Demystifying PyRedner

Differentiable Monte Carlo Ray Tracing through Edge Sampling

TZU-MAO LI, MIT CSAIL
MIIKA AITTALA, MIT CSAIL
FRÉDO DURAND, MIT CSAIL
JAAKKO LEHTINEN, Aalto University

Udit Jain

Motivation behind the Paper.

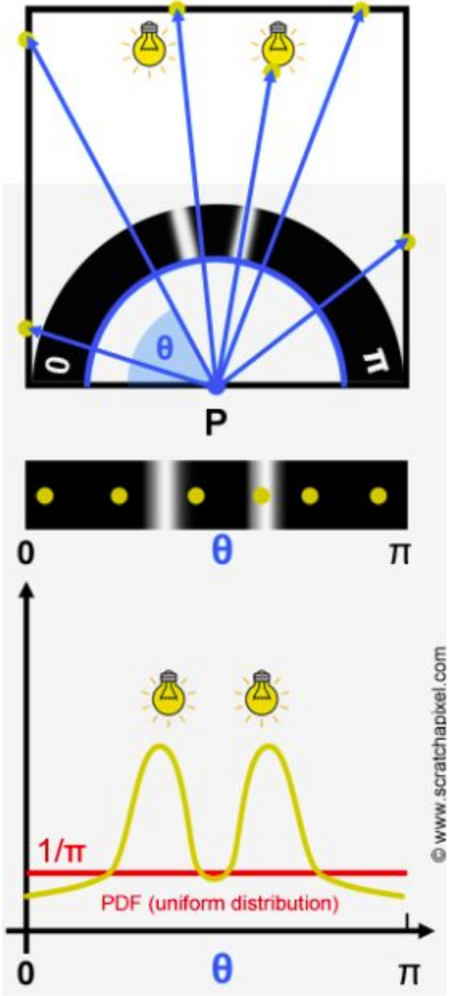
- Derivatives are central to Graphics, Vision and ML. Critical for solution of optimization and inverse problems.
- Need for rendering algorithms that can be differentiated wrt to arbitrary input parameters such as pose, geometry, light, material etc.
- Rendering integral has visibility terms which are not differentiable at object boundaries.

Background

- The pixel integral equation.

$$I(x, x') = g(x, x') \left[\epsilon(x, x') + \int_S \rho(x, x', x'') I(x', x'') dx'' \right].$$

- Previous work is : on fast, approximate solutions using simpler rendering models that only handle primary visibility, and ignore secondary effects
- Analytical solutions exist for interreflection but are difficult to generalize.
- Importance sampling of edges is done for Monte Carlo sampling to get the relevant edges.



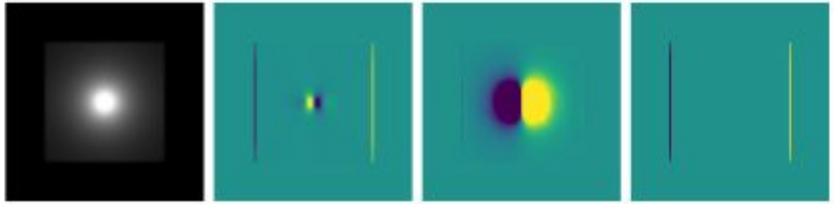
Key Deliverables

- Algorithm to compute derivatives of scalar functions over a rendered image wrt to arbitrary input parameters. Stochastic solution, builds on MC ray tracing. Techniques to explicitly sample edges of triangles in addition to the usual solid angle sampling.

- Integrate it with the autograd library in PyTorch for efficient integration with optimization and learning approaches. Scene geometry, lighting, camera and materials are parameterized by Py-Torch tensors.
- Complex combination of 3D graphics, light transport, and neural networks. Backpropagation runs seamlessly across PyTorch and the renderer.

