# EWA Volume Splatting

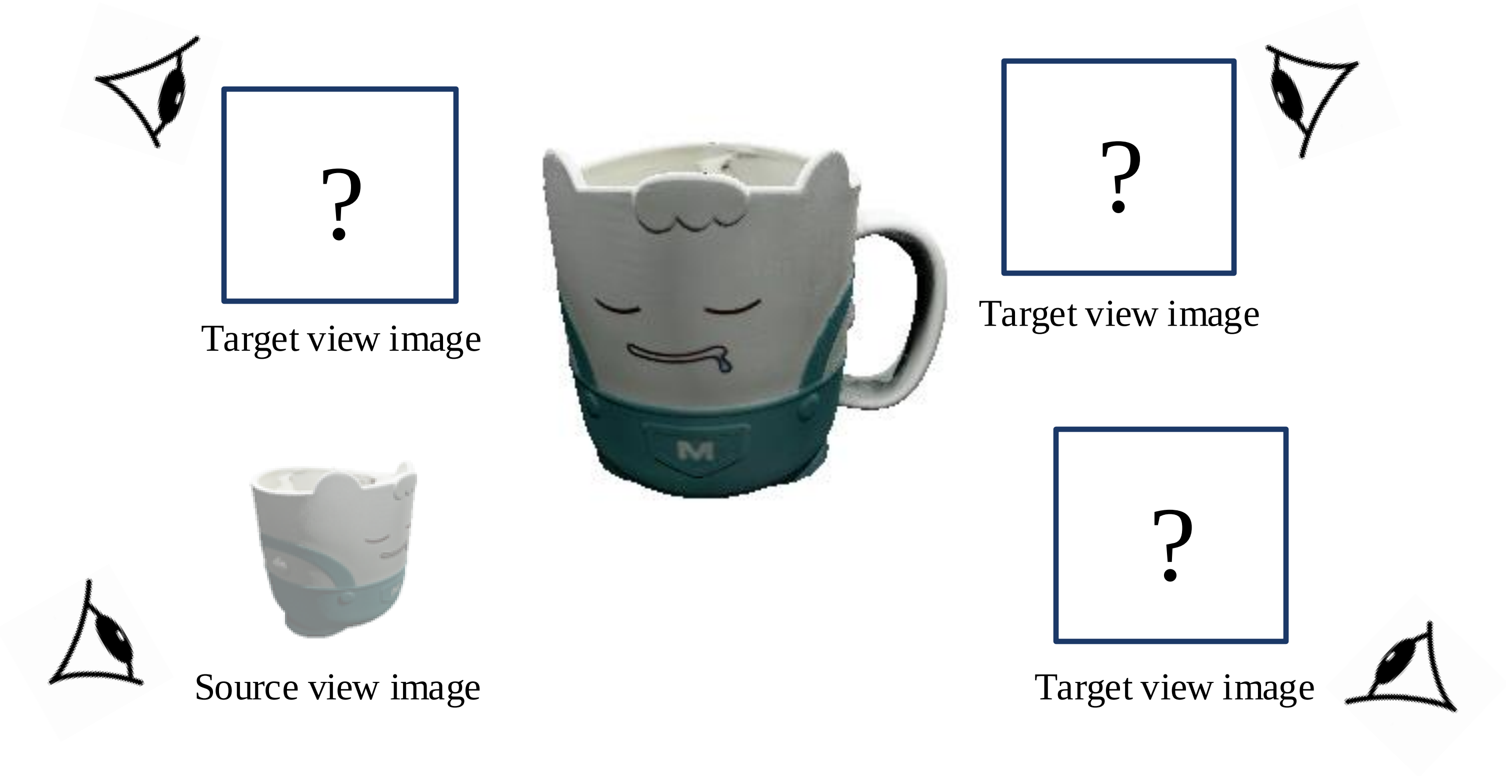
**Matthias Zwicker    Hanspeter Pfister    Jeroen van Baar    Markus Gross**
**Proceedings Visualization, 2001. VIS'01**

**Weekly Meeting - 2024-01-12**
**KAIST Geometric AI Lab - Jaihoon Kim**

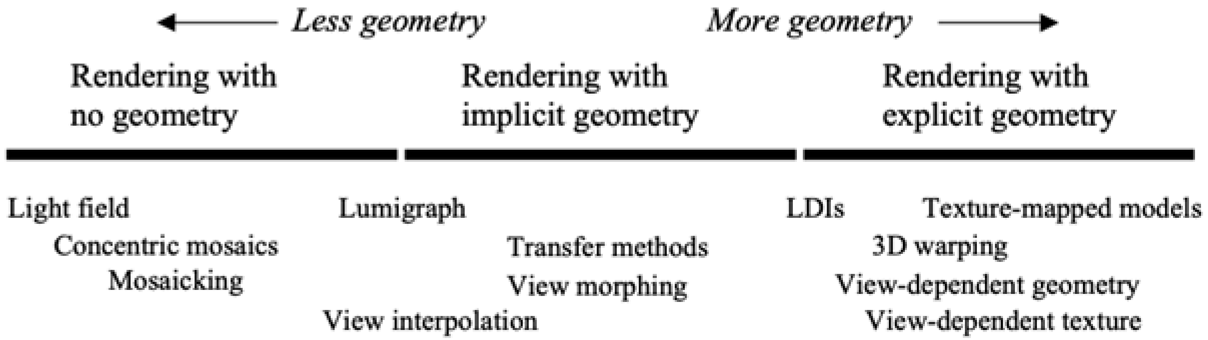
# Introduction

# Novel View Synthesis

Given an input image, synthesizing new images of the same object or scene observed from arbitrary viewpoints.



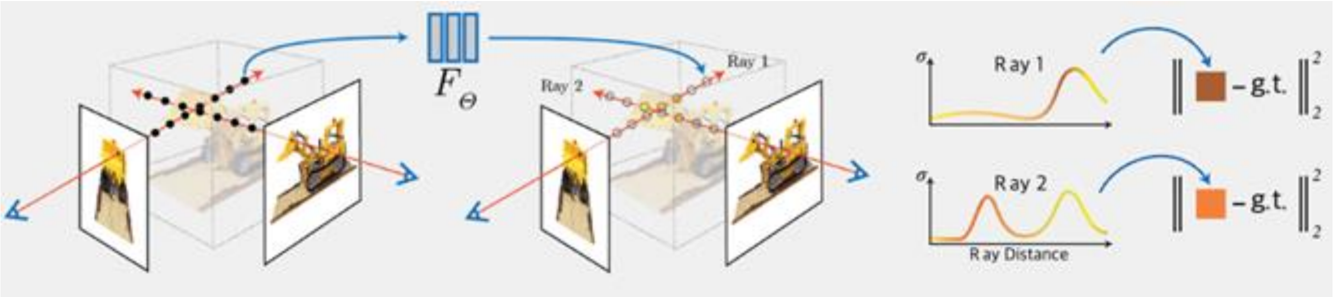
# Novel view synthesis: History




| Dimension | Year | Viewing space | Name |
|---|---|---|---|
| 7 | 1991 | free | Plenoptic function |
| 5 | 1995 | free | Plenoptic modeling |
| 4 | 1996 | bounding box | Lightfield/Lumigraph |
| 3 | 1999 | bounding plane | Concentric mosaics |
| 2 | 1994 | fixed point | Cylindrical/Spherical panorama |

Gortler, Steven J., et al. "The lumigraph." Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. 1996.

