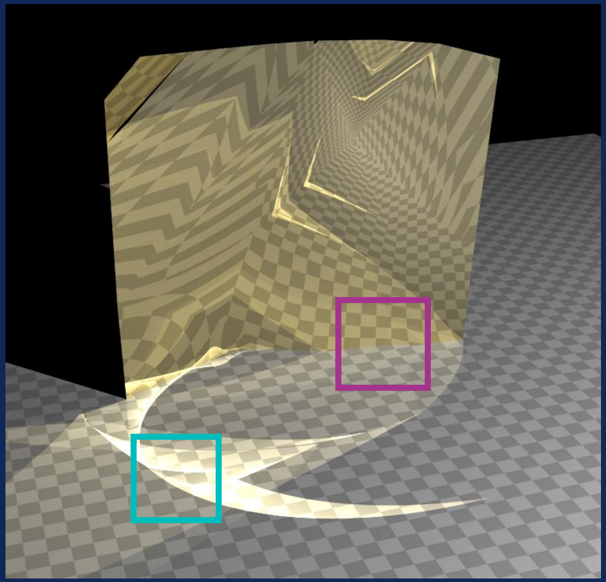
OPEN PROBLEM



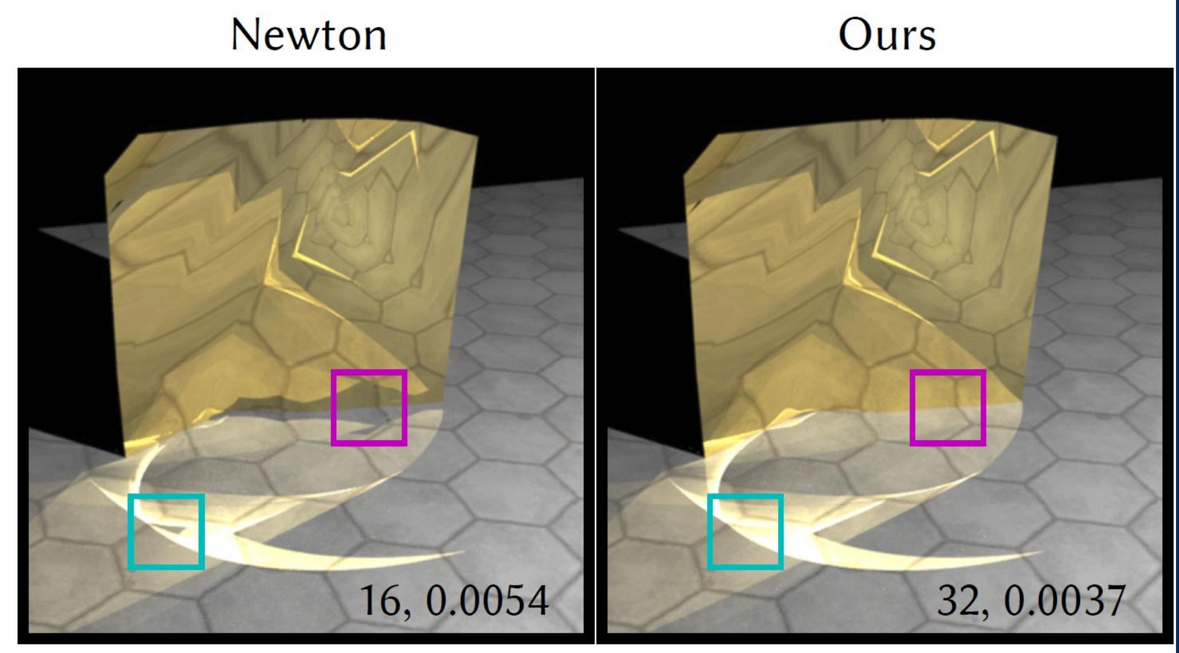
Path Cuts Cost Function



Norm Free Cost Function



mesh/slab1sss.obj



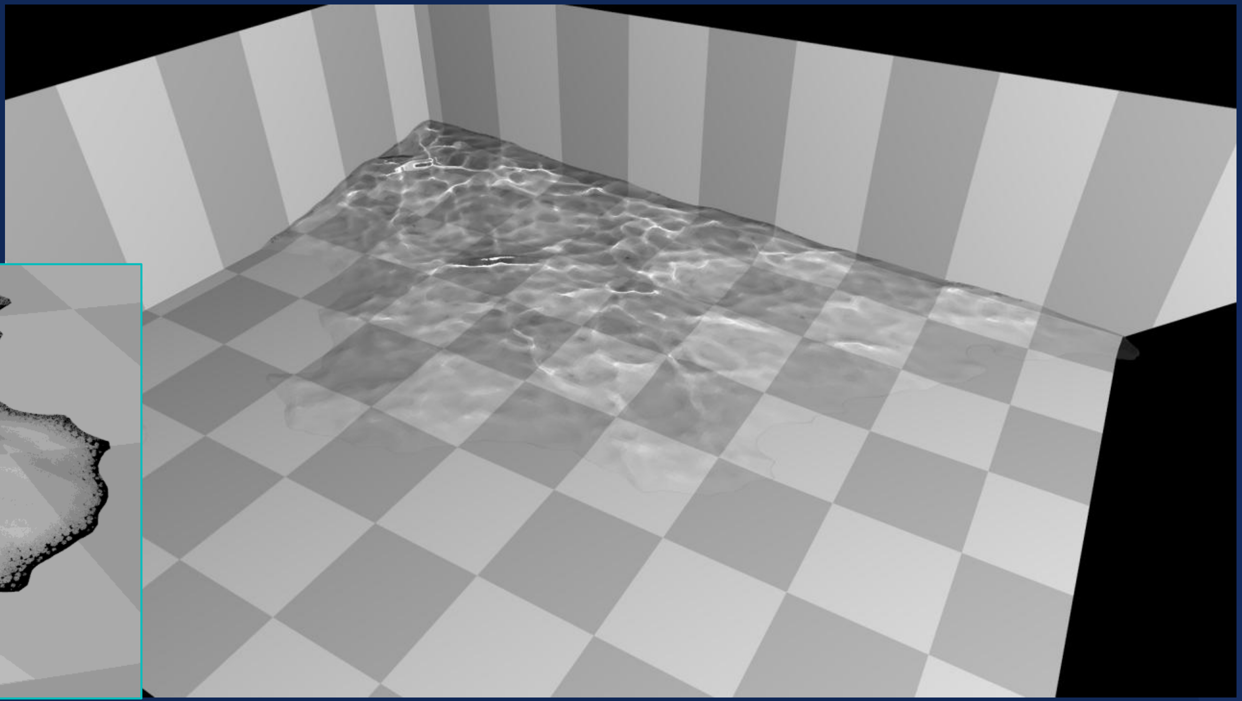
My Implementation

The initial params are $(\alpha, \beta) = (\frac{1}{3}, \frac{1}{3})$

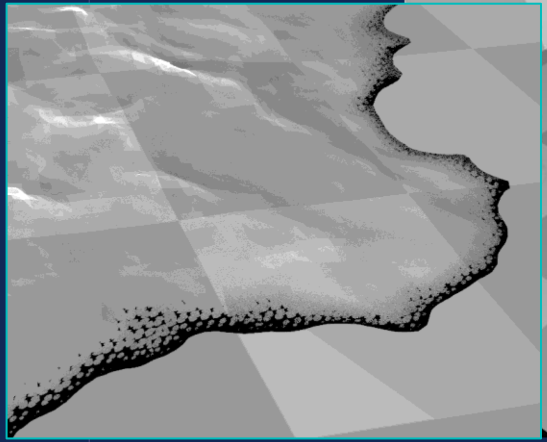
Specular Polynomials

