# A Generative Model for Volume Rendering

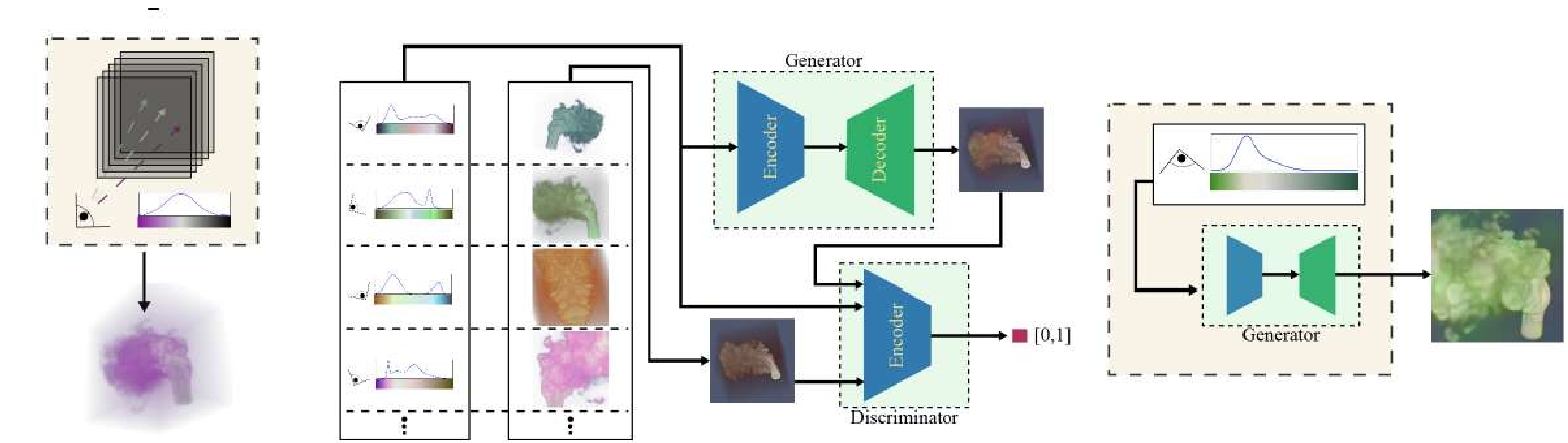
Matthew Berger, Jixian Li, and Joshua A. Levine

University of Arizona

# Introduction

- Technique to synthesize and analyze volume-rendered images
- Using Generative Adversarial Model (GAN) model
- Dataset: volume rendered images, conditioned on
  - 1) viewpoint
  - 2) transfer function for opacity and color
- Byproduct
  - Guide the user in transfer function editing
  - Transform transfer function into a view-invariant latent space

# Overall Process

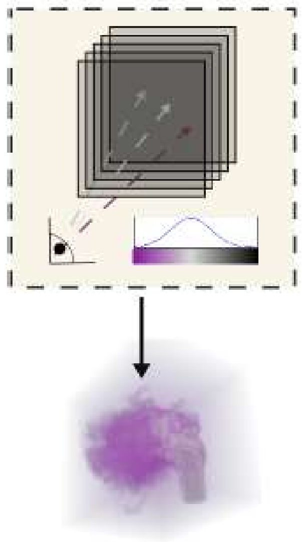



(a) Volume Rendering (b) Training data (View, TF, Image) (c) GAN Training (d) Rendering via Generator

# Traditional Volume Renedering

- A basic algorithm
- Input parameter:
  - Input volumetric scalar field
  - Viewpoint
  - Opacity and color transfer function
- Composting

