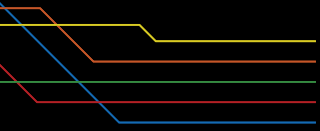


Spherical Harmonic Exponentials for Efficient Glossy Reflections

Ari Silvennoinen, Peter-Pike Sloan, Michal Iwanicki and Derek Nowrouzezahrai

Presented by Peter Shirley





Motivation

- Glossy reflections are crucial for realism in games, films, AR/VR

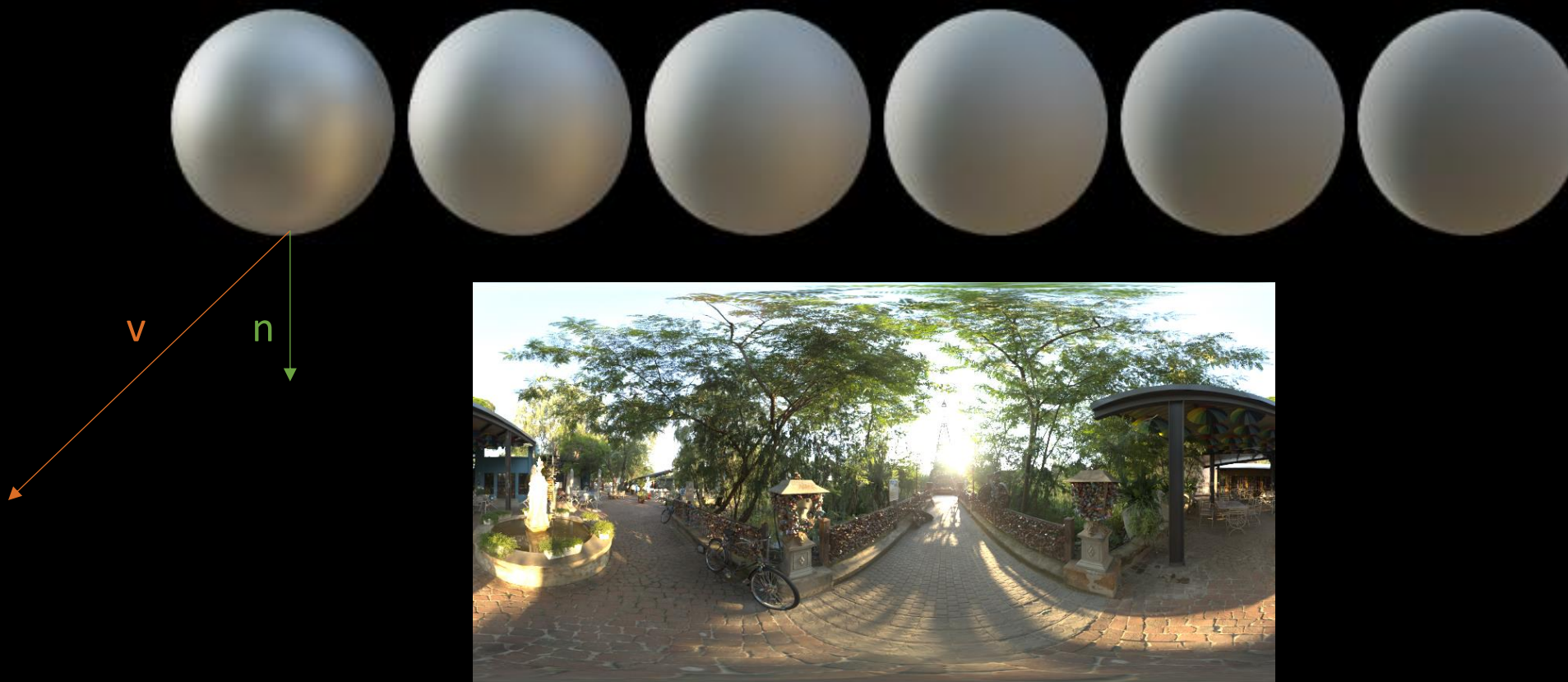


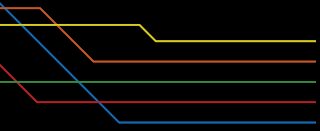
Increasing roughness 



Motivation

- Glossy reflections are crucial for realism in games, films, AR/VR





Motivation

- Glossy reflections are crucial for realism in games, films, AR/VR



\mathbf{v} \mathbf{n} \mathbf{l}

$$E(\mathbf{v}, \mathbf{n}, \Theta) = \int_{\Omega} L(\mathbf{l}) f(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$$



Motivation

- Glossy reflections are crucial for realism in games, films, AR/VR



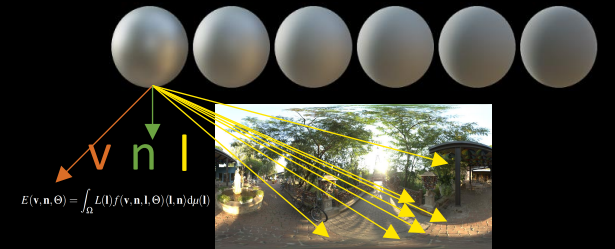
$$E(\mathbf{v}, \mathbf{n}, \Theta) = \int_{\Omega} L(\mathbf{l}) f(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$$

Expensive to evaluate!