Advantages over previous works

- Computation of gradients using a hybrid approach that mixes automatic differentiation and manually derived derivatives focusing on discontinuous integrand.
- Handling Primary & Secondary visibility and Global Illumination. (Shadows)
- Estimation of gradient directly by auto diff and edge sampling.
- Taking into account the geometric discontinuities.



(a) planar scene (b) OpenDR (c) Neural (d) ours

Mathematical Challenges

- Rendering integral has visibility terms that are non-differentiable at object boundaries.
- Final image function is differentiable once radiance has been integrated over pixel prefilters, light source areas, etc., the integrand of rendering algorithms is not.
- The derivative of the integrand has Dirac delta terms at occlusion boundaries that cannot be handled by traditional sampling strategies.

(a) area sampling (b) edge sampling

Problem Definition

Given:

- 3D Scene
- Continuous Parameter Set, Φ (including camera pose, scene geometry, material and lightning parameters)
- A scalar function computed from the rendered image.(Eg. Loss)

Goal: Backpropagate the gradient of the scalar with respect to all scene params , Φ

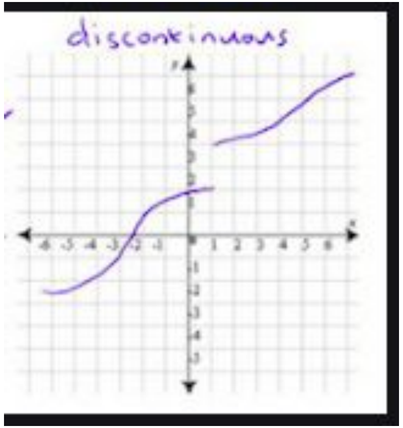
Assumptions:

- Triangular mesh representation.
- No point sources of light.

Procedure

- Generate the image using path-tracing algorithm [Kajiya 1986]
- Use MC (area) sampling to estimate the Intensity integral and its derivative.
- Discontinuous on edges of geometry. Sudden change in I.
- Derivative of Discontinuous integrand is dirac delta function and by traditional sampling it has a zero probability (P=0) of being sampled. Handle in phases

(a) area sampling (b) edge sampling



Phase I - Primary Visibility

$$\nabla I = \nabla \iint f(x, y; \Phi) dxdy.$$

$$\theta(\alpha(x,y)) f_u(x,y) + \theta(-\alpha(x,y)) f_l(x,y),$$

- Pixel Color integral. f(x,y) = k(x,y) L(x,y). K- pixel filter L-radiance.
- **All** discontinuities happen at triangle edges. So we model the equation as Heaviside Step Func and explicitly differentiate.
- **α**(x,y) is the equation of line between the triangles, Fu is the u
- the lower half space (**α**(x,y)<0).
- Bring the gradient inside the integral and use product rule.
- 1st term estimates integral over triangle edges containing dirac
- second estimates the original pixel function f's gradient which i
- diff.
- Dirac delta from first term can be eliminated by integration w
- substitute dxdy by ds
- Denominator is the L2 lenght of graident of the edge equatic
- the measure of length edge

$$\iint f(x,y) dxdy = \sum_i \iint \theta(\alpha_i(x,y)) f_i(x,y) dxdy.$$

$$\nabla \iint \theta(\alpha_i(x,y)) f_i(x,y) dxdy$$
$$= \iint \delta(\alpha_i(x,y)) \nabla \alpha_i(x,y) f_i(x,y) dxdy$$
$$+ \iint \nabla f_i(x,y) \theta(\alpha_i(x,y)) dxdy.$$

$$\iint \delta(\alpha_i(x,y)) \nabla \alpha_i(x,y) f_i(x,y) dxdy$$
$$= \int_{\alpha_i(x,y)=0} \frac{\nabla \alpha_i(x,y)}{\left\| \nabla_{x,y} \alpha_i(x,y) \right\|} f_i(x,y) d\sigma(x,y),$$