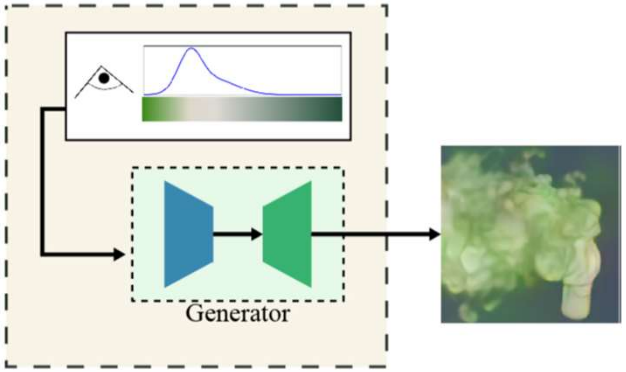
$$\mathbf{I}(x,y)_{i+1} = \mathbf{I}(x,y)_i + (1-\tau_i')\mathbf{c}_i\tau_i$$
$$\tau_{i+1}' = \tau_i' + (1-\tau_i')\tau_i,$$



(a) Volume Rendering

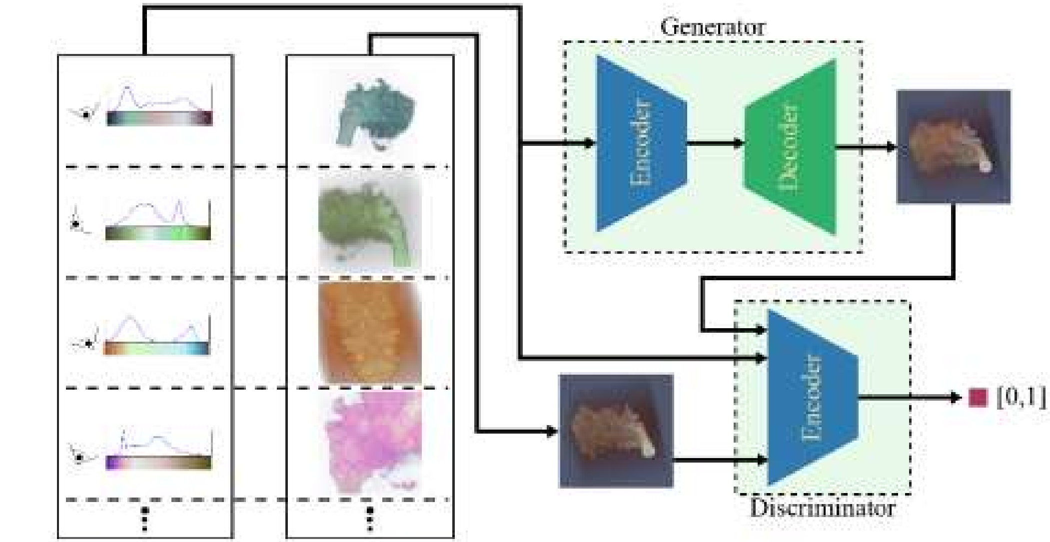
# GAN based volume rendering

- This paper instead view volume rendering as a purely computational process
    - Using a generative model to approximate the function (see the right figure)
- Inputs:
    - viewpoints and TFs
- Outputs:
    - volume rendered images



# Traing process

- Data preparation
  - Viewpoint
    - Randomly sampled
  - Opacity TF
    - Sample from a Gaussian mixture model
    - Randomly sample number of modes in the GMM (1-5)
    - Each model -> Gaussian model with a random mean and std
  - Color TF
    - First sample random color at opacity TF GMM means & scalar value global optimal
    - Then piecewise lerp



(b) Training data (View, TF, Image) (c) GAN Training

# Network training

- Use GAN as a model
  - input of G: viewpoint and transfer function
    - Viewpoint information
      - nv parameters $v \in R^{nv}$
      - nv = 5 – for azimuth, elevation, in-plane rotation, and distance to the camera
    - TF information
      - $to \in R^{nt}$
      - $tc \in R^{3nt}$
      - nt = 256, which is the sample step
  - Output of G: a color image $I \in R3wh$

- input of D: viewpoint, transfer function & an image

- Output of D: score 0-1 – whether the image is a true volume-rendering or is fake one produced by G

# Network training (cont'd)

- adversarial loss in a GAN

$$L_{adv}(G,D) = \mathbb{E}_{\mathbf{I},\mathbf{w}\sim p_{data}} \log(D(\mathbf{w},\mathbf{I})) + \mathbb{E}_{\mathbf{w}\sim p_{vis}} \log(D(\mathbf{w},G(\mathbf{w}))), \quad (4)$$