Note that color calc is not accurate on my impl

Norm Free Cost Function



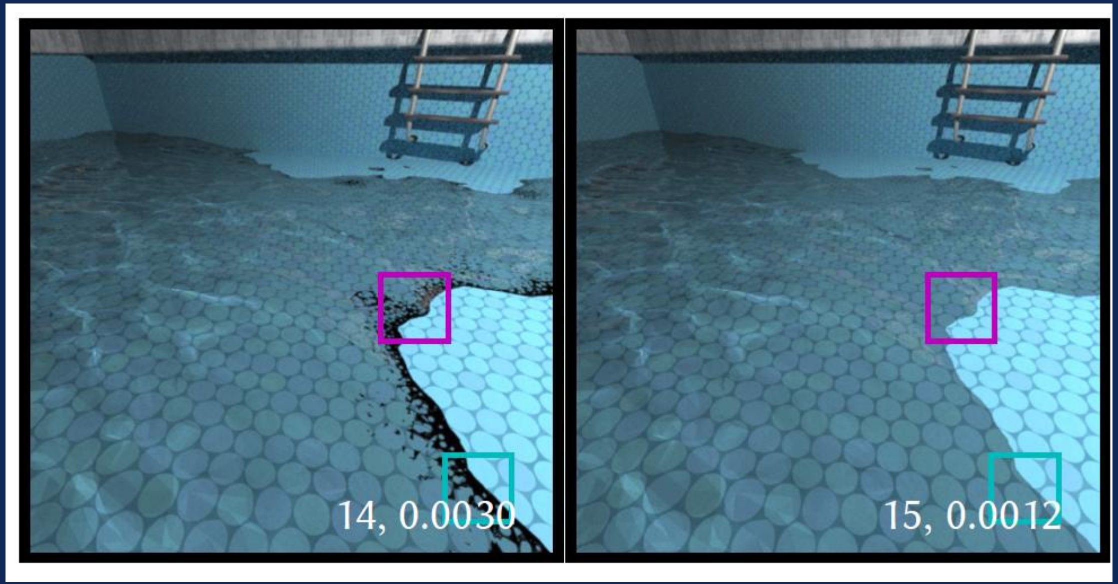
Path Cuts Cost Function



sp_pool/opsr_new.obj
mesh_pool_rotate/pool_inside.obj

My Implementation

The initial params are $(\alpha, \beta) = (\frac{1}{3}, \frac{1}{3})$



Specular Polynomials

SUMMARY

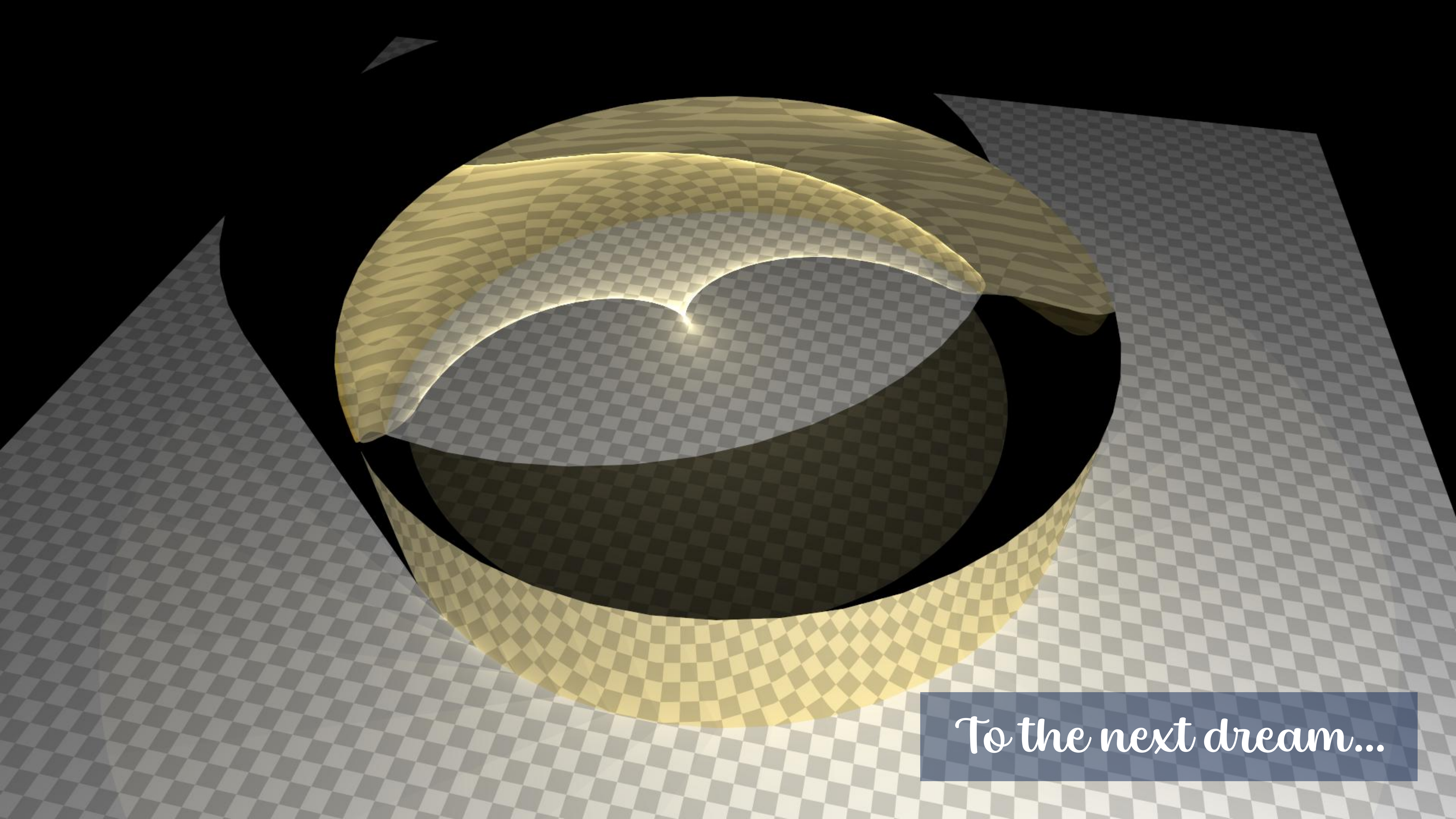
- Introduced Path Cuts
  - Built by basic techniques(interval arithmetic, auto diff, least squares)
  - fully deterministic
- Proposed alternative solutions
  - Photon Tracing for admissible tuples
  - Better convergence with normalization free form
- Open Problems
  - Cons of photon tracing
  - Still difficult path for newton’s method
- Thank you, linear algebra