Phase I - Primary Visibility

$$I = \iint k(x,y)L(x,y)dxdy.$$

$$\nabla I = \nabla \iint f(x, y; \Phi)dxdy.$$

$$\theta(\alpha(x,y)) f_u(x,y) + \theta(-\alpha(x,y)) f_l(x,y),$$

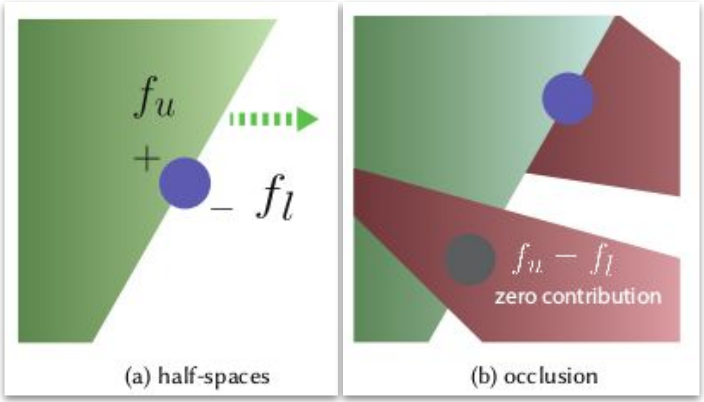
$$\alpha(x,y) = (a_y - b_y)x + (b_x - a_x)y + (a_x b_y - b_x a_y).$$

$$\iint f(x,y)dxdy = \sum_i \iint \theta(\alpha_i(x,y)) f_i(x,y)dxdy.$$

$$\begin{aligned} &\nabla \iint \theta(\alpha_i(x,y)) f_i(x,y)dxdy \\ &= \iint \delta(\alpha_i(x,y)) \nabla \alpha_i(x,y) f_i(x,y)dxdy \\ &+ \iint \nabla f_i(x,y) \theta(\alpha_i(x,y))dxdy. \end{aligned}$$

$$\begin{aligned} &\iint \delta(\alpha_i(x,y)) \nabla \alpha_i(x,y) f_i(x,y)dxdy \\ &= \int_{\alpha_i(x,y)=0} \frac{\nabla \alpha_i(x,y)}{\left\| \nabla_{x,y} \alpha_i(x,y) \right\|} f_i(x,y) d\sigma(x,y), \end{aligned}$$

Phase I - Primary Visibility Cont.



- Using MC sampling to estimate modified first term.
- Compute the values Fu(x,y) and Fl(x,y) on the edge.
- We get MC estimation of first term on a single edge E as:

$$\frac{1}{N}\sum_{j=1}^{N}\frac{\|E\|\nabla\alpha_i(x_j,y_j)(f_u(x_j,y_j)-f_l(x_j,y_j))}{P(E)\left\|\nabla_{x_j,y_j}\alpha_i(x_j,y_j)\right\|},$$

- || E || is the edge length and P(E) is the probability of selecting edge E. Num samples = N
- In practice, for smooth shading, most of triangular edges are in continuous regions and dirac delta function is 0. Because Fu and Fl values are equal at boundary. (b)
- **Only Silhouette Edges have non-zero contributions to gradients.**

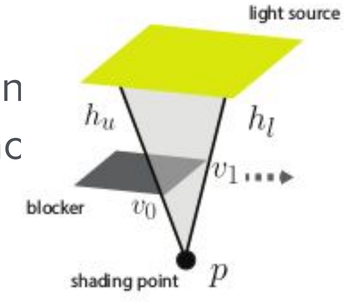
Phase II - Secondary Visibility

$$g(p) = \int_{\mathcal{M}} h(p, m)\, dA(m),$$

$$\theta(\alpha(p, m))h_u(p, m) + \theta(-\alpha(p, m))h_l(p, m).$$

$$\alpha(p, m) = (m - p) \cdot (v_0 - p) \times (v_1 - p).$$

- Generalized method to handle shadow and global illumination.
- Focus on point P, The shading equation is given by g(p).
- A is the area measure of point m, and h is scene function.
- An edge (v0, v1) introduces a ‘step function’ in scene fx, h.
- Derive the edge function **α**(m) by forming a plane using point p, v0 an between (m-p) and plane normal determines the different edges spac