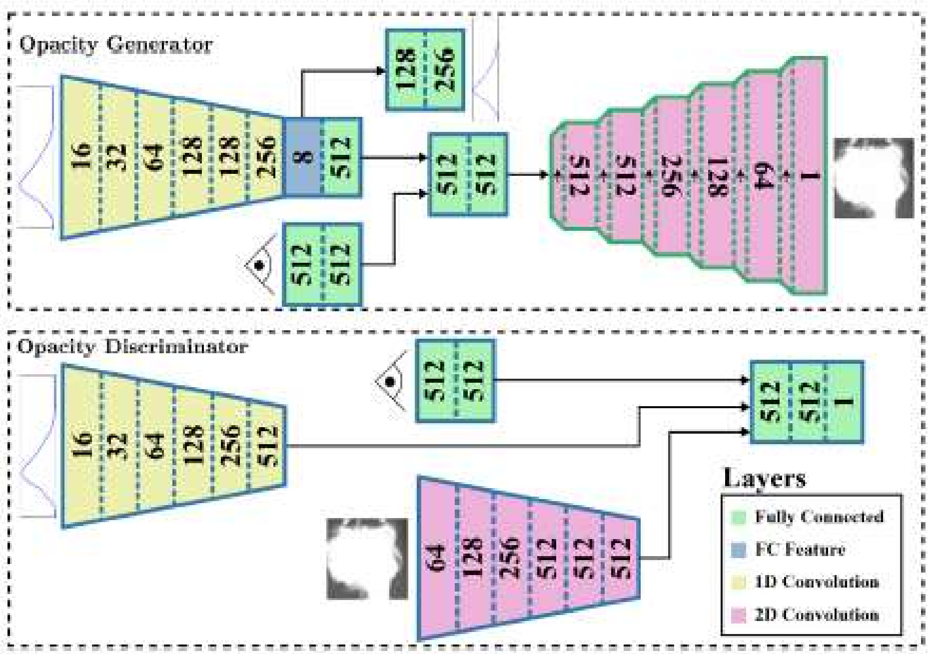
- generator and discriminator compete in a min-max game

$$\min_G \max_D L_{adv}(G,D).$$

# Network design

- Generating 256x256 image – difficult
  - Only at that time, InSituNet generate 256x256 image in one stage
- Solution: break the problem into two simpler generation tasks
  - Both represented as separate GANs