# NeRF

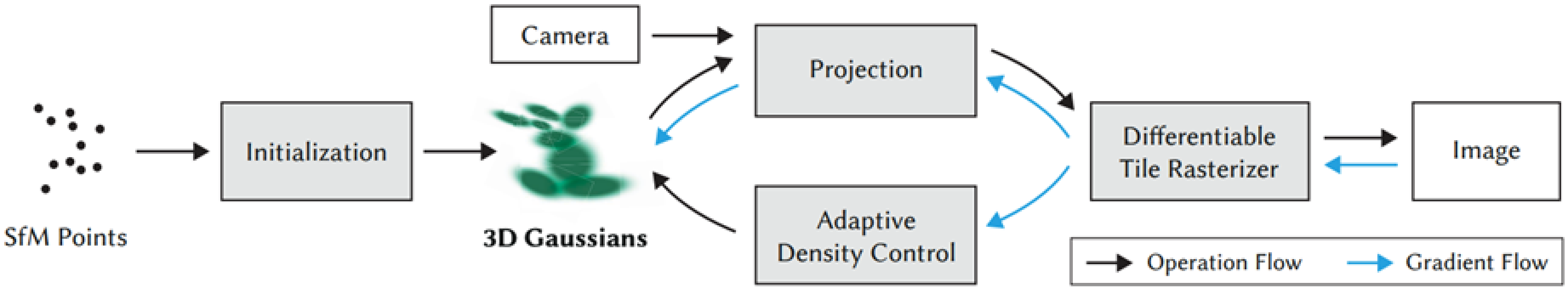



NeRF (ECCV 20)

NeRFacto (SIGGRAPH 23)

MipNeRF360 (CVPR 22)

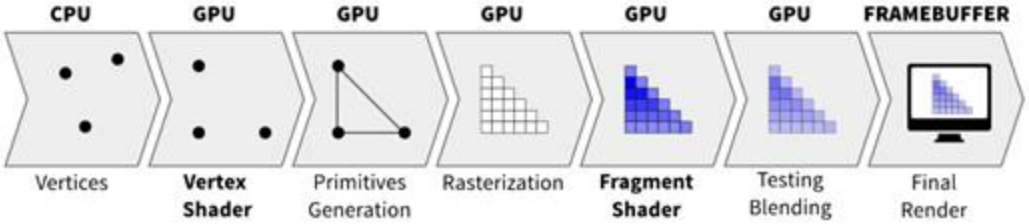
# 3D Gaussian Splatting



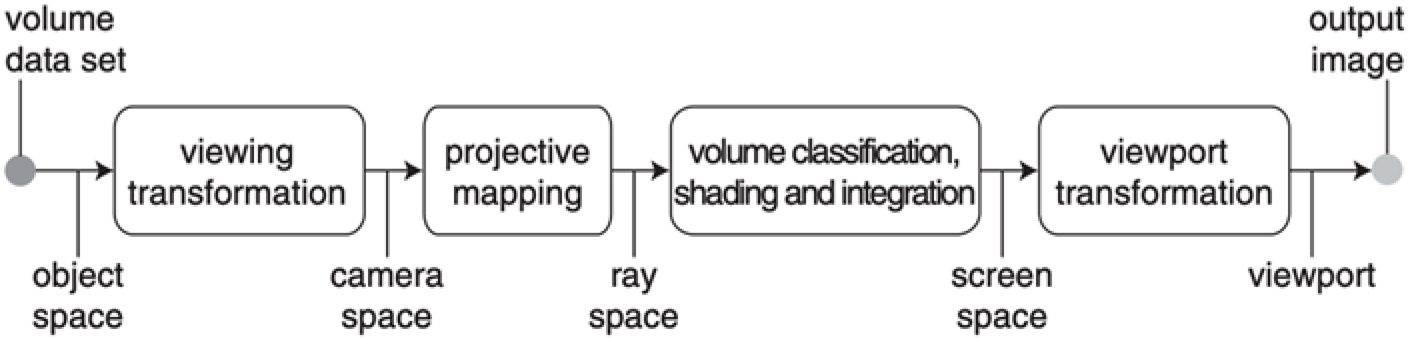

3D Gaussian Splatting (SIGGRAPH 23)

# The First Gaussian Splatting

Framework for direct volume rendering using a splatting approach based on elliptical Gaussian kernels




Conventional rendering pipeline

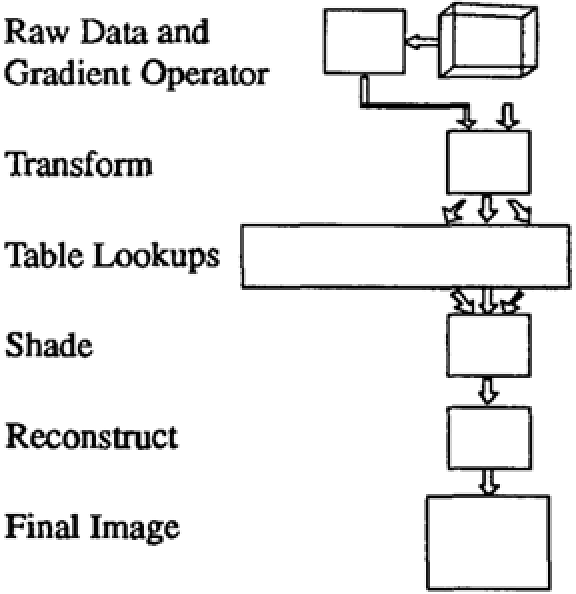



EWA Volume Splatting (VIS 01)

# Related Work

# Related Work (i)

- **The original Splatting Algorithm**

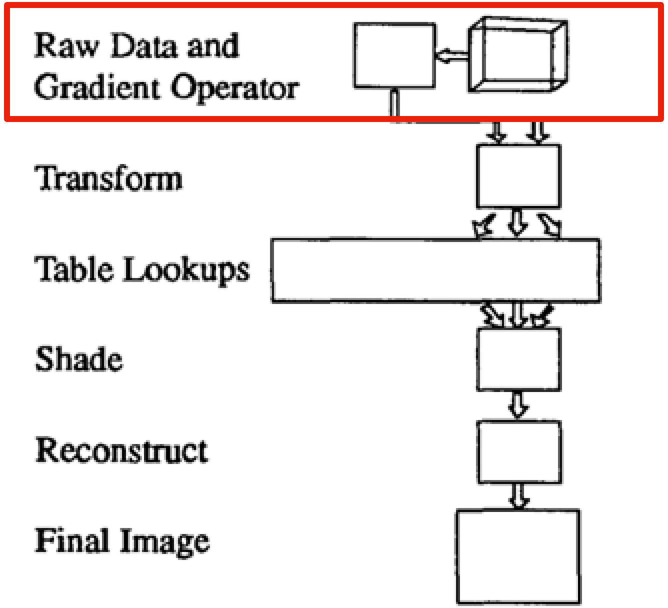



Interactive Volume Rendering (VVS 89)

Wang, Qianqian, et al. "Ibrnet: Learning multi-view image-based rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

# Related Work (i)

- **The original Splatting Algorithm**




Interactive Volume Rendering (VVS 89)