(a) secondary visibility

- Apply the same derivation with p and m as the x and y. The analogous edge integral equation is given. Nm is surface normal.
- Here sigma’ is the length of projection of point on the edge from sampling point p to point m on scene manifold. Extra correction term || Nm x Nh ||, because of projection of scene surface element on edge.

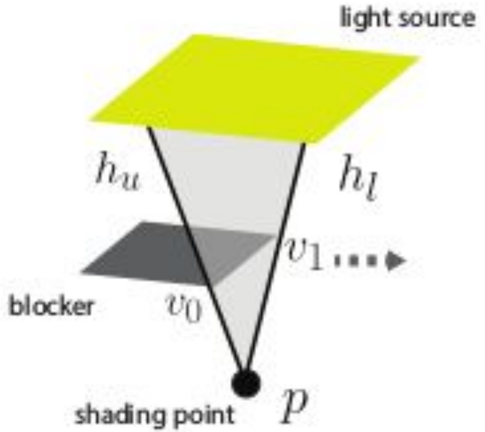
$$\int_{\alpha(p,m)=0} \frac{\nabla \alpha(p,m)}{\|\nabla_m \alpha(p,m)\|} h(p,m) \frac{1}{\|n_m \times n_h\|} d\sigma'(m)$$

$$n_h = \frac{(v_0 - p) \times (v_1 - p)}{\|(v_0 - p) \times (v_1 - p)\|},$$

Phase II - Secondary Visibility

$$g(p) = \int_{\mathcal{M}} h(p, m)\, dA(m),$$

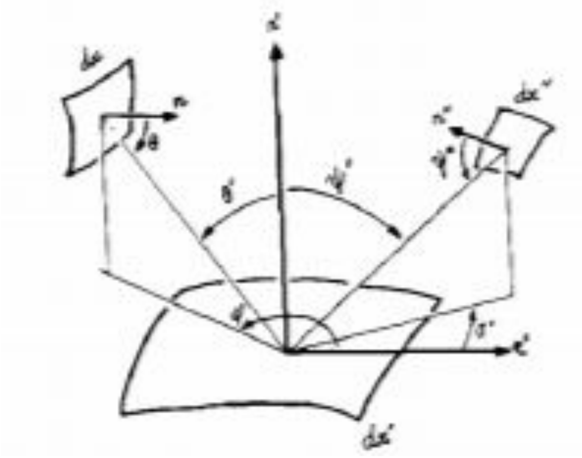
$$\theta(\alpha(p, m))h_u(p, m) + \theta(-\alpha(p, m))h_l(p, m).$$



$$\alpha(p, m) = (m - p) \cdot (v_0 - p) \times (v_1 - p).$$

$$\int_{\alpha(p, m)=0} \frac{\nabla \alpha(p, m)}{\|\nabla_m \alpha(p, m)\|} h(p, m) \frac{1}{\|n_m \times n_h\|} d\sigma'(m)$$

$$n_h = \frac{(v_0 - p) \times (v_1 - p)}{\|(v_0 - p) \times (v_1 - p)\|},$$



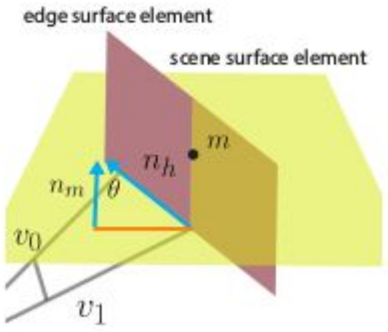
ty

Phase II - Secondary Visibility Cont.

$$\int_0^1 \frac{\nabla\alpha(p, m(t))}{\|\nabla_m \alpha(p, m(t))\|} h(p, m(t)) \frac{\|J_m(t)\|}{\|n_m \times n_h\|} dt,$$

- To integrate the 3D edge integral using MC sampling, substitute the variable again from the point m on the surface to the line parameter t on the edge v0 + t (v1 − v0).:
- the Jacobian Jm(t) is a 3D vector describing the projection of edge (v0, v1) onto the scene manifold with respect to the line parameter.