# Opacity GAN

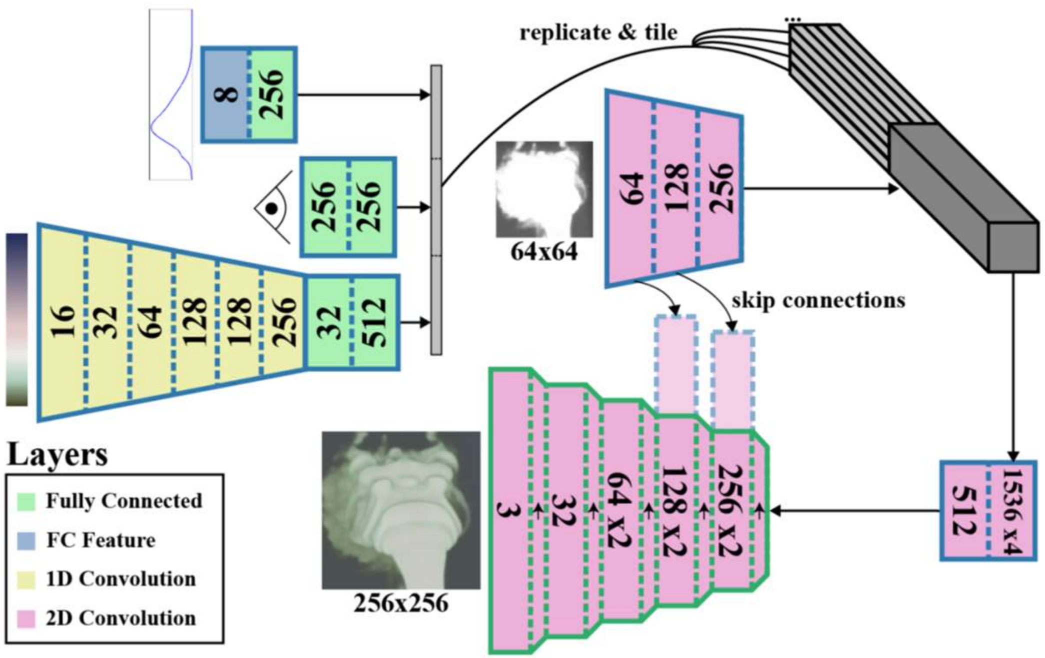



Network structure

- Objective
  - Combine adversarial loss with autoencoder loss

$$\min_G \max_D L_{adv}(G,D) + \|G_{dec}(G_{enc}(\mathbf{t}_o)) - \mathbf{t}_o\|_2^2,$$

# Opacity-to-Color Translation GAN




The generator for the opacity-to-color translation GAN

- discriminator – very similar to the Opacity GAN's discriminator
  - Main addition: inclusion of the color TF transformation

- Objective:

$$\min_{G}\max_{D} L_{adv}(G,D) + \lambda\|G(\mathbf{v},\mathbf{t}_o,\mathbf{t}_c) - I\|_1,$$

  - I – the ground truth image
  - λ weights the importance of the $l_1$ loss.

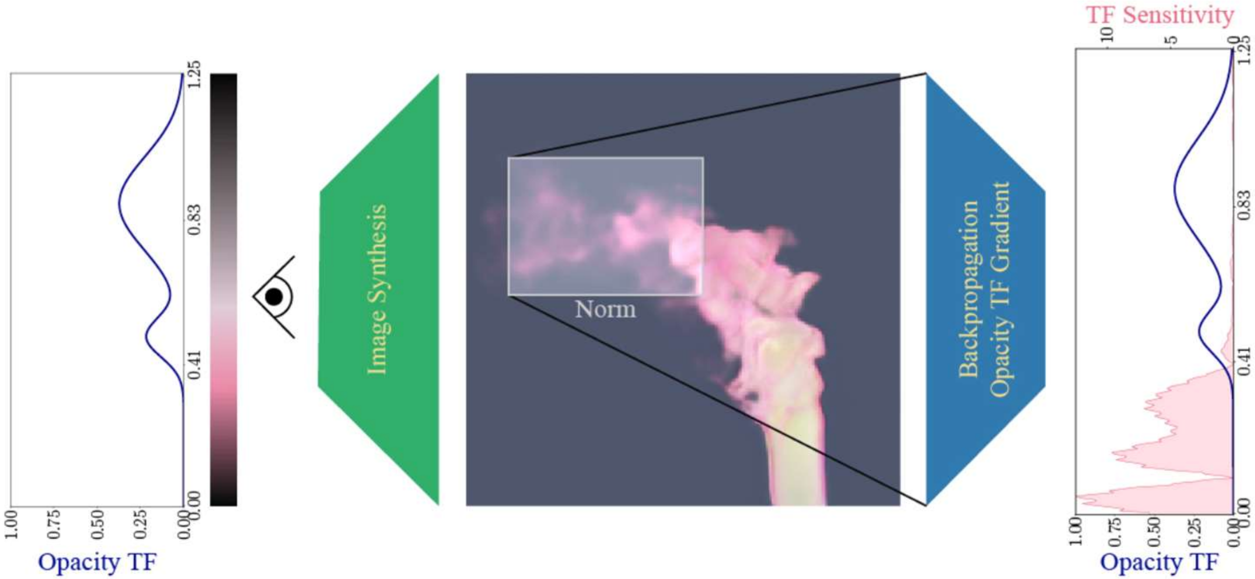
# Applications

- $1^{st}$ application: transfer function sensitivity and exploration
- transfer function sensitivity σ : R -> $R^{256}$

$$\sigma(R) = \nabla_{\mathbf{t}_o} \| G_t((G_o(\mathbf{v}, \mathbf{t}_o)), \mathbf{v}, \mathbf{t}_o, \mathbf{t}_c) \|_R,$$