Split-Sum Approximation

- Split-sum (split-integral) approximation is the most common method of approximating reflected radiance [Kar13]

$$E(\mathbf{v}, \mathbf{n}, \Theta) = \int_{\Omega} L(\mathbf{l}) f(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$$

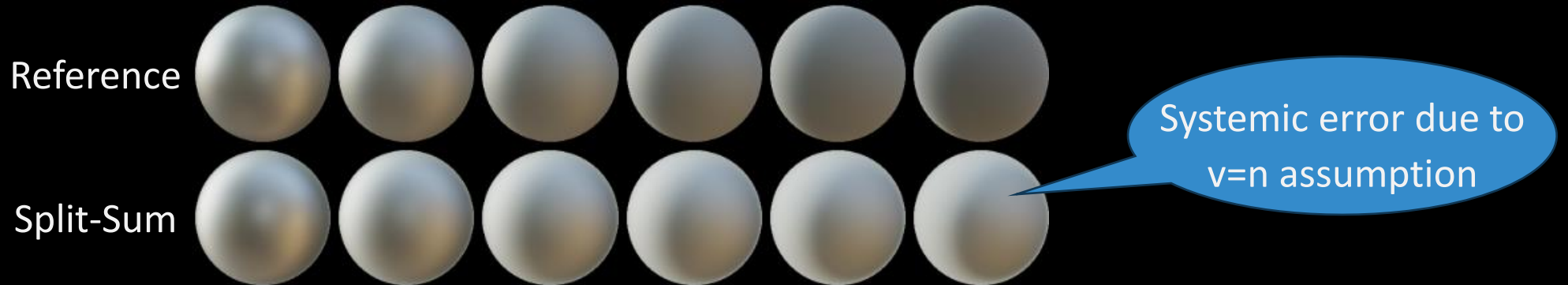


$$= \int_{\Omega} L(\mathbf{l}) \underbrace{\left[\frac{f(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta) \langle \mathbf{l}, \mathbf{n} \rangle}{\int f(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})} \right]}_{K(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta)} d\mu(\mathbf{l}) \times \int_{\Omega} f(\mathbf{v}, \mathbf{n}, \mathbf{l}, \Theta) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$$

- Further assuming $v=n$, we can prefilter the original environment light with a circularly symmetric kernel and tabulate the results for a single roughness value
- Reconstruction at runtime consists of precomputed texture lookups

Problems with Split-Sum

- View-independence assumption introduces systemic error
- High memory cost
 - For example, our biggest maps typically have ~8k reflection probes
 - ~700MB of storage even in low-resolution and compressed format
- Aliasing
 - Magnification artifacts are visible with low-resolution textures
 - To reduce aliasing, we need higher-resolution and thus potentially add to the memory cost





Our Contributions

- Alias-free representation for environment lighting
 - Analytic reconstruction
- Continuous roughness encoding
 - No need for a discrete roughness mip-chain compared to split-sum
- Explicitly take view-dependence into account
 - Reduces systemic approximation error from $v=n$ assumption
- Low memory cost and roughly equal runtime performance
 - For example, we can keep all the reflection probes in memory in our biggest maps
 - Reduces memory cost from 700MB to roughly 3MB
 - Over **200x** memory savings!



Method Overview

- Step 1: View-dependent factorization
- Step 2: Continuous roughness encoding
- Step 3: Solve optimization problem in log-space using linear least-squares
- Step 4: Runtime reconstruction

Log-space SH encoding



Pretville Street (HDR environment map)

Log-space SH encoding



Linear SH
Ringing artifacts



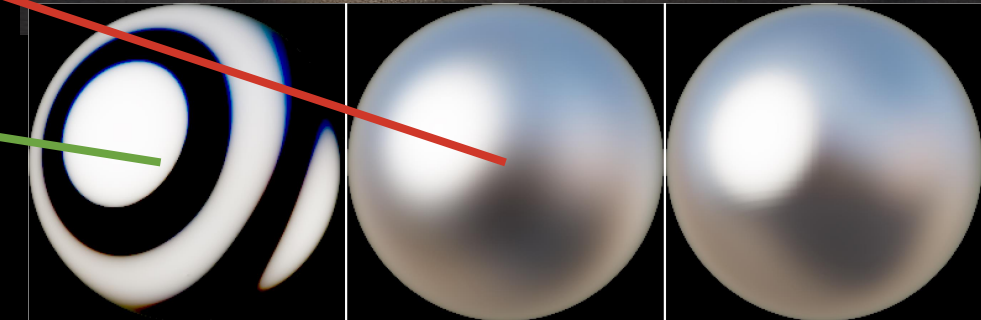
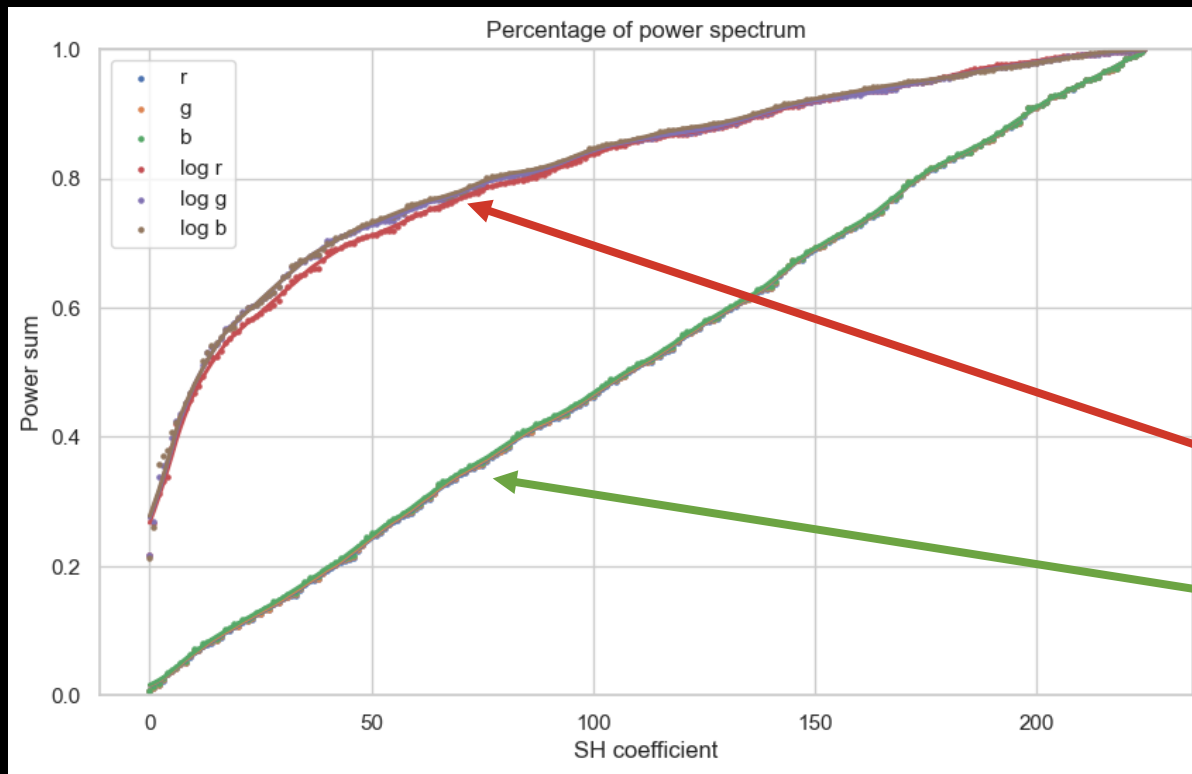
Log-space SH
Alias-free pointwise reconstruction