- Need for a sophisticated data structure to prune the edges with zero contribution. Novel method for **importance sampling of edges**.



(b) width correction

Phase III - Importance Sampling

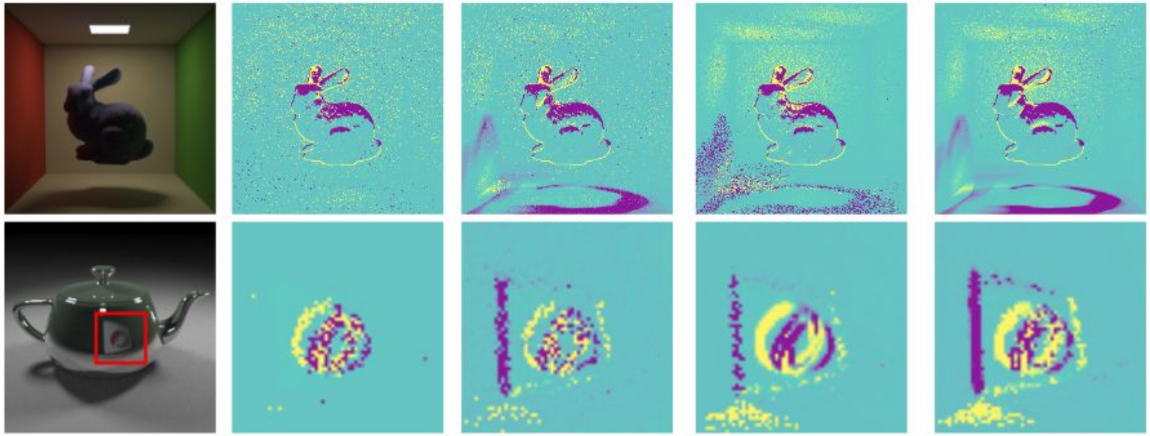
- Sample from millions of triangular edges in the scene.
- Two tasks: Sample an edge and then sample point on the edge efficiently.
- Only tiny fraction contribute to gradients (silhouette edges) and many of them have small solid angle. Native sampling fails to select important edges.
- For secondary visibility, viewpoint can be anywhere and need to take into account the material response between the viewpoint and the point on edge.
- Solution: Hierarchical Edge Sampling: The first contains the triangle edges that associate with only one face and meshes that do not use smooth shading normals. The second contains the remaining edges.
- For first set, build a 3D bounding volume hierarchy using 3D positions of two endpoints of an edge.
- For quick rejection of non-silhouette edges, for each node store a cone direction.

Phase III - Importance Sampling Cont.

- Edges blocking light sources are usually the most significant source of contribution. Traverse the hierarchy twice. The first traversal focuses on edges that overlap with the cone subtended by the light source at the shading point, and the second traversal samples all edges. We combine the two sets of samples using multiple importance sampling [Veach and Guibas 1995]. We use a box-cone intersection to quickly discard the edges that do not intersect the light sources.
- During the traversal, for each node in the hierarchy we compute an importance value for selecting which child to traverse next, based on an upper bound estimation of the contribution, similar to the lightcuts algorithm [Walter et al. 2005]. We estimate the bound using the total length of edges times inverse squared distance times a Blinn-Phong BRDF. Set the importance to zero if the node does not contain any silhouette. We traverse into both children if the shading point is inside both of their bounding boxes, or when the BRDF bound is higher than a certain threshold (for all examples in the paper we set it to 1), or when the angle subtended by the light cone is smaller than a threshold (we set it to $\cos^{-1}(0.95)$).