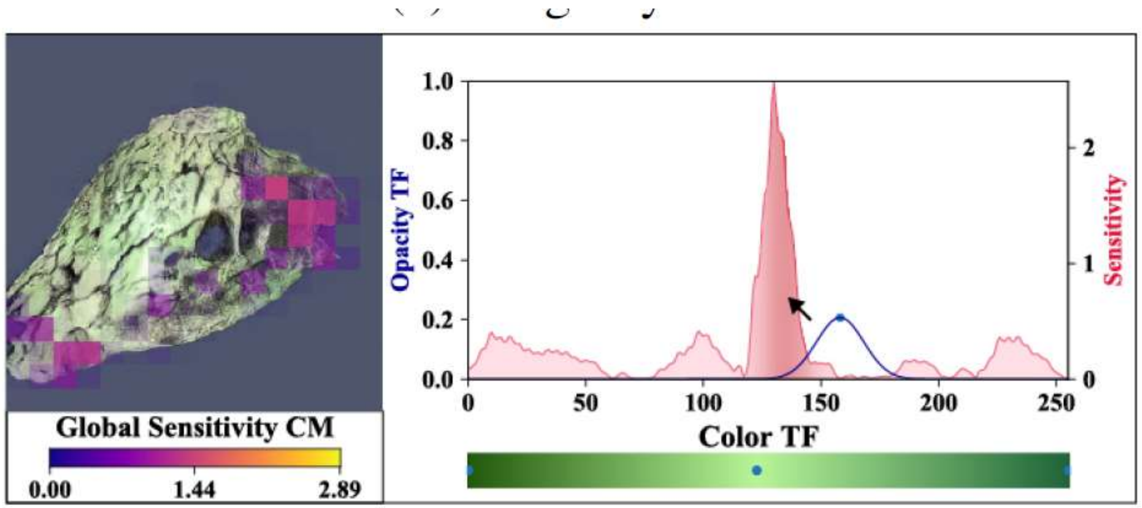
  - $1^{st}$ feed input parameters into the network
  - $2^{nd}$ compute the $l_2$ norm of a region R
  - $3^{rd}$ perform backpropagation to compute opacity TF gradient

# Visulization techniques



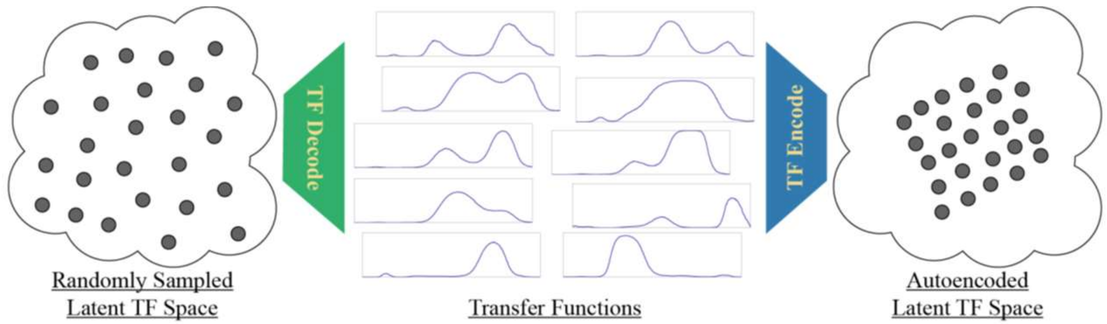

- Region Sensitivity Plots
  - plot σ directly with the opacity TF
- Example: right-hand side of the figure above
  - Region Sensitivity Plot for a user-specified region

# Visulization techniques




- Scalar Value Sensitivity Field
  - 1st define a grid resolution r & divide the image into r x r blocks
  - 2nd $$\sigma(R) = \nabla_{\mathbf{t}_o} \| G_t((G_o(\mathbf{v}, \mathbf{t}_o)), \mathbf{v}, \mathbf{t}_o, \mathbf{t}_c) \|_R,$$
    - Compute TF sensitivity for for each block
    - Produce 3-tensor S $\in$ $R^{256xrxr}$
- Example: left hand-side in the figure above