Split-sum environment map
High memory cost, aliasing artifacts

Log-space SH encoding

- Log-space radiance SH coefficient decay is superlinear and has zero ringing artifacts



Linear SH

Log-space SH

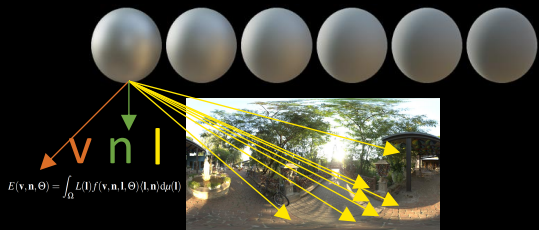


Method Overview

- Step 1: View-dependent factorization
- Step 2: Continuous roughness encoding
- Step 3: Solve optimization problem in log-space using linear least-squares
- Step 4: Runtime reconstruction

Our Factorization

- Similar to split-sum, we use Cook-Torrance BRDF with GGX [WMLT07]
- We apply Schlick Fresnel approximation to decouple the Fresnel term from the BRDF



$$E(\mathbf{v}, \mathbf{n}, \alpha) \approx F_0 E_0(\mathbf{v}, \mathbf{n}, \alpha) + (1 - F_0) E_1(\mathbf{v}, \mathbf{n}, \alpha)$$

Base reflectance

$$E_0(\mathbf{v}, \mathbf{n}, \alpha) = \int_{\Omega} L(\mathbf{l}) f_{DV}(\mathbf{v}, \mathbf{n}, \mathbf{l}, \alpha) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$$

Fresnel tail reflectance

$$E_1(\mathbf{v}, \mathbf{n}, \alpha) = \int_{\Omega} L(\mathbf{l}) f_{DV}(\mathbf{v}, \mathbf{n}, \mathbf{l}, \alpha) (1 - \langle \mathbf{v}, \mathbf{h} \rangle)^5 \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$$



Our Factorization

- First, we factorize base reflectance as a product of two separable terms, P and Q [MAA01]
 - Enables dimensionality reduction
 - Models explicit view-dependence
- We use a new reflection-half-reflection parametrization

Base reflectance $E_0(\mathbf{v}, \mathbf{n}, \alpha) \approx P(\mathbf{r}, \alpha) Q(\mathbf{h}_r, \alpha)$

Half-reflection $\mathbf{h}_r = (\mathbf{n} + \mathbf{r}) / \|\mathbf{n} + \mathbf{r}\|$



Method Overview

- Step 1: View-dependent factorization
- **Step 2: Continuous roughness encoding**
- Step 3: Solve optimization problem in log-space using linear least-squares
- Step 4: Runtime reconstruction

Continuous Roughness Encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding [VHM*22]

$$E_0(\mathbf{v}, \mathbf{n}, \alpha) \approx P(\mathbf{r}, \alpha) Q(\mathbf{h}_r, \alpha)$$

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

$$Q(\mathbf{h}_r, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{h}_r), \mathbf{q} \right\rangle\right)$$

Continuous Roughness Encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding [VHM*22]

$$E_0(\mathbf{v}, \mathbf{n}, \alpha) \approx P(\mathbf{r}, \alpha) Q(\mathbf{h}_r, \alpha)$$

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

$$Q(\mathbf{h}_r, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{h}_r), \mathbf{q} \right\rangle\right)$$

p and q are the
unknown coefficient
vectors

Continuous Roughness Encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding [VHM*22]

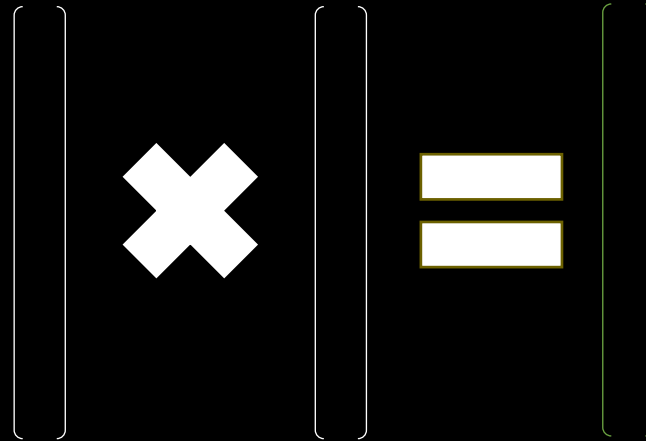
$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

Continuous Roughness Encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding [VHM*22]

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

Elementwise product
of two SH vectors




Continuous Roughness Encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding [VHM*22]

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

Inner product of two SH vectors



Continuous roughness encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

Exponentiate the dot product \mathbf{z}

$\exp(\mathbf{z})$

Continuous Roughness Encoding

- Introduce SH exponentials for the terms P and Q
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
 - Also known as directional encoding [VHM*22]

$$E_0(\mathbf{v}, \mathbf{n}, \alpha) \approx P(\mathbf{r}, \alpha) Q(\mathbf{h}_r, \alpha)$$

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

$$Q(\mathbf{h}_r, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{h}_r), \mathbf{q} \right\rangle\right)$$

p and q are the
unknown coefficient
vectors



Method Overview

- Step 1: View-dependent factorization
- Step 2: Continuous roughness encoding
- Step 3: Solve optimization problem in log-space using linear least-squares
- Step 4: Runtime reconstruction