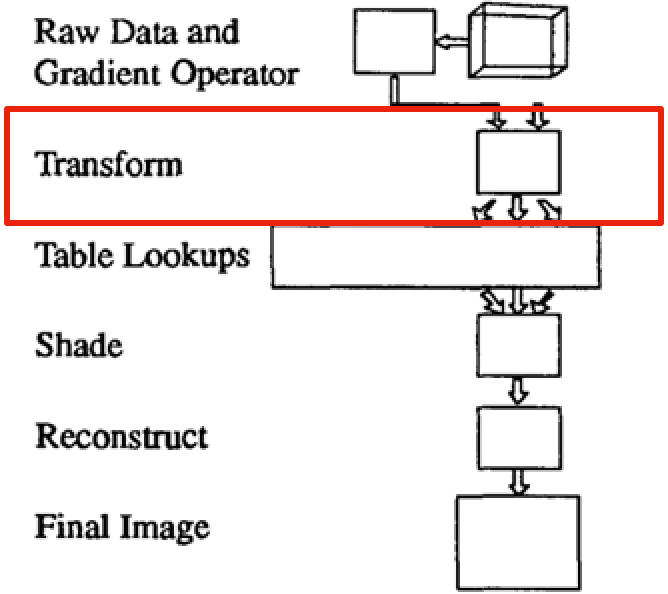
$$\begin{bmatrix} \textit{density value} \\ \textit{gradient strength} \\ \textit{gradient } \langle i,j,k \rangle \textit{ direction} \\ \textit{grid } \langle i,j,k \rangle \end{bmatrix}$$

Input sample packet

Wang, Qianqian, et al. "Ibrnet: Learning multi-view image-based rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

# Related Work (i)

- **The original Splatting Algorithm**




Interactive Volume Rendering (VVS 89)

$$\begin{bmatrix} \textit{density value} \\ \textit{gradient strength} \\ \textit{gradient } \langle i,j,k\rangle \textit{ direction} \\ \textit{grid } \langle i,j,k\rangle \end{bmatrix}$$

$$\Downarrow$$

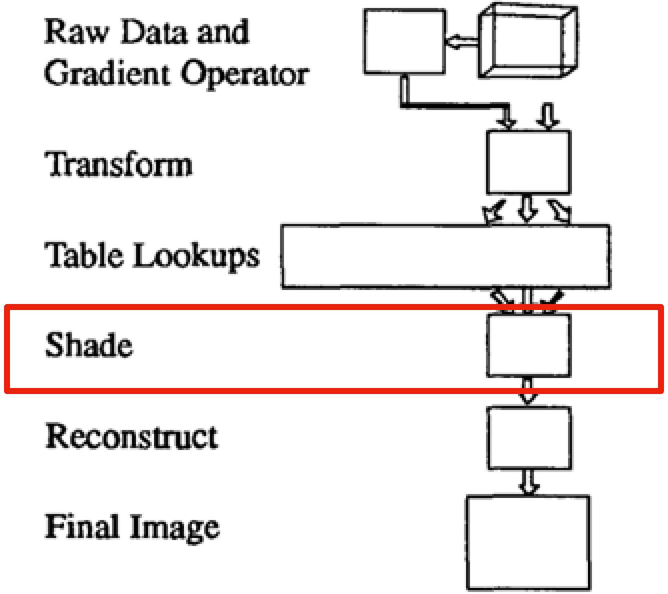
$$\begin{bmatrix} \textit{density value} \\ \textit{gradient strength} \\ \textit{gradient } \langle i,j,k\rangle \textit{ direction} \\ \textit{screen } \langle x,y,z\rangle \end{bmatrix} \qquad \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} dx/di & dx/dj & dx/dk \\ dy/di & dy/dj & dy/dk \\ dz/di & dz/dj & dz/dk \end{bmatrix} \begin{bmatrix} \Delta i \\ \Delta j \\ \Delta k \end{bmatrix}$$

Grid space → screen space ***orthographic*** views

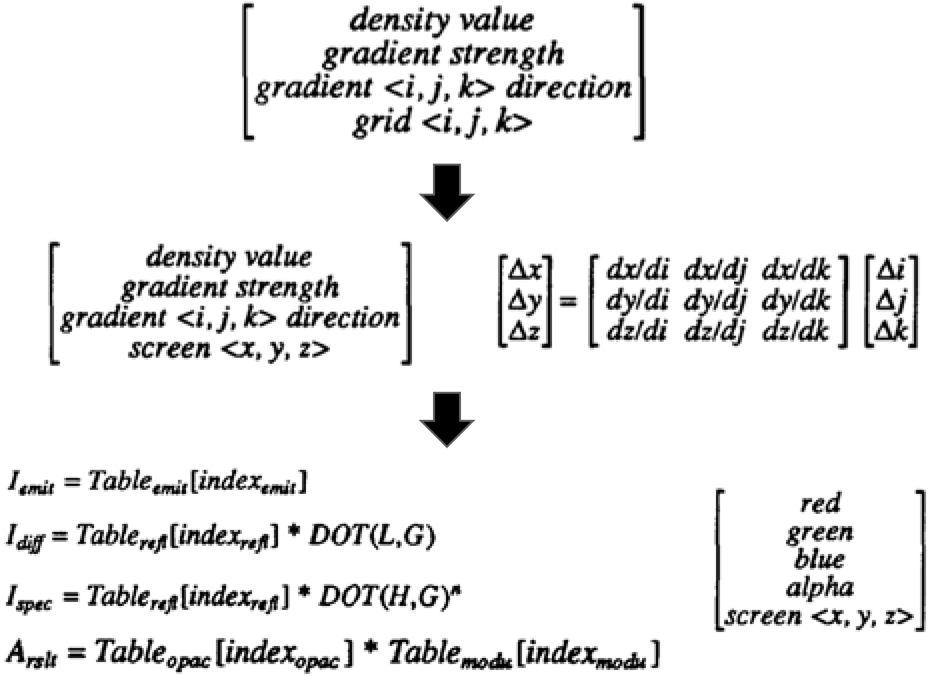

Wang, Qianqian, et al. "Ibrnet: Learning multi-view image-based rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

# Related Work (i)

- **The original Splatting Algorithm**




Interactive Volume Rendering (VVS 89)

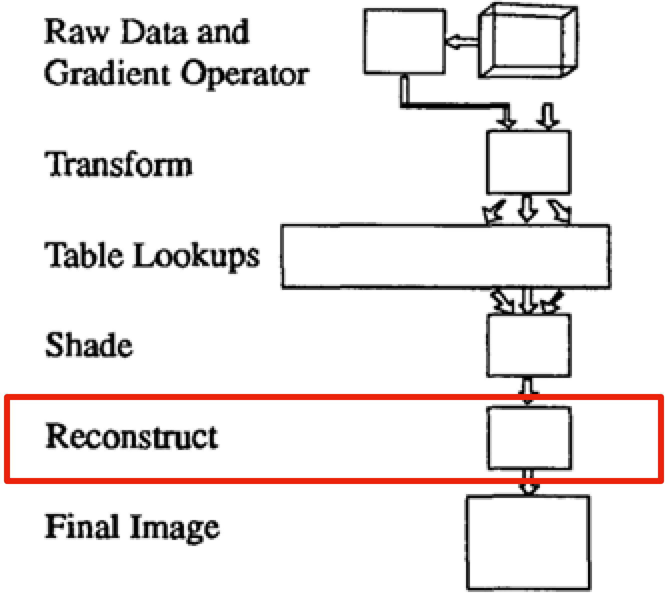



G (Gradient direction), A (Opacity), L (light vector),
H (Vector pointing midpoint between eye and light)