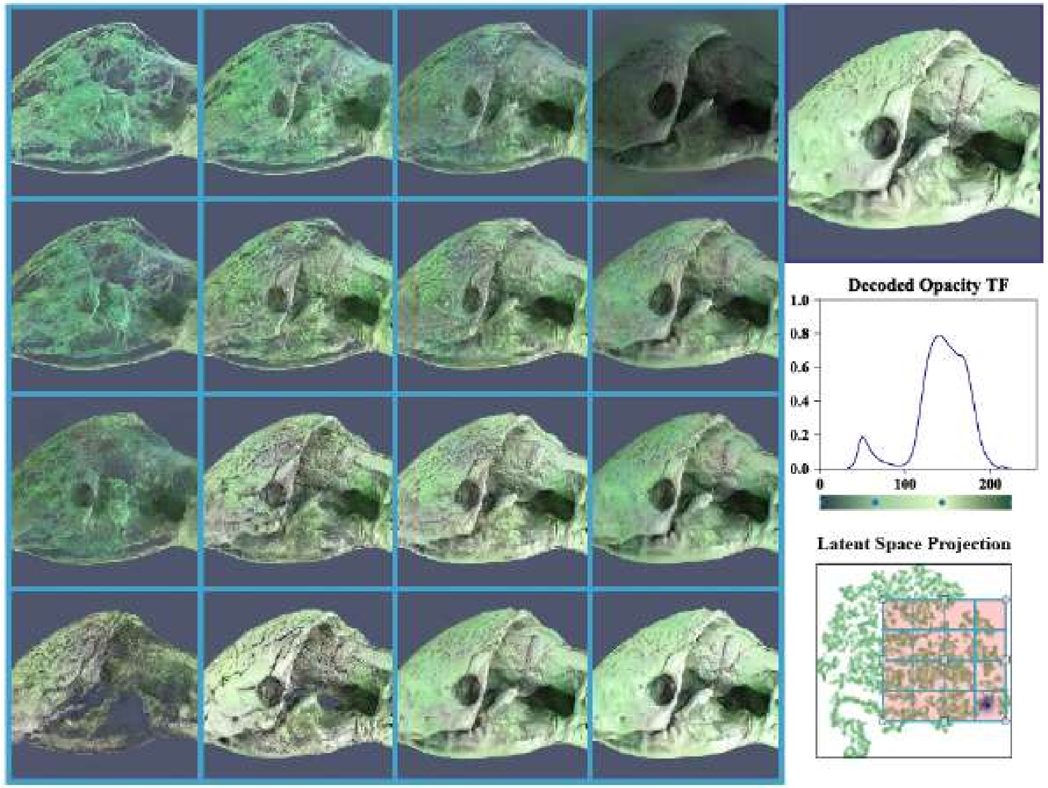
# Applicatons




- 2nd application: Exploring the Opacity TF Latent Space
- Make use of the 8-dim latent space for transfer function
- Steps:
  - 1. sample the latent space
    - Perfrom uniform sampling
    - Decode each sample to reconstruct a TF
    - Encode the set of TFs back into the latent space
  - 2. 2D Projection
    - Use t-SNE to project the latent vector into 2D space