Optimization Problem

- We find the unknown coefficients vectors \mathbf{p} and \mathbf{q} by solving an optimization problem
 - Linear least squares problem in log-space
 - Log-transform “linearizes” the product $P \times Q$ [MAA01]

Objective

$$\underset{\mathbf{p}, \mathbf{q}}{\text{minimize}} \left(\log(P(\mathbf{r}, \alpha)Q(\mathbf{h}_r, \alpha)) - \log(E_0(\mathbf{v}, \mathbf{n}, \alpha)) \right)^2$$

Optimization Problem

- We find the unknown coefficients vectors \mathbf{p} and \mathbf{q} by solving an optimization problem
 - Linear least squares problem in log-space
 - Log-transform “linearizes” the product $P \times Q$
- We solve the optimization problem using **stochastic least squares** over the continuous roughness range

Objective

$$\underset{\mathbf{p}, \mathbf{q}}{\text{minimize}} \left(\log(P(\mathbf{r}, \alpha)Q(\mathbf{h}_r, \alpha)) - \log(E_0(\mathbf{v}, \mathbf{n}, \alpha)) \right)^2$$

Row constraint

$$\left[\mathbf{vMF} \left(\frac{1}{\alpha} \right) \odot \mathbf{Y}(\mathbf{r}) \quad \mathbf{vMF} \left(\frac{1}{\alpha} \right) \odot \mathbf{Y}(\mathbf{h}_r) \right] \begin{bmatrix} \mathbf{p} \\ \mathbf{q} \end{bmatrix} = \log(E_0(\mathbf{v}, \mathbf{n}, \alpha))$$

\mathbf{p} and \mathbf{q} are the unknown coefficient vectors



Method Overview

- Step 1: View-dependent factorization
- Step 2: Continuous roughness encoding
- Step 3: Solve optimization problem in log-space using linear least-squares
- Step 4: Runtime reconstruction

Runtime Reconstruction

- Efficient reconstruction using the only coefficient vectors \mathbf{p} and \mathbf{q}
- Use von Mises-Fisher distributions $\mathbf{vMF}(\kappa)$ encode roughness $\mathbf{vMF}_l(\kappa) = \exp\left(\frac{-l(l+1)}{2\kappa}\right)$
- Supports continuous roughness range using the same coefficients \mathbf{p} and \mathbf{q}

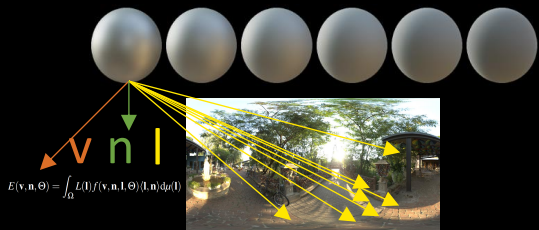
$$E_0(\mathbf{v}, \mathbf{n}, \alpha) \approx P(\mathbf{r}, \alpha) Q(\mathbf{h}_r, \alpha)$$

$$P(\mathbf{r}, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{r}), \mathbf{p} \right\rangle\right)$$

$$Q(\mathbf{h}_r, \alpha) = \exp\left(\left\langle \mathbf{vMF}\left(\frac{1}{\alpha}\right) \odot \mathbf{Y}(\mathbf{h}_r), \mathbf{q} \right\rangle\right)$$

Fresnel Tail Reflectance

- We model only base reflectance directly and approximate Fresnel tail reflectance, similar to [Kar13]



$$E(\mathbf{v}, \mathbf{n}, \alpha) \approx F_0 E_0(\mathbf{v}, \mathbf{n}, \alpha) + (1 - F_0) E_1(\mathbf{v}, \mathbf{n}, \alpha)$$

Base reflectance $E_0(\mathbf{v}, \mathbf{n}, \alpha) = \int_{\Omega} L(\mathbf{l}) f_{DV}(\mathbf{v}, \mathbf{n}, \mathbf{l}, \alpha) \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$

Fresnel tail reflectance $E_1(\mathbf{v}, \mathbf{n}, \alpha) \approx E_0(\mathbf{v}, \mathbf{n}, \alpha) \int_{\Omega} f_{DV}(\mathbf{v}, \mathbf{n}, \mathbf{l}, \alpha) (1 - \langle \mathbf{v}, \mathbf{h} \rangle)^5 \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$

Fresnel Tail Reflectance

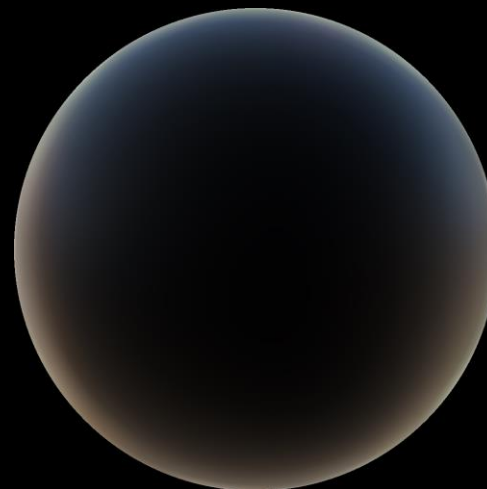


- We model only base reflectance directly and approximate Fresnel tail reflectance, similar to [Kar13]