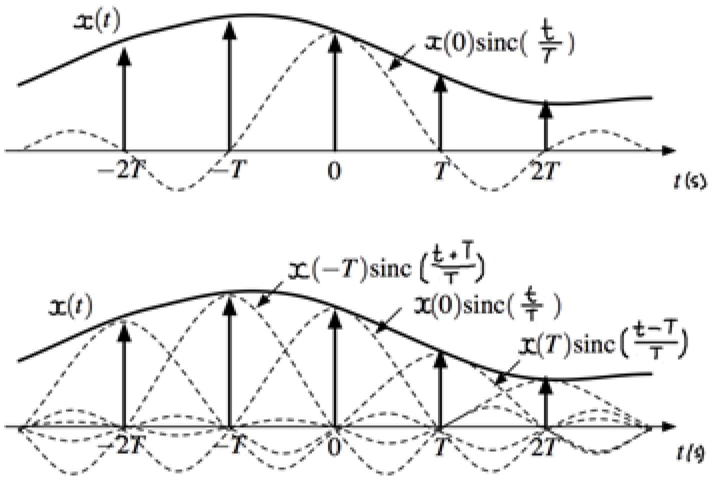
Wang, Qianqian, et al. "Ibrnet: Learning multi-view image-based rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

# Related Work (i)

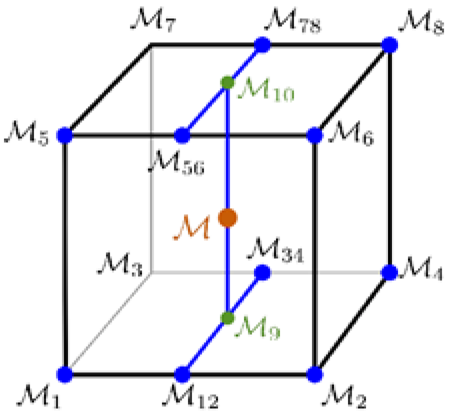
## - The original Splatting Algorithm




Interactive Volume Rendering (VVS 89)




Sinc function convolution

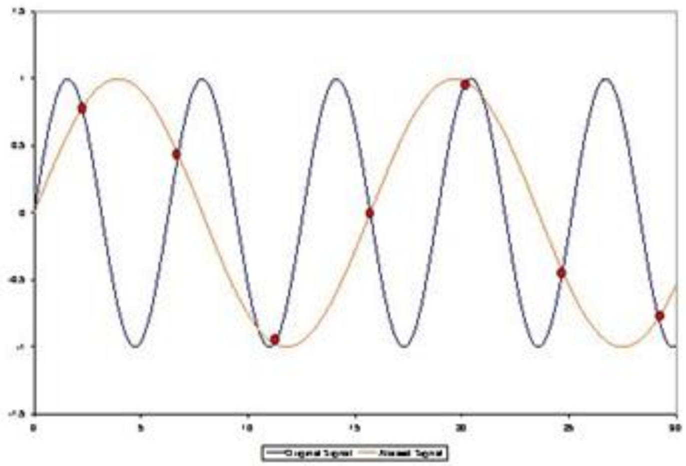



Trilinear interpolation

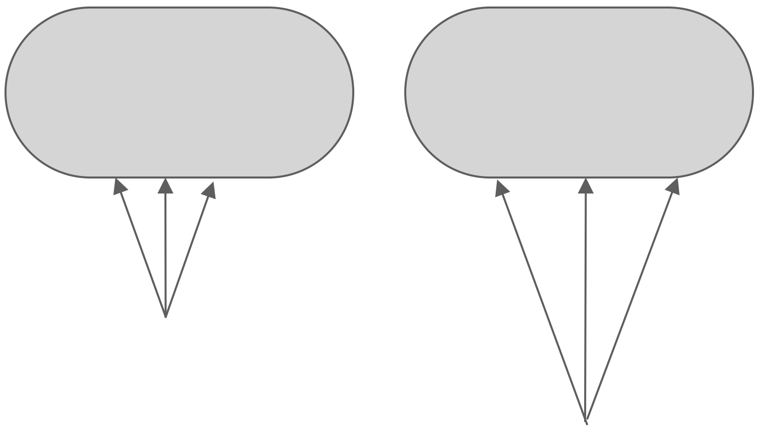
Wang, Qianqian, et al. "Ibrnet: Learning multi-view image-based rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

# Related Work (i)

- **The original Splatting Algorithm**



Aliasing



Sampling rate and depth

**→ Inaccurate visibility information**

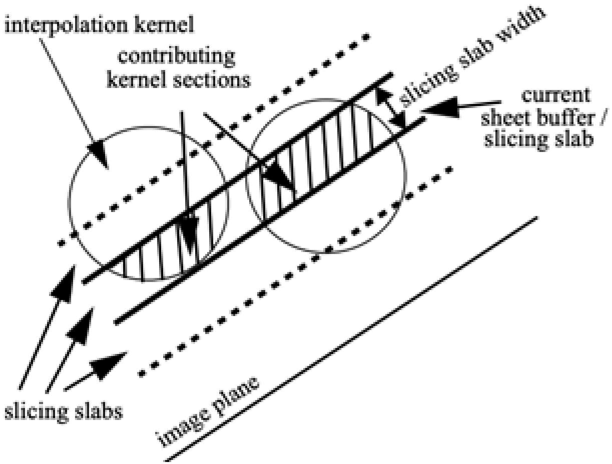
**→ Does not deal with sampling rate changes (Depth)**

https://www.ece.iastate.edu/~namrata/EE424/l1.pdf

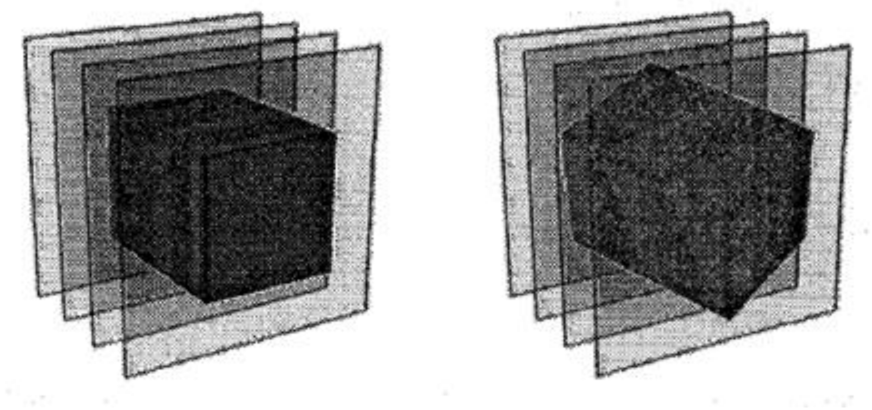
# Related Work (ii)

- **Slice-based volume rendering**

Uses axis-aligned sheet buffer to render volumes at different depths




Eliminating Popping Artifacts in Sheet Buffer-Based Splatting (VIS 98)



Direct Volume Rendering with Shading via Three-Dimensional Textures (VIS 96)

Wang, Qianqian, et al. "Ibrnet: Learning multi-view image-based rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