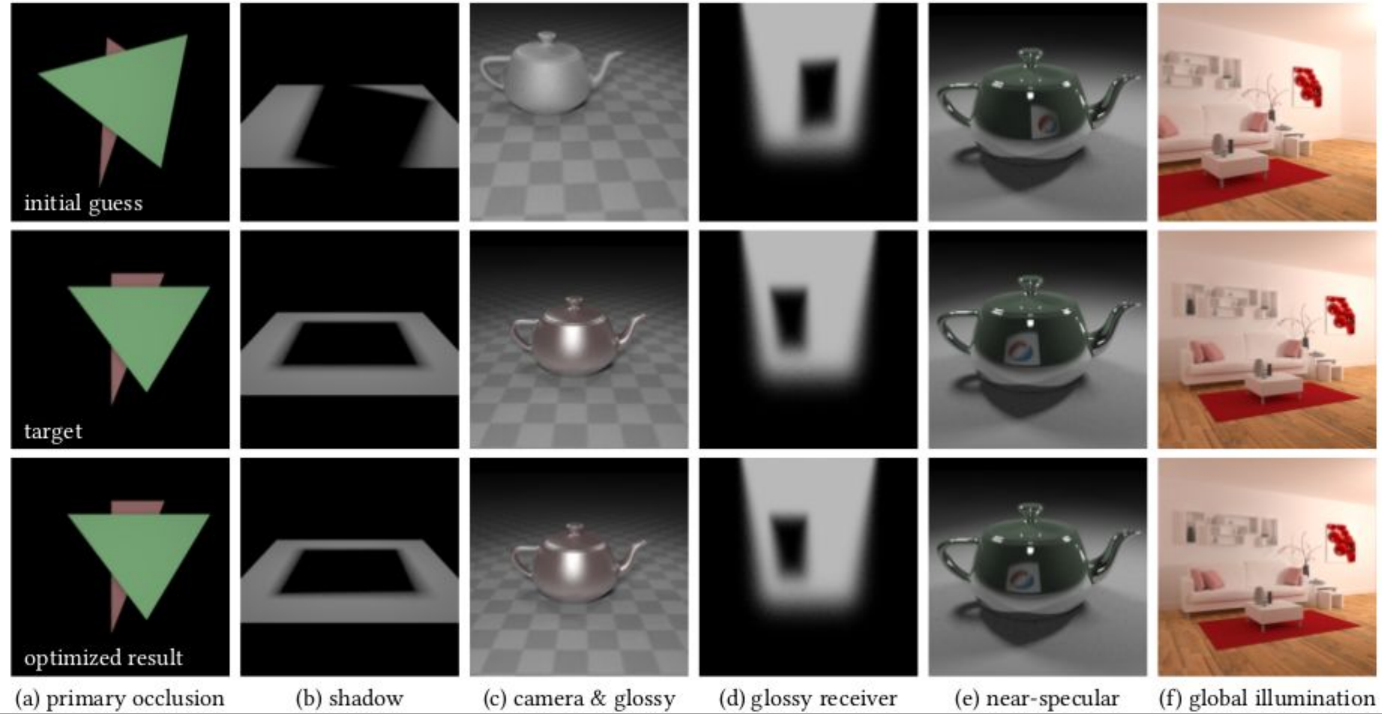
Phase III - Importance Sampling of an Edge.

- Oftentimes with in a BRDF, only a small portion of the edge has significant contribution.
- They numerically invert the integrated cumulative distribution function using Newton's method for importance sampling.
- Compare against the baseline approach of uniformly sampling all edges by length. The baseline approach is not able to efficiently sample rare events such as shadows cast by a small light source or very specular reflection of edges, while our importance sampling generates images with much lower variance.