# Exploring the Opacity TF Latent Space (cont'd)

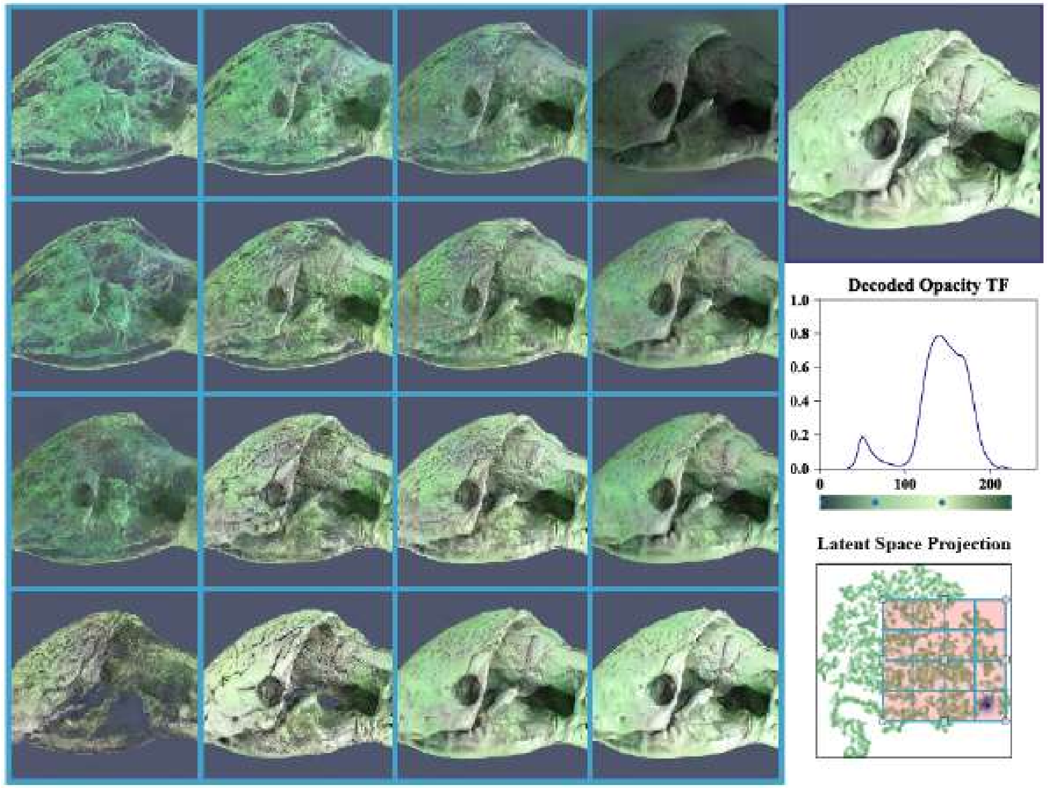
- Steps:
  - 3. Structured Latent Space Browsing
    - brush a 4 x 4 rectangular grid on the 2D projection
    - For a given cell, compute the mean & synthesize the image from the mean
    - Eg. Figure c (left)




(c) Transfer Function Latent Space Projection

# Exploring the Opacity TF Latent Space (cont'd)

- Steps:
  - 4. Latent Space Interpolation
    - For a given point in the 2D projection (highlighted in blue, lower left)
    - scattered data interpolation of latent opacity
    - synthesized image shown at the upper-right