Fresnel tail approximation

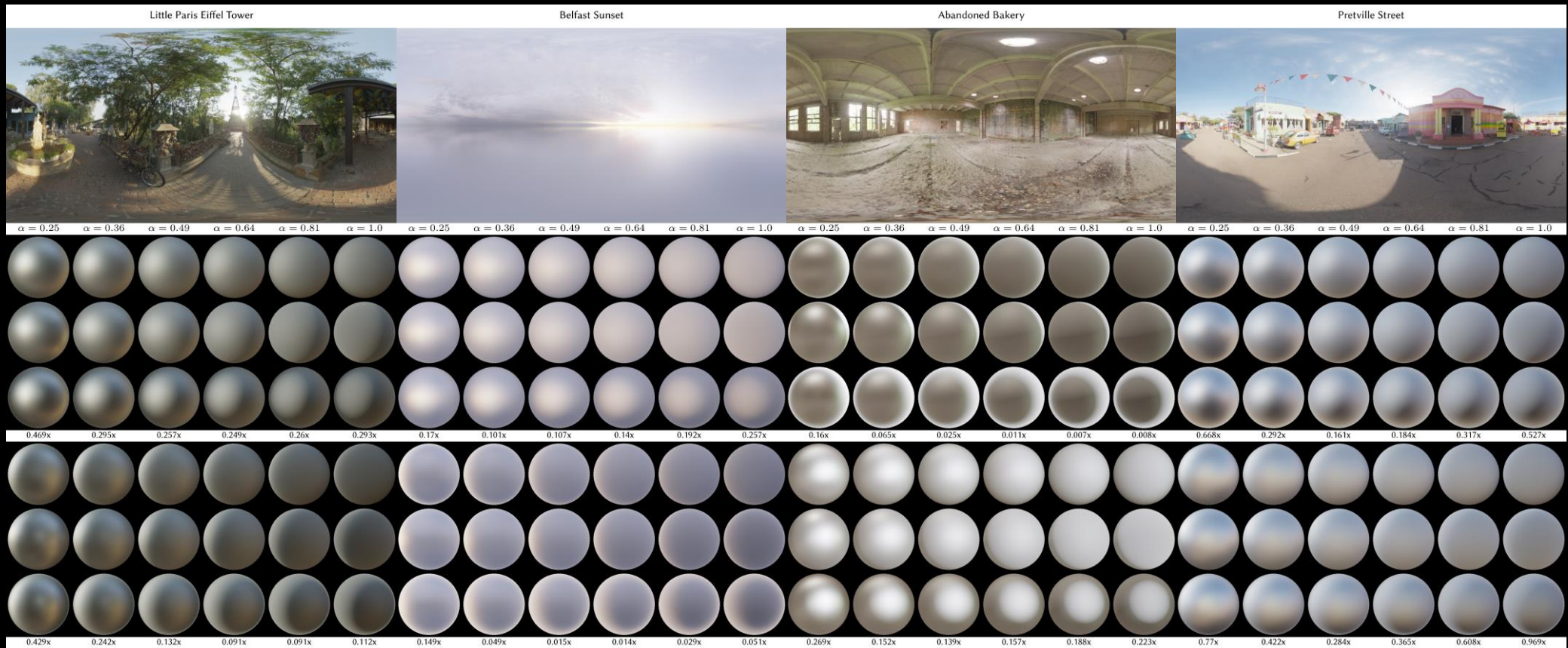


Reference

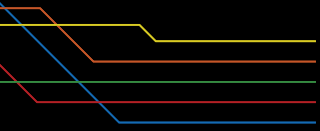


Fresnel tail reflectance $E_1(\mathbf{v}, \mathbf{n}, \alpha) \approx E_0(\mathbf{v}, \mathbf{n}, \alpha) \int_{\Omega} f_{DV}(\mathbf{v}, \mathbf{n}, \mathbf{l}, \alpha) (1 - \langle \mathbf{v}, \mathbf{h} \rangle)^5 \langle \mathbf{l}, \mathbf{n} \rangle d\mu(\mathbf{l})$

Results



More results in the paper



Results



$\alpha = 0.25$ $\alpha = 0.36$ $\alpha = 0.49$ $\alpha = 0.64$ $\alpha = 0.81$ $\alpha = 1.0$

Ours



Reference

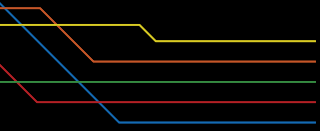


Split-sum



0.469x 0.295x 0.257x 0.249x 0.26x 0.293x





Results



$\alpha = 0.25$ $\alpha = 0.36$ $\alpha = 0.49$ $\alpha = 0.64$ $\alpha = 0.81$ $\alpha = 1.0$