REFERENCES

- きなこもち, Pathtracing technics for Caustics (https://speakerdeck.com/kinakomoti321/pathtracing-technics-for-caustics-f93e343a-2d66-4d7c-a78f-23a20a7968af)
- きなこもち, Specular Polynomial (https://scrapbox.io/ShaderMemorandom/Specular_Polynomial)
- @ykozw88, Interval Arithmetic, Affine Arithmetic (https://speakerdeck.com/ykozw/ahuinyan-suan)
- @Shocker_0x15, アフィン演算で求める三角形上の関数の範囲 (https://qiita.com/shocker-0x15/items/f2d7f6135c1bbfa16859)

とってもとってもありがとうございます

- Walter, et al. “Single Scattering in Refractive Media with Triangle Mesh Boundaries” [2009]
- Jakob and Marschner, ” Manifold exploration: a Markov Chain Monte Carlo technique for rendering scenes with difficult specular transport” [2012]
- Hanika, et al., ”Manifold Next Event Estimation” [2015]
- Hanika et al., “Specular Manifold Sampling for Rendering High-Frequency Caustics and Glints” [2020]
- Wang, at al, “Path Cuts: Efficient Rendering of Pure Specular Light Transport” [2020]
- Fan, at al, ”Specular Polynomials” [2024]
  - https://github.com/mollnn/spoly
- Fan, at al, “Bernstein Bounds for Caustics” [2025]


To the next dream...

APPENDIX

DIFFUSE SHADING FOR A POINT LIGHT

$$L_o = E\frac{R}{\pi}$$

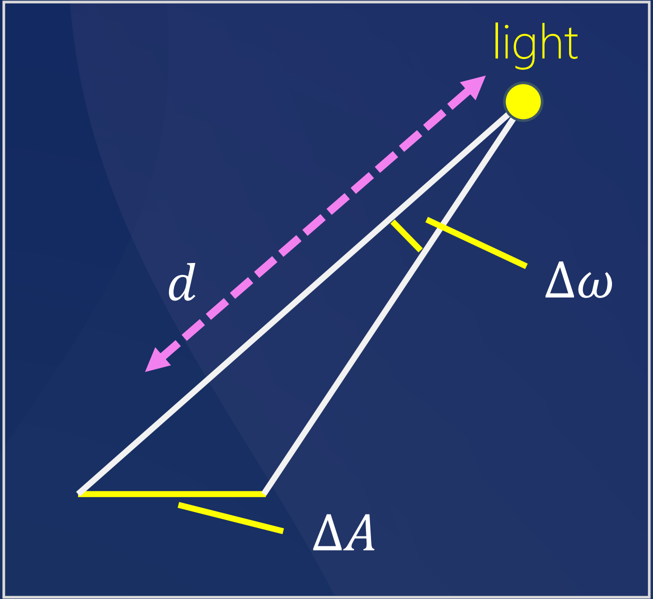
$E = \frac{d\Phi}{dA}$: Irradiance

$\frac{R}{\pi}$: Diffuse BRDF

$$= \frac{d\Phi}{dA}\frac{R}{\pi} = \frac{d\Phi}{d\omega}\frac{\cos\theta}{d^2}\frac{R}{\pi}$$