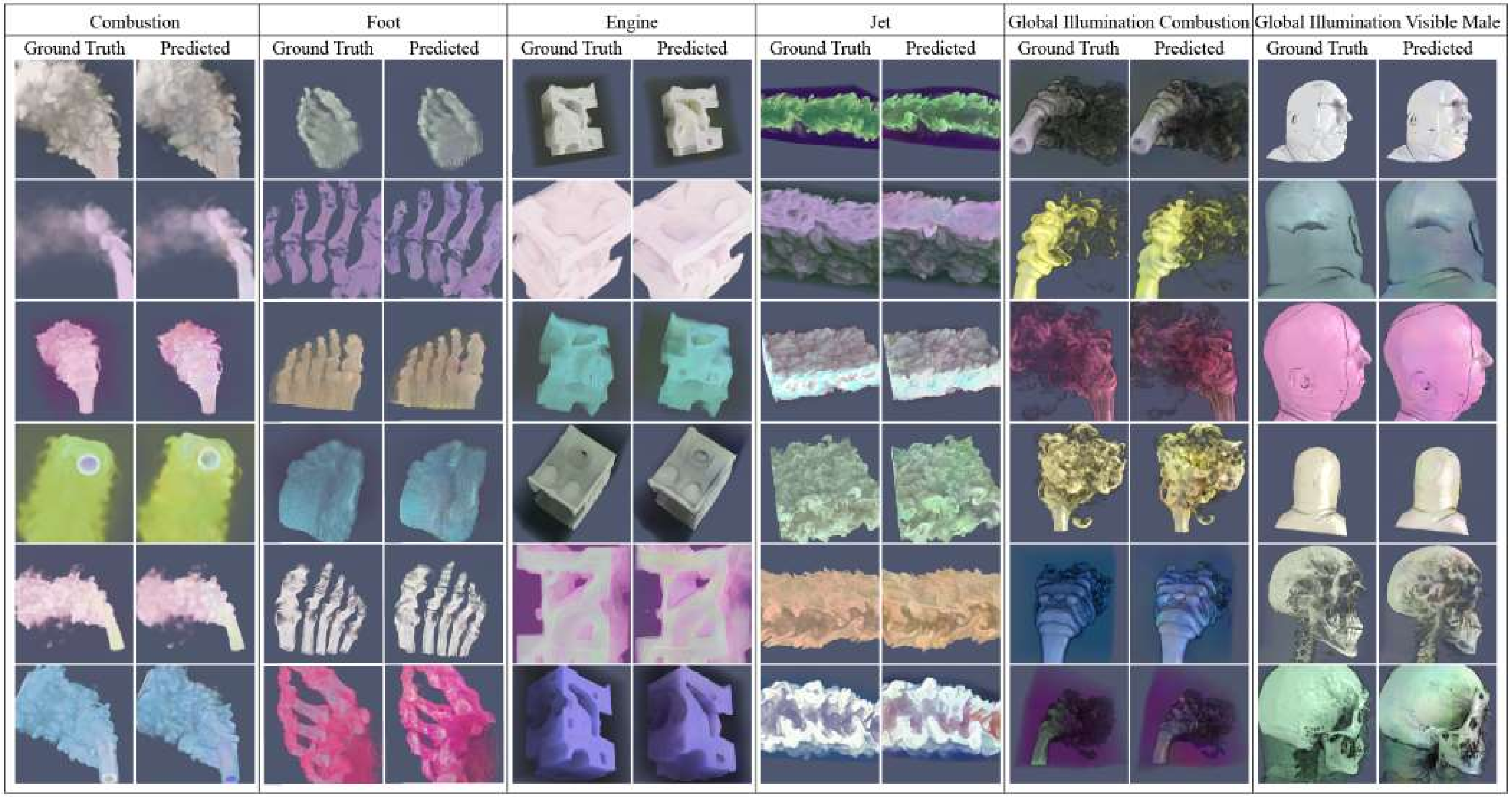

(c) Transfer Function Latent Space Projection

# experimental results

- Steps:
  - Dataset characteristics (left)
  - Quantitative evaluation of the model (right)

| Dataset | Resolution | Precision | Size (MB) | Rendering Model | Training Images Creation | Image RMSE | Color EMD |
|---|---|---|---|---|---|---|---|
| Combustion | $170 \times 160 \times 140$ | float | 15 | No Illumination | 2.7 hours | 0.046 | 0.011 |
| | | | | Direct Illumination | 5 hours | 0.060 | 0.011 |
| | | | | Global Illumination | 14 hours | 0.060 | 0.010 |
| Engine | $256 \times 256 \times 110$ | byte | 7 | No Illumination | 3 hours | 0.061 | 0.015 |
| Visible Male | $128 \times 256 \times 256$ | byte | 8 | Global Illumination | 14 hours | 0.075 | 0.013 |
| Foot | $256 \times 256 \times 256$ | byte | 16 | No Illumination | 3.3 hours | 0.064 | 0.017 |
| Jet | $768 \times 336 \times 512$ | float | 504 | No Illumination | 4 hours | 0.086 | 0.022 |
| Spathorynchus | $1024 \times 1024 \times 750$ | byte | 750 | Global Illumination | 5 days | 0.116 | 0.020 |

# experimental results



# Discussion

- Potential future work
  - Train the network in-situ
  - Design a network to learn from both time-varying and multivariant data
  - Improve the quality of results
    - Apply depth based measurements
  - Explore different ways of sampling views and TFs
    - maybe data-driven manner
  - explore different kinds of TFs
    - One direction is 2D transfer functions
  - let the GAN condition on the volume
  - Consider GAN to synthsize TFs rather than images