Ours



Reference



Split-sum

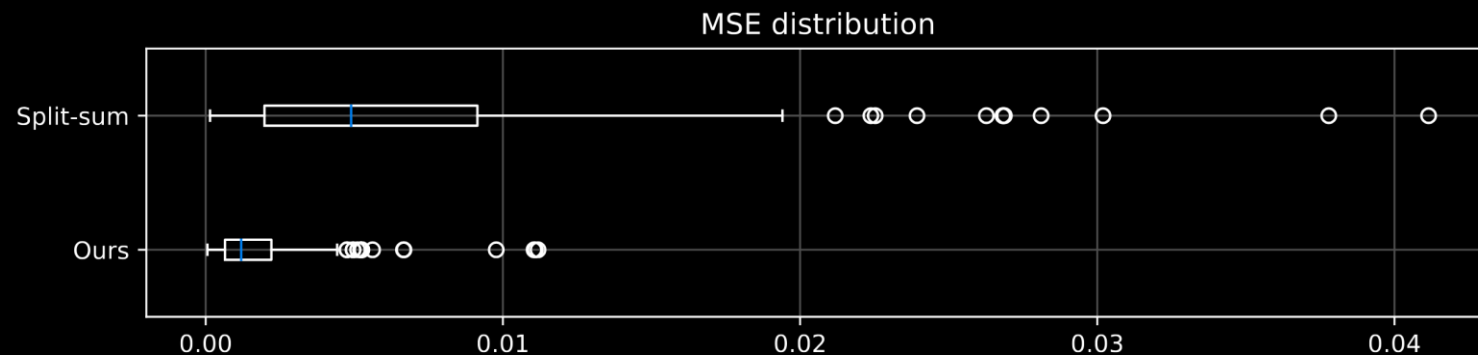


0.469x 0.295x 0.257x 0.249x 0.26x 0.293x



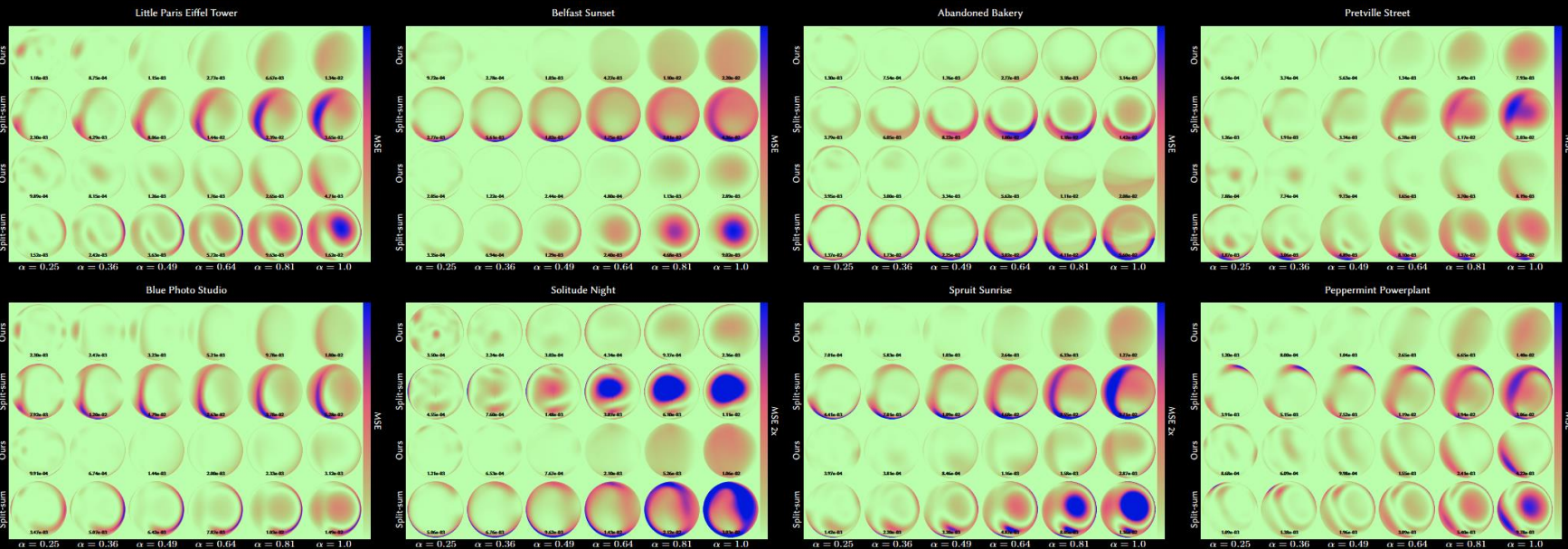
Performance

- Precomputation time is **0.328ms** compared to **0.536ms** for split-sum
 - Averaged over 1000 GPU runs
- Runtime overhead is **0.1ms** in 1920x1080 of total GPU frame time
- Our method has lower MSE in 95% of our test cases while using over 200x less memory compared to split sum