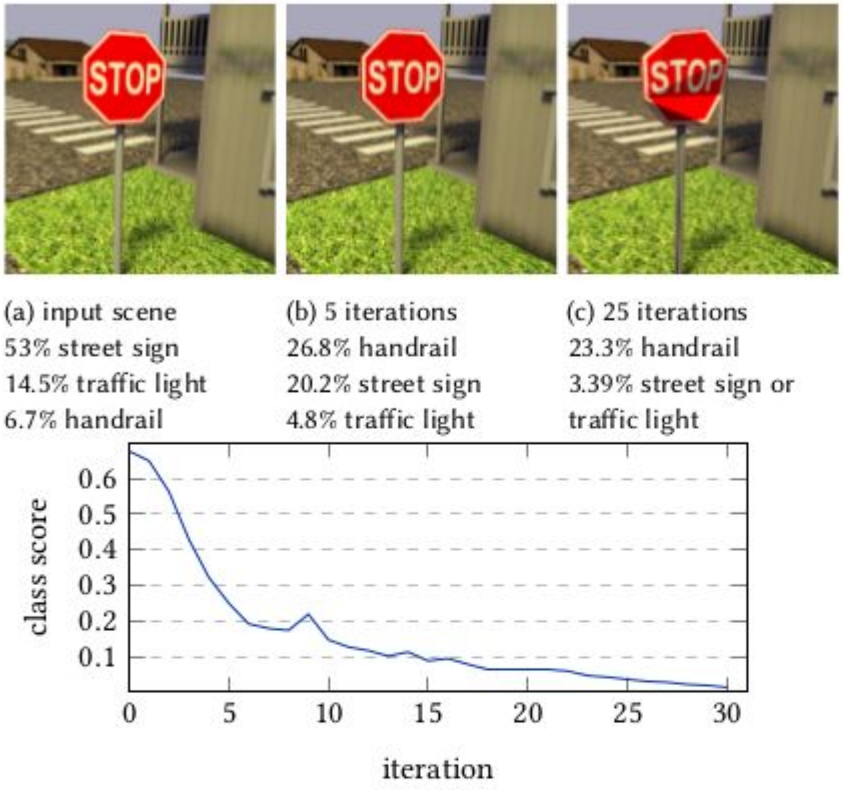
Results - Verification, Comparison, Application

Starting from an initial parameter, and try to optimize the parameters to minimize the L2 difference between the rendered image and target image using gradients generated by this method



Application - Inverse Rende

- To mine adversarial examples of 3D scenes, with the ability to backpropagate from image to scene parameters.
- A stop sign classified correctly as a street sign by the VGG16 . Then optimize for parameters including camera pose, light intensity, sun position, global translation, rotation, and vertex displacement of the stop sign.
- Gradient Descent
- In 5 iter, network starts to output handrail as the most probable class.



(d) combined class score of street sign and traffic light

Conclusion

- Introduced a differentiable Monte Carlo ray tracing algorithm that is capable of generating correct and unbiased gradients with respect to arbitrary input parameters such as scene geometry,camera, lights and materials.

- Introduced a novel edge sampling algorithm to take the geometric discontinuities into consideration, and derived the appropriate measure conversion.

- Integrated it whole with the autograd library of pytorch so we can backpropagate freely.

- **Wide range of applications including understanding workings of Neural Networks.**

Thank You!

Questions?

References

- Differentiable Rendering: https://people.csail.mit.edu/tzumao/diffrt/
- The Rendering Eq: http://www.cse.chalmers.se/edu/year/2011/course/TDA361/2007/rend_eq.pdf
- https://dai.fmph.uniba.sk/upload/c/cc/Ris_lesson04.pdf
- https://www.scratchapixel.com/lessons/mathematics-physics-for-computer-graphics/monte-carlo-methods-in-practice/monte-carlo-methods