# Preliminaries

# Preliminaries

- **Volume rendering**

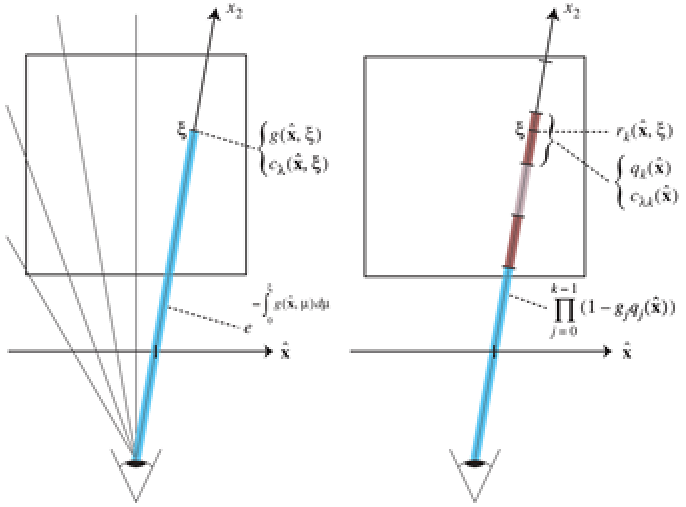
$$I_\lambda(\hat{\mathbf{x}}) = \int_0^L c_\lambda(\hat{\mathbf{x}}, \xi) g(\hat{\mathbf{x}}, \xi) e^{-\int_0^\xi g(\hat{\mathbf{x}}, \mu)\, d\mu}\, d\xi$$

g - extinction function (density)
c - emission coefficient (radiance)

$$g(\mathbf{x}) = \sum_k g_k r_k(\mathbf{x})$$

weighted sum of coefficients $g_k$ and reconstruction kernels $r_k(\mathbf{x})$

$$I_\lambda(\hat{\mathbf{x}}) = \sum_k \left( \int_0^L c_\lambda(\hat{\mathbf{x}}, \xi) g_k r_k(\hat{\mathbf{x}}, \xi) \prod_j e^{-g_j \int_0^\xi r_j(\hat{\mathbf{x}}, \mu)\, d\mu}\, d\xi \right).$$



# Preliminaries

- **Volume rendering**

**Assumptions:**

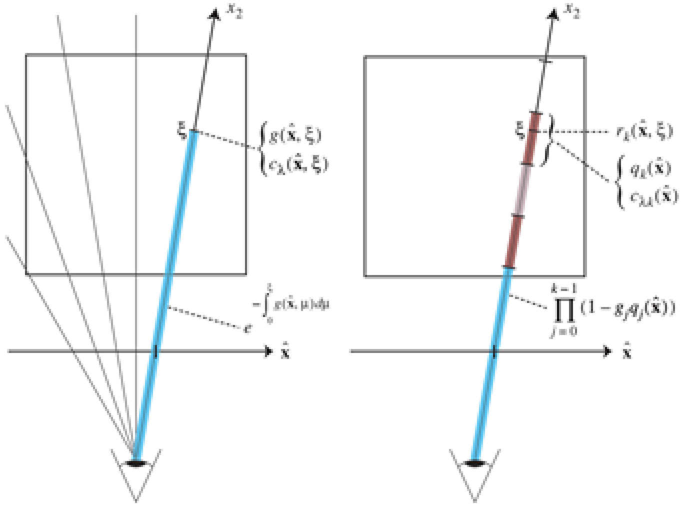
- Constant emission coefficient $c_{\lambda k}(\hat{\mathbf{x}}) = c_\lambda(\hat{\mathbf{x}}, x_2)$
- Taylor expansion of exponential function $e^x \approx 1 - x$

$$I_\lambda(\hat{\mathbf{x}}) = \sum_k c_{\lambda k}(\hat{\mathbf{x}}) g_k q_k(\hat{\mathbf{x}}) \prod_{j=0}^{k-1} (1 - g_j q_j(\hat{\mathbf{x}}))$$

$$q_k(\hat{\mathbf{x}}) = \int_{\mathbb{R}} r_k(\hat{\mathbf{x}}, x_2)\, dx_2.$$

Footprint function: Integrated reconstruction kernel

- Contribution of a 3D kernel to each point on the image plane



# Method

# Proposed Method (i)

- **Aliasing in Volume Splatting**

$$(I_\lambda \otimes h)(\hat{\mathbf{x}}) = \int_{\mathbb{R}^2} \sum_k c_{\lambda k}(\eta) g_k q_k(\eta)$$

$$\prod_{j=0}^{k-1} (1 - g_j q_j(\eta))\, h(\hat{\mathbf{x}} - \eta)\, d\eta.$$

Apply a low pass filter $h(\hat{\mathbf{x}})$

# Proposed Method (i)

- **Aliasing in Volume Splatting**

$$(I_\lambda \otimes h)(\hat{\mathbf{x}}) = \int_{\mathbb{R}^2} \sum_k c_{\lambda k}(\eta) g_k q_k(\eta) \prod_{j=0}^{k-1} (1 - g_j q_j(\eta))\, h(\hat{\mathbf{x}} - \eta)\, d\eta.$$

Apply a low pass filter $h(\hat{\mathbf{x}})$

**Assumptions:**

- Constant emission coefficient in the support area

  $c_{\lambda k}(\hat{\mathbf{x}}) \approx c_{\lambda k}$

- Constant attenuation factor (Transmittance)

  $\prod_{j=0}^{k-1} (1 - g_j q_j(\hat{\mathbf{x}})) \approx o_k$

# Proposed Method (i)

- **Aliasing in Volume Splatting**

$$(I_\lambda \otimes h)(\hat{\mathbf{x}}) = \int_{\mathbb{R}^2} \sum_k c_{\lambda k}(\eta) g_k q_k(\eta) \prod_{j=0}^{k-1} (1 - g_j q_j(\eta))\, h(\hat{\mathbf{x}} - \eta)\, d\eta.$$

Apply a low pass filter $h(\hat{\mathbf{x}})$

**Assumptions:**

- Constant emission coefficient in the support area

  $c_{\lambda k}(\hat{\mathbf{x}}) \approx c_{\lambda k}$

- Constant attenuation factor (Transmittance)

  $\prod_{j=0}^{k-1} (1 - g_j q_j(\hat{\mathbf{x}})) \approx o_k$

$$\begin{aligned}(I_\lambda \otimes h)(\hat{\mathbf{x}}) &\approx \sum_k c_{\lambda k} o_k g_k \int_{\mathbb{R}^2} q_k(\eta) h(\hat{\mathbf{x}} - \eta)\, d\eta \\ &= \sum_k c_{\lambda k} o_k g_k (q_k \otimes h)(\hat{\mathbf{x}}).\end{aligned}$$

Band-limiting applies only to the ***footprint function q***

$$\boxed{\rho_k(\hat{\mathbf{x}}) = (q_k \otimes h)(\hat{\mathbf{x}})}$$

Resampling filter

# Proposed Method (ii)

- **Elliptical Gaussian Kernels**

$r_k(\mathbf{x})$ Reconstruction kernel: **Elliptical Gaussian**

$$\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{MVM}^T}(\mathbf{u} - \Phi(\mathbf{p})).$$

# Proposed Method (ii)

- **Elliptical Gaussian Kernels**

$r_k(\mathbf{x})$ Reconstruction kernel: **Elliptical Gaussian**

$$\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{MVM}^T}(\mathbf{u} - \Phi(\mathbf{p})).$$

1. Closed under affine transformation

$$\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{MVM}^T}(\mathbf{u} - \Phi(\mathbf{p})).$$

# Proposed Method (ii)

- **Elliptical Gaussian Kernels**

$r_k(\mathbf{x})$ Reconstruction kernel: **Elliptical Gaussian** $\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{M}\mathbf{V}\mathbf{M}^T}(\mathbf{u} - \Phi(\mathbf{p})).$

1. Closed under affine transformation

$$\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{M}\mathbf{V}\mathbf{M}^T}(\mathbf{u} - \Phi(\mathbf{p})).$$

