Volumetric Isosurface Rendering with Deep Learning-Based Super-Resolution

Sebastian Weiss , Mengyu Chu, Nils Thuerey and Rüdiger Westermann

Technical University of Munich

Motivation

- isosurface volume ray-casting
  - A efficient rendering method to render isosurface given a viewpoint
  - Workload of ray-casting linear in the number of pixels
    - i.e. Decrease the number of pixles -> decrease workload
    - Visual quality not satisfactory –> **super resolution by deep learning**.
  - Consider global illumination -> increase visual quality
    - E.g. AO (ambient occlusion)
    - **Problem:** calculate AO is time-consuming
    - AO can be generated from other stored information (eg. depth and normal maps)

Ambient occlusion

- Ambient light
  - In phong model, a fixed lighting **constant** to simulate the scattering of light
  - In reality, it should not be consansant
  - Eg. darkening creases, holes and surfaces that are close to each other
    - occluded by surrounding geometry -> light rays have less places to escape -> darker

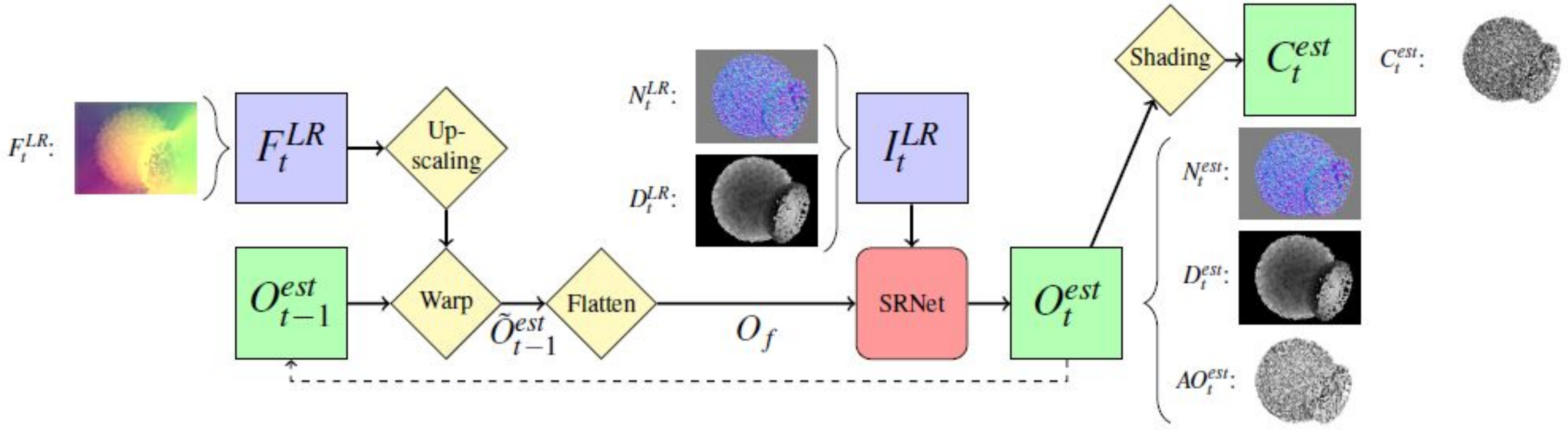
- AO: expensive
  - Reason: take surrounding geometry into account
- Paper: approximate by low-resolution stored information

Contribution

- 1. learning the upscaling of a low resolution sampling of an isosurface -> a higher resolution
  - By reconstruction of spatial detail and shading
  - Why?
    - More information; robust to lighting
- 2. infer AO in the high resolution w/o AO in the low resolution inputs
  - No need to simulate AO
- 3. add a motion loss -> maintain frame-to-frame coherence
- 4. perform a quality evaluation

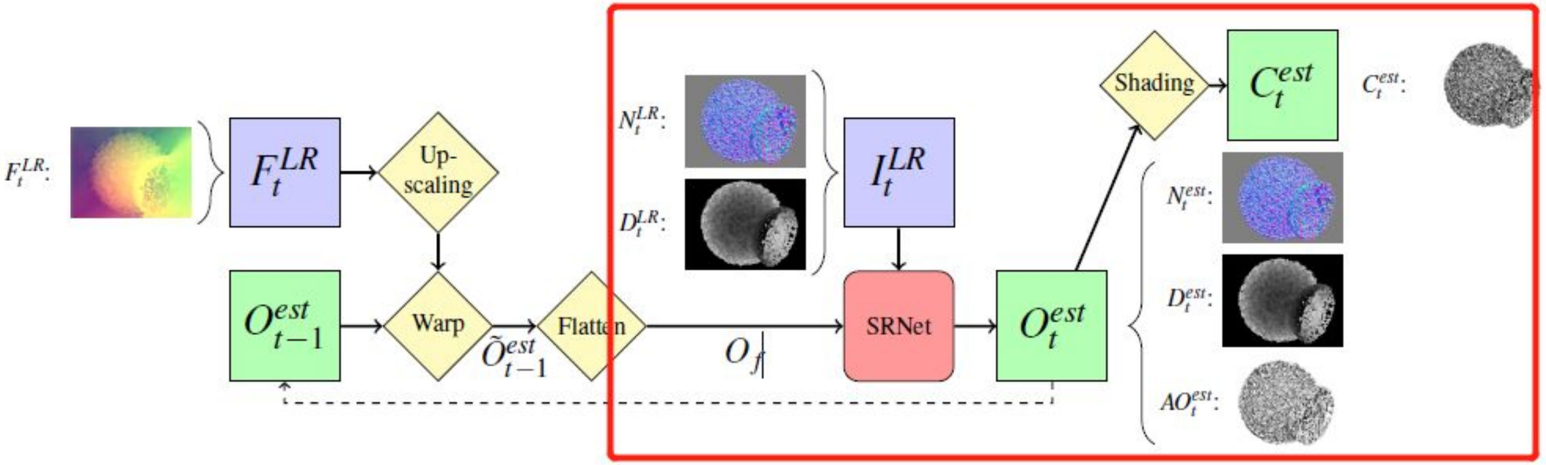
isosurface learning

Blue: low resolution inputs green: high resolution outputs
Yellow: fixed propressing steps Red: trained network