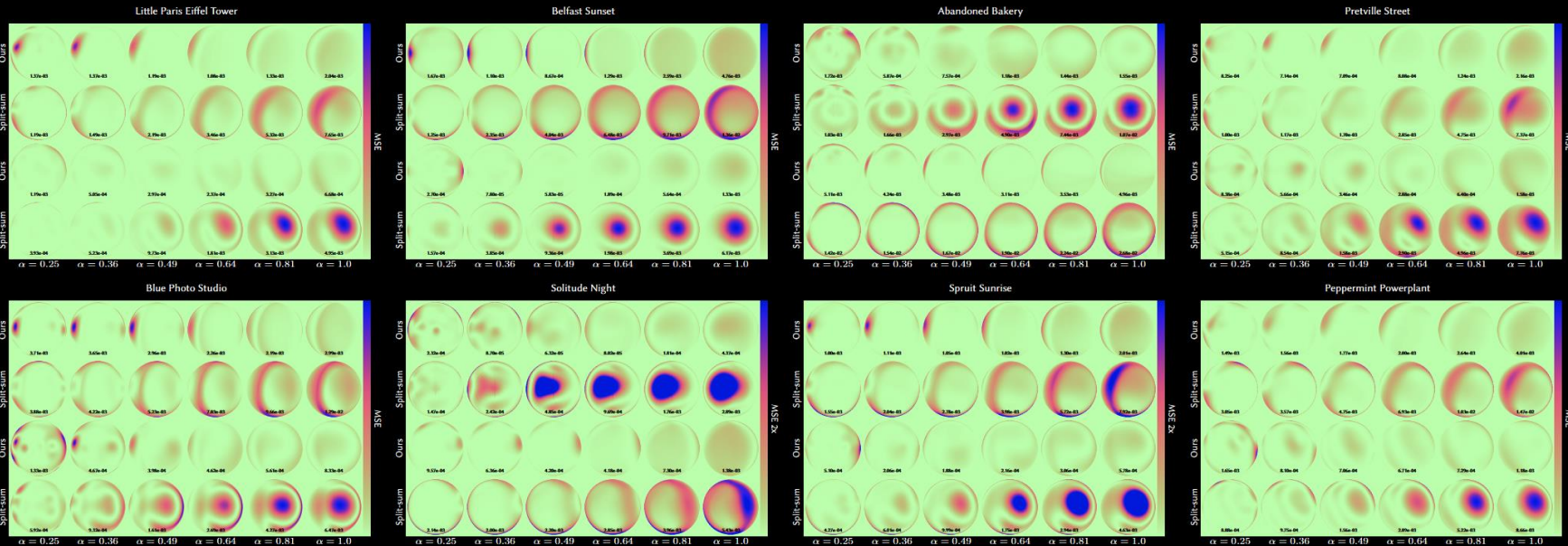


Error Distribution

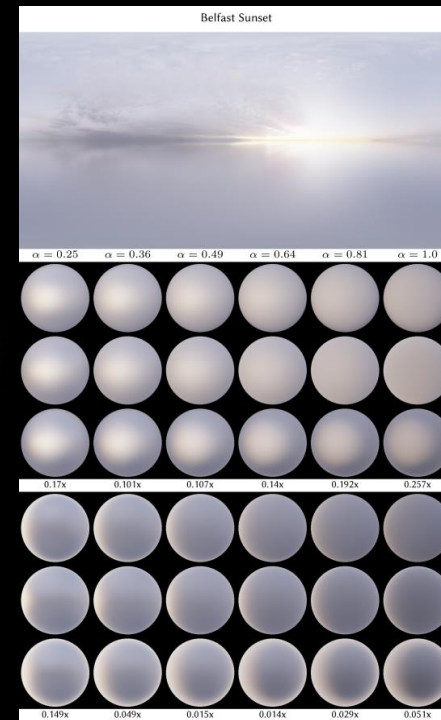
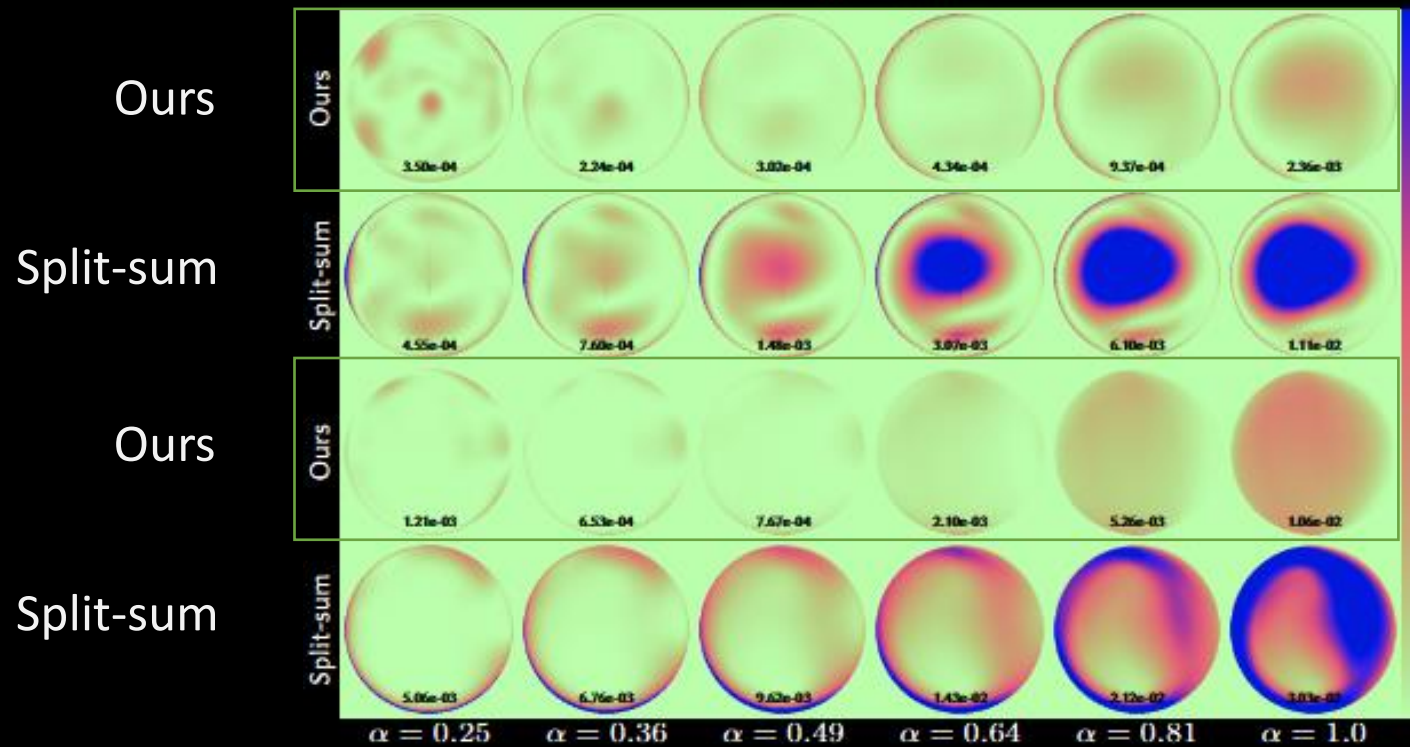
Metals