2. Closed under convolution

$$(\mathcal{G}_{\mathbf{V}} \otimes \mathcal{G}_{\mathbf{Y}})(\mathbf{x} - \mathbf{p}) = \mathcal{G}_{\mathbf{V}+\mathbf{Y}}(\mathbf{x} - \mathbf{p}).$$

# Proposed Method (ii)

- **Elliptical Gaussian Kernels**

$r_k(\mathbf{x})$ Reconstruction kernel: **Elliptical Gaussian** $\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{MVM}^T}(\mathbf{u} - \Phi(\mathbf{p})).$

1. Closed under affine transformation

$$\mathcal{G}_{\mathbf{V}}(\Phi^{-1}(\mathbf{u}) - \mathbf{p}) = \frac{1}{|\mathbf{M}^{-1}|}\mathcal{G}_{\mathbf{MVM}^T}(\mathbf{u} - \Phi(\mathbf{p})).$$

2. Closed under convolution

$$(\mathcal{G}_{\mathbf{V}} \otimes \mathcal{G}_{\mathbf{Y}})(\mathbf{x} - \mathbf{p}) = \mathcal{G}_{\mathbf{V}+\mathbf{Y}}(\mathbf{x} - \mathbf{p}).$$

3. 3D integration along an axis yields 2D Gaussian

$$\int_{\mathbb{R}} \mathcal{G}_{\mathbf{V}}(\mathbf{x} - \mathbf{p})\, dx_2 = \mathcal{G}_{\hat{\mathbf{V}}}(\hat{\mathbf{x}} - \hat{\mathbf{p}}), \qquad \mathbf{V} = \begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix} \Leftrightarrow \begin{pmatrix} a & b \\ b & d \end{pmatrix} = \hat{\mathbf{V}}$$

**We can map arbitrary 3D Gaussians in object to ray space**

# Proposed Method (iii)

- **Viewing Transformation**

**Object space → Camera space**

Object space kernel $r''_k(\mathbf{t}) = \mathcal{G}_{\mathbf{V}''_k}(\mathbf{t} - \mathbf{t}_k)$

Camera coordinates $\mathbf{u} = (u_0, u_1, u_2)^T$

Affine transformation $\varphi(\mathbf{t}) = \mathbf{W}\mathbf{t} + \mathbf{d}$ $\quad \mathbf{u} = \varphi(\mathbf{t})$

$$\mathcal{G}_{\mathbf{V}''_k}(\varphi^{-1}(\mathbf{u}) - \mathbf{t}_k) = \frac{1}{|\mathbf{W}^{-1}|}\mathcal{G}_{\mathbf{V}'_k}(\mathbf{u} - \mathbf{u}_k) = r'_k(\mathbf{u}) \qquad \mathbf{V}'_k = \mathbf{W}\mathbf{V}''_k\mathbf{W}^T$$

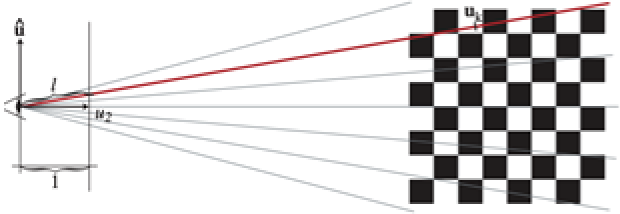
# Proposed Method (iv)

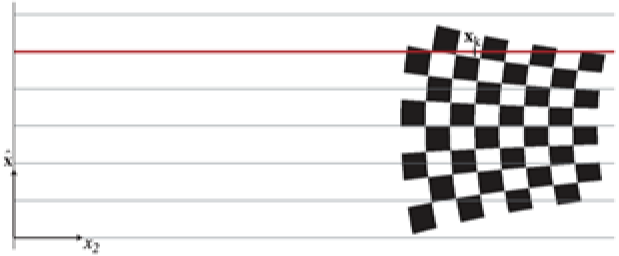
- Projective Transformation

Camera space → Ray space

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \mathbf{m}(\mathbf{u}) = \begin{pmatrix} u_0/u_2 \\ u_1/u_2 \\ \|(u_0, u_1, u_2)^T\| \end{pmatrix}$$

$$\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} = \mathbf{m}^{-1}(\mathbf{x}) = \begin{pmatrix} x_0/l \cdot x_2 \\ x_1/l \cdot x_2 \\ 1/l \cdot x_2 \end{pmatrix}, \text{ where } l = \|(x_0, x_1, 1)^T\|.$$





→ **Not affine transformation**

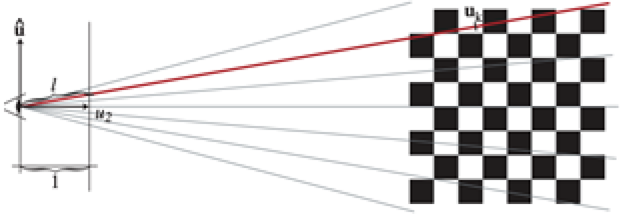
# Proposed Method (iv)

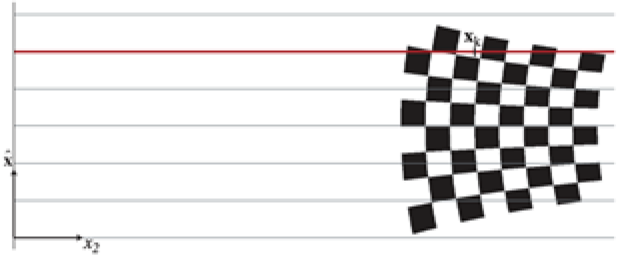
- Projective Transformation

**Camera space → Ray space**

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \mathbf{m}(\mathbf{u}) = \begin{pmatrix} u_0/u_2 \\ u_1/u_2 \\ \|(u_0,u_1,u_2)^T\| \end{pmatrix}$$

$$\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} = \mathbf{m}^{-1}(\mathbf{x}) = \begin{pmatrix} x_0/l \cdot x_2 \\ x_1/l \cdot x_2 \\ 1/l \cdot x_2 \end{pmatrix}, \text{ where } l = \|(x_0,x_1,1)^T\|.$$





**→ Not affine transformation**

Taylor expansion of $\mathbf{m}$ at the point $\mathbf{u}_k$:

$$\mathbf{m}_{\mathbf{u}_k}(\mathbf{u}) = \mathbf{x}_k + \mathbf{J}_{\mathbf{u}_k} \cdot (\mathbf{u} - \mathbf{u}_k),$$

where $\mathbf{x}_k = \mathbf{m}(\mathbf{u}_k)$ is the center of a Gaussian in ray space. $\mathbf{J}_{\mathbf{u}_k} = \frac{\partial \mathbf{m}}{\partial \mathbf{u}}(\mathbf{u}_k)$

# Proposed Method (iv)

- **Projective Transformation**

$$
\begin{aligned}
r_k(\mathbf{x}) &= \frac{1}{|\mathbf{W}^{-1}|}\mathcal{G}_{\mathbf{V}'_k}(\mathbf{m}^{-1}(\mathbf{x}) - \mathbf{u}_k) \\
&= \frac{1}{|\mathbf{W}^{-1}||\mathbf{J}^{-1}|}\mathcal{G}_{\mathbf{V}_k}(\mathbf{x} - \mathbf{x}_k),
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{V}_k &= \mathbf{J}\mathbf{V}'_k\mathbf{J}^T \\
&= \mathbf{J}\mathbf{W}\mathbf{V}''_k\mathbf{W}^T\mathbf{J}^T.
\end{aligned}
$$