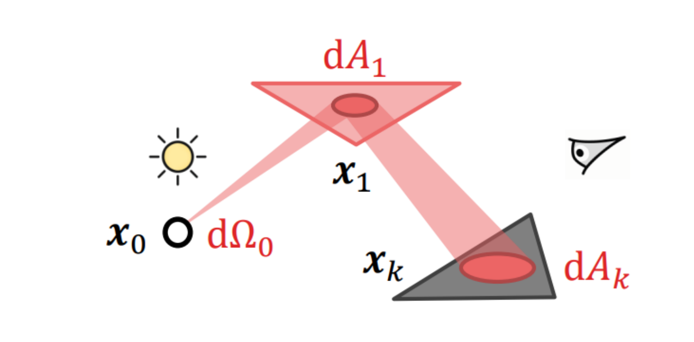
$$= I\underbrace{\frac{\cos\theta}{d^2}}_{\frac{d\omega}{dA}}\frac{R}{\pi}$$

$I = \frac{d\Phi}{d\omega}$: light intensity



$$\Delta A = \frac{d^2}{\cos\theta}\Delta\omega$$

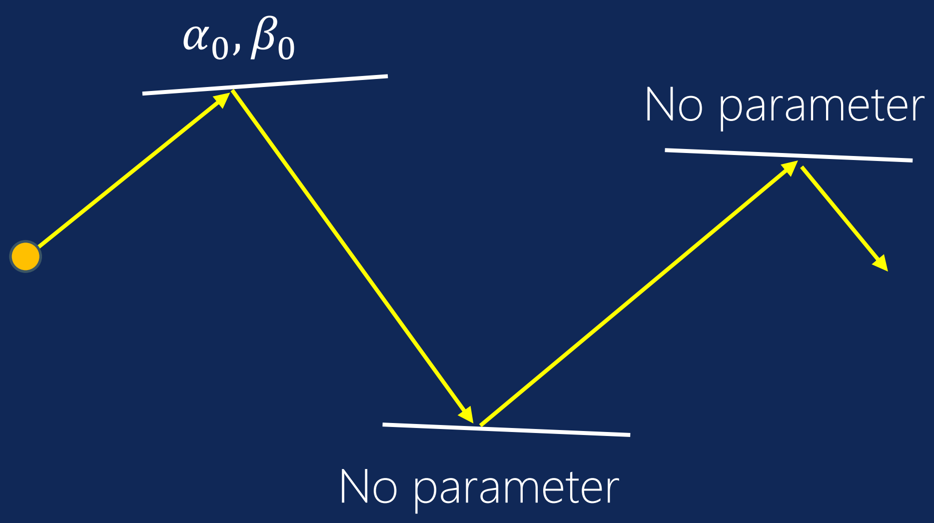
Illustration of the generalized geometric term(GGT)
Bernstein Bounds for Caustics

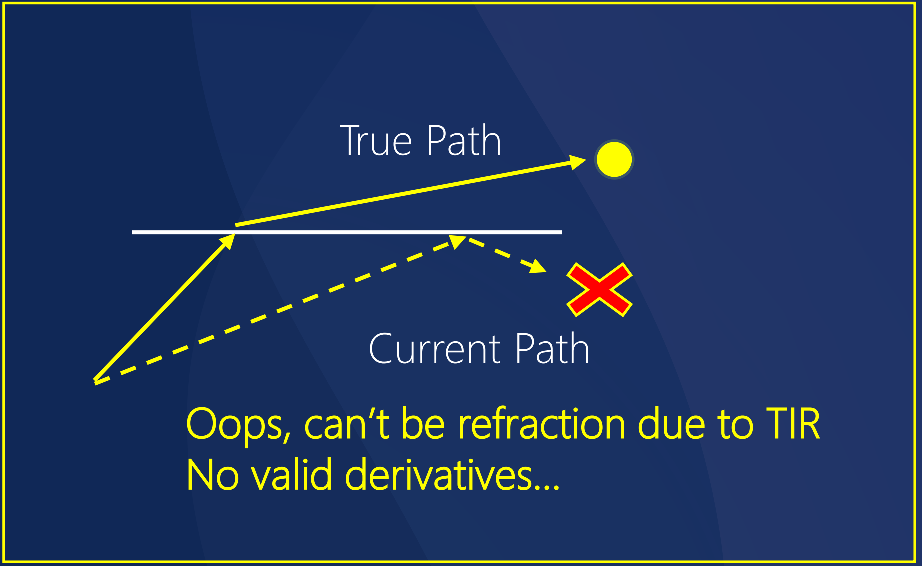


Auto diff or something clever method can be used

OPEN PROBLEM 2

- A specular path is actually defined by the first vertex
  - Variable reduction could be possible like specular polynomials
- However, it can't handle TIR situation…☹





A corner case