Non-metals



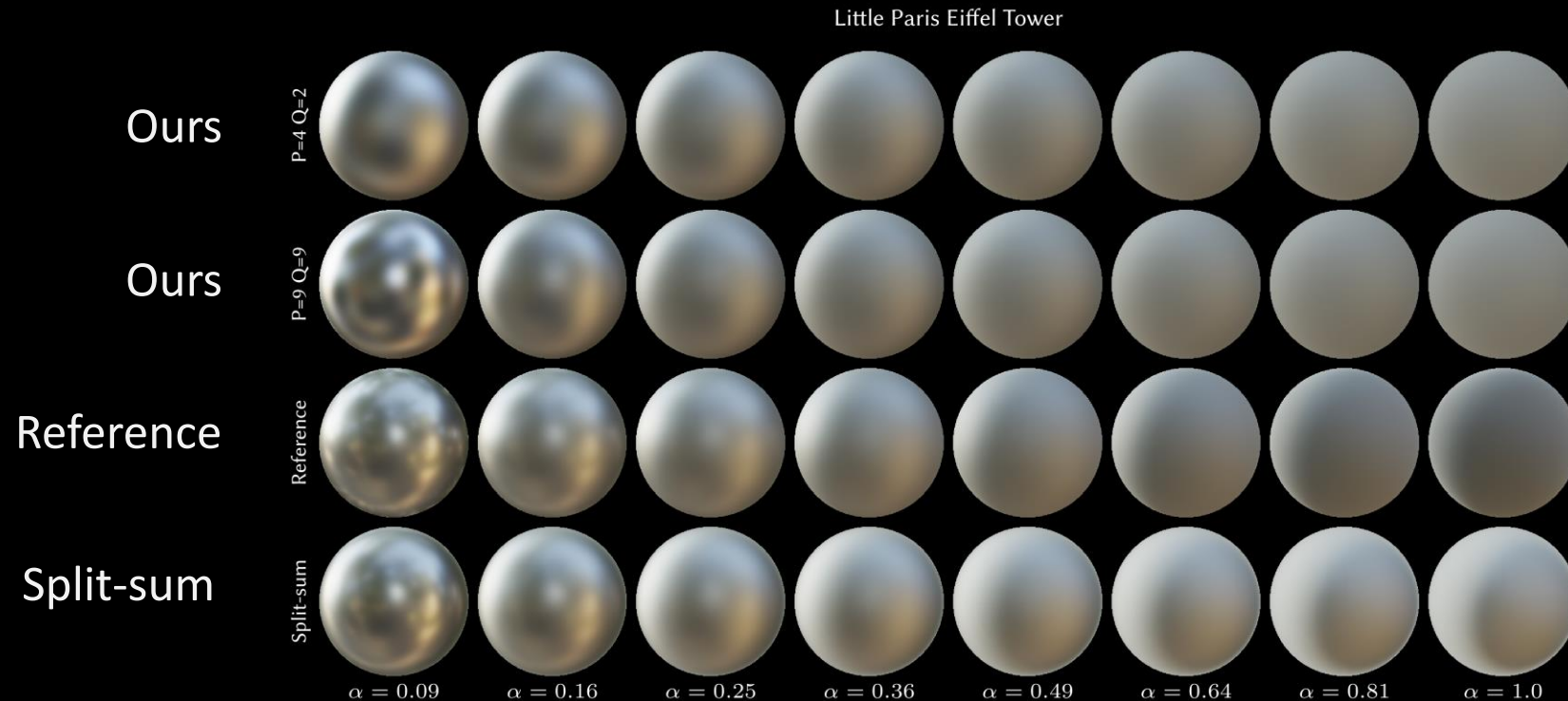
Error Distribution



Increasing roughness \longrightarrow

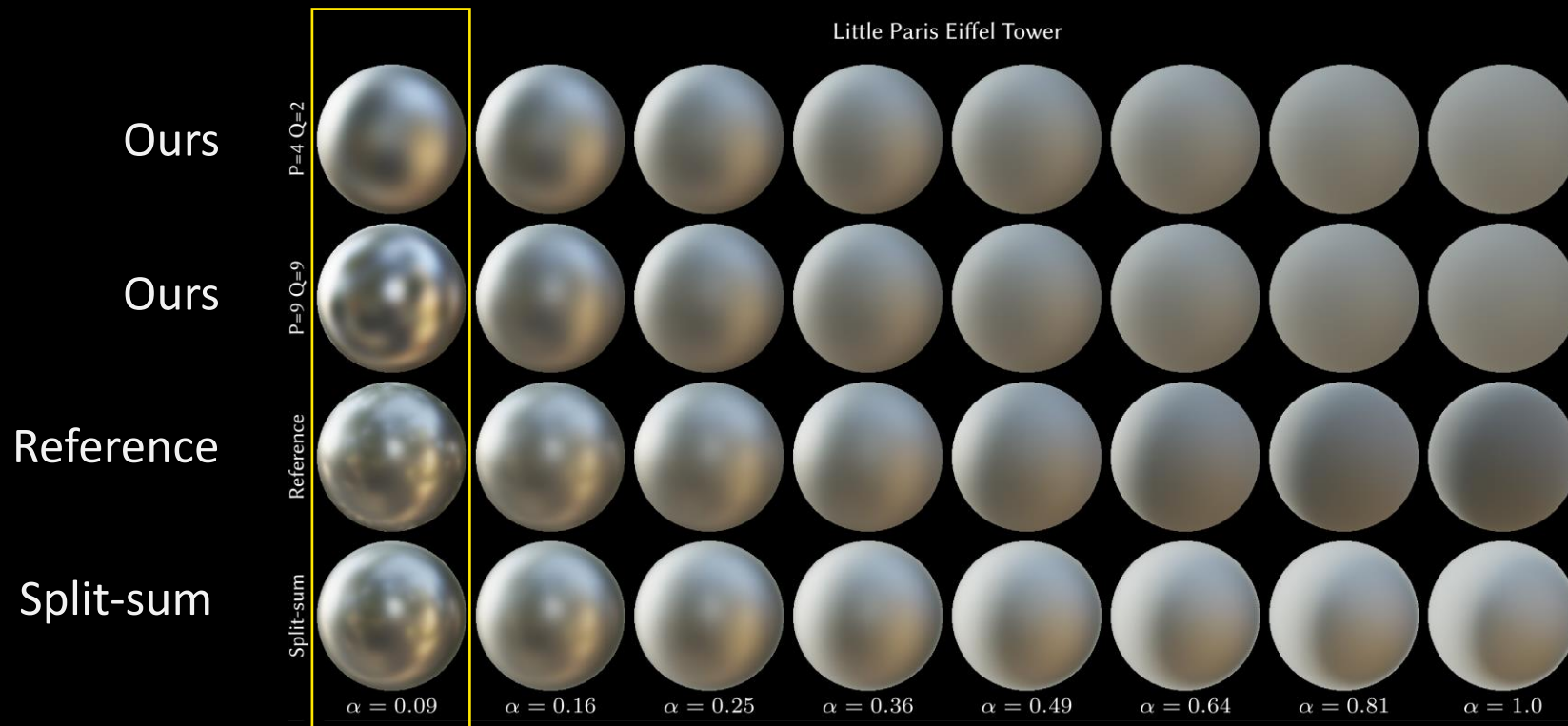
Limitations

- Main limitation is the representation capacity of the low-order SH model
- Experimentally, we found that 33 coefficients per color channel provides reasonable approximation for roughness range of [0.25, 1.0]



Limitations

- Main limitation is the representation capacity of the low-order SH model
- Experimentally, we found that 33 coefficients per color channel provides reasonable approximation for roughness range of [0.25, 1.0]



Summary

- Introduced a novel glossy reflection representation using factorized spherical harmonic exponentials
- Achieves higher reflection fidelity with significantly lower memory usage
- Enables **efficient, alias-free** reconstruction across continuous angular and roughness domains
- Enables scalable, spatially localized glossy reflections previously infeasible due to high memory cost



All materials use our method



Thank You!

- Questions?

Spherical Harmonic Exponentials for
Efficient Glossy Reflections



Acknowledgements

Peter Shirley

Feng Xie

Michael Vance

Danny Chan

Julie Haining

Jennifer Velazquez

Natasha Tatarchuk

Adrien Dubouchet

Activision Central Tech



Bonus Slides



Ablation study

Varying the degrees for both P and Q terms