# Proposed Method (v)

- Integration and Band-Limiting

**Footprint function**

$$q_k(\hat{\mathbf{x}}) = \int_{\mathbb{R}} r_k(\hat{\mathbf{x}}, x_2)\, dx_2.$$

$$\begin{aligned} q_k(\hat{\mathbf{x}}) &= \int_{\mathbb{R}} \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\mathbf{V}_k}(\hat{\mathbf{x}} - \hat{\mathbf{x}}_k, x_2 - x_{k2})\, dx_2 \\ &= \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\hat{\mathbf{V}}_k}(\hat{\mathbf{x}} - \hat{\mathbf{x}}_k), \end{aligned}$$

# Proposed Method (v)

- **Integration and Band-Limiting**

### Footprint function

$$q_k(\hat{\mathbf{x}}) = \int_{\mathbb{R}} r_k(\hat{\mathbf{x}}, x_2)\, dx_2.$$

$$\begin{aligned} q_k(\hat{\mathbf{x}}) &= \int_{\mathbb{R}} \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\mathbf{V}_k}(\hat{\mathbf{x}} - \hat{\mathbf{x}}_k, x_2 - x_{k2})\, dx_2 \\ &= \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\hat{\mathbf{V}}_k}(\hat{\mathbf{x}} - \hat{\mathbf{x}}_k), \end{aligned}$$

### EWA volume resampling filter

$$\begin{aligned} (I_\lambda \otimes h)(\hat{\mathbf{x}}) &\approx \sum_k c_{\lambda k} o_k g_k \int_{\mathbb{R}^2} q_k(\eta) h(\hat{\mathbf{x}} - \eta)\, d\eta \\ &= \sum_k c_{\lambda k} o_k g_k (q_k \otimes h)(\hat{\mathbf{x}}). \end{aligned}$$

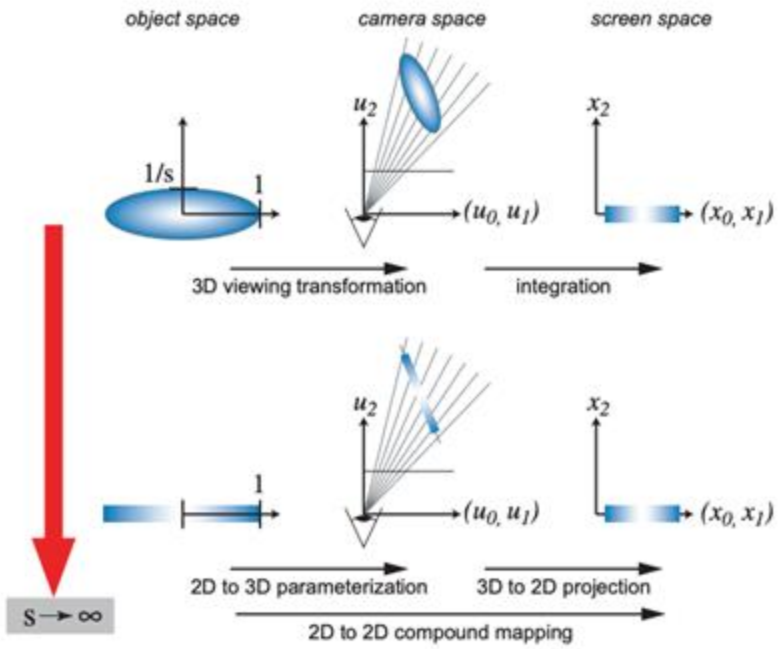
$$\begin{aligned} \rho_k(\hat{\mathbf{x}}) &= (q_k \otimes h)(\hat{\mathbf{x}}) \\ &= \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} (\mathcal{G}_{\hat{\mathbf{V}}_k} \otimes \mathcal{G}_{\mathbf{V}^h})(\hat{\mathbf{x}} - \hat{\mathbf{x}}_k) \\ &= \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\hat{\mathbf{V}}_k + \mathbf{V}^h}(\hat{\mathbf{x}} - \hat{\mathbf{x}}_k). \end{aligned}$$

# Proposed Method (vi)

- **Reduction from Volume to Surface Reconstruction Kernels**

Flat Gaussian: Reconstruction kernel for isosurface reconstruction

$$\mathbf{V}'' = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{1}{s^2} \end{pmatrix}$$



# Proposed Method (vi)

- **Reduction from Volume to Surface Reconstruction Kernels**

$$\mathbf{V} = \mathbf{J}\mathbf{W}\mathbf{V}''\mathbf{W}^T\mathbf{J}^T = \mathbf{T}^{3D}\mathbf{V}''\mathbf{T}^{3D^T}. \quad \mathbf{T}^{3D} = \mathbf{J}\mathbf{W}$$

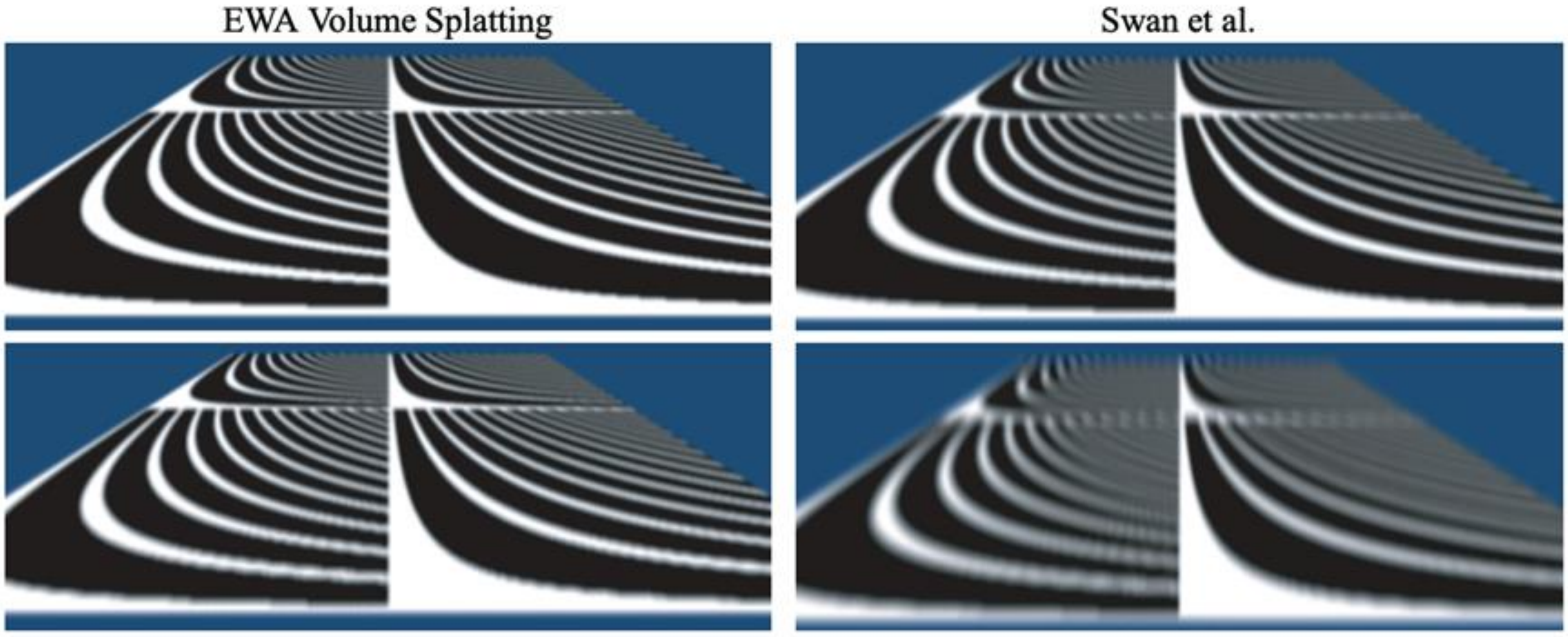
Hence, the elements $v_{ij}$ of $\mathbf{V}$ are given by:

$$\begin{aligned}
v_{00} &= t_{00}^2 + t_{01}^2 + \frac{t_{02}^2}{s^2} \\
v_{01} &= v_{10} = t_{00}t_{10} + t_{01}t_{11} + \frac{t_{02}t_{12}}{s^2} \\
v_{02} &= v_{20} = t_{00}t_{20} + t_{01}t_{21} + \frac{t_{02}t_{22}}{s^2} \\
v_{11} &= t_{10}^2 + t_{11}^2 + \frac{t_{12}^2}{s^2} \\
v_{12} &= v_{21} = t_{10}t_{20} + t_{11}t_{21} + \frac{t_{12}t_{22}}{s^2} \\
v_{22} &= t_{20}^2 + t_{21}^2 + \frac{t_{22}^2}{s^2},
\end{aligned}$$

# Experiments

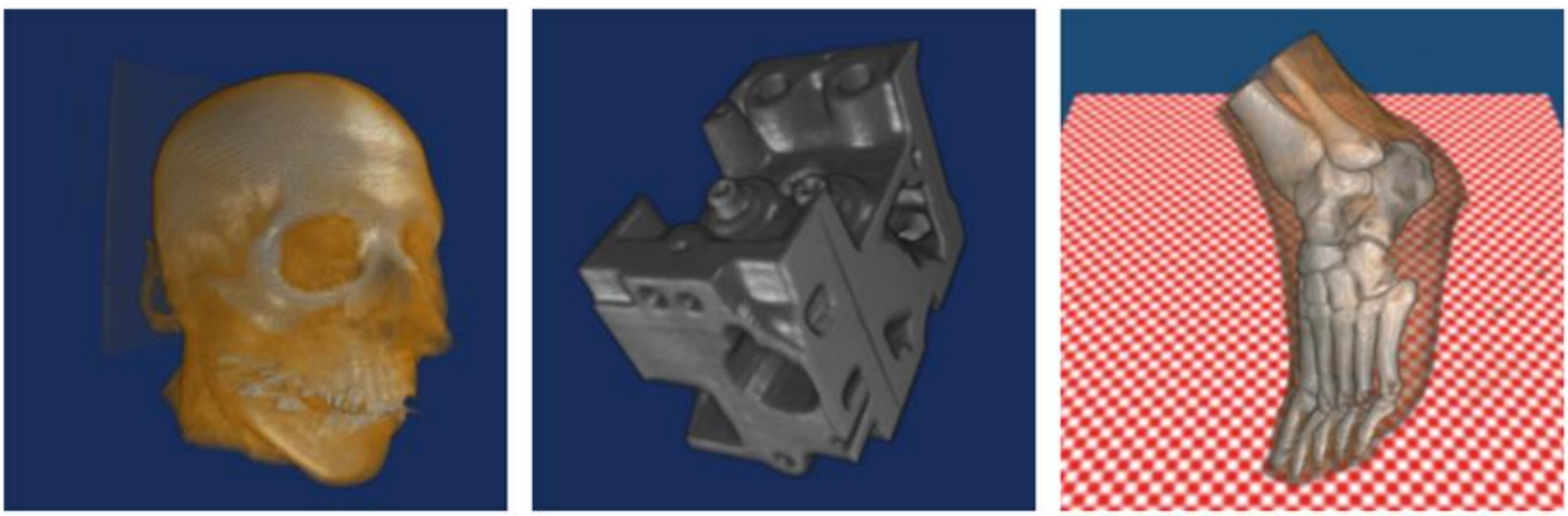
# Evaluation: Qualitative

**Baseline:** Swan *et al* "An AntiAliasing Technique for Splatting" (Uniform kernel)



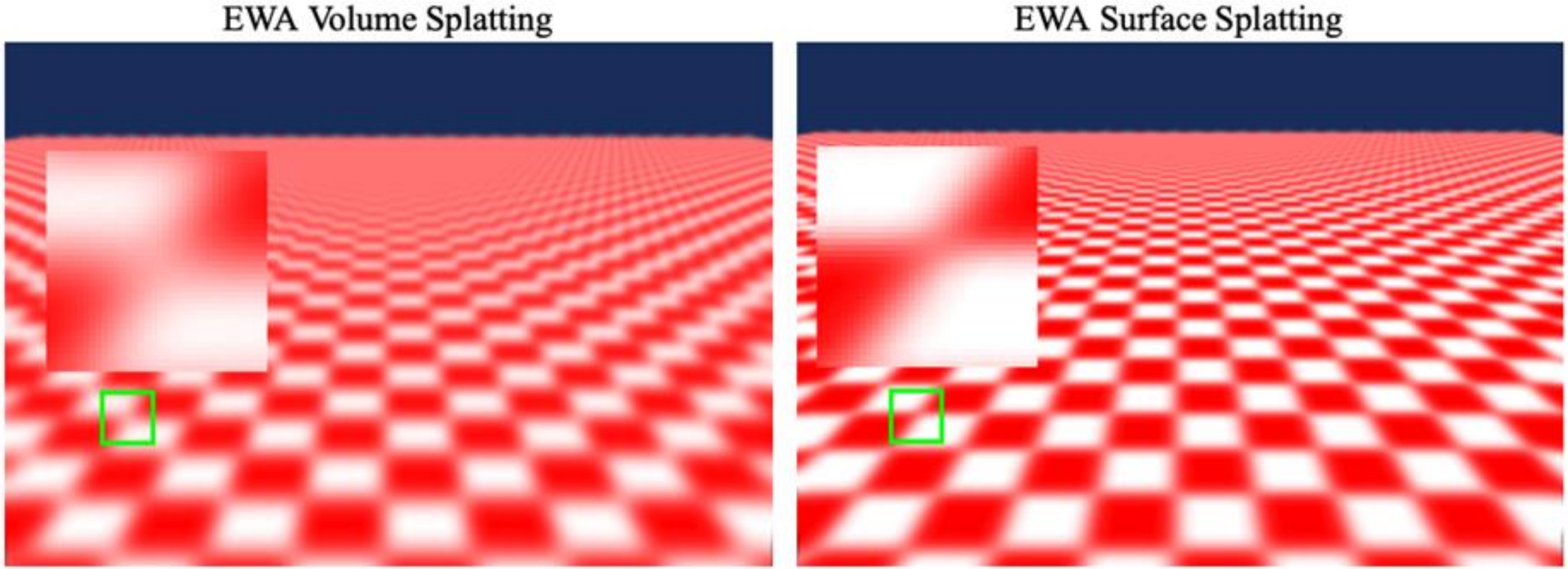
# Evaluation: Qualitative

EWA Volume Rendering on Semi-transparent object



# Evaluation: Qualitative

EWA Volume/Surface Splatting



# Conclusion

**Antialising for splatting algorithms:** Gaussian reconstruction kernel with a low pass filter

**Render irregular volume data:** Elliptical reconstruction kernel

**Surface reconstruction:** Flat Gaussian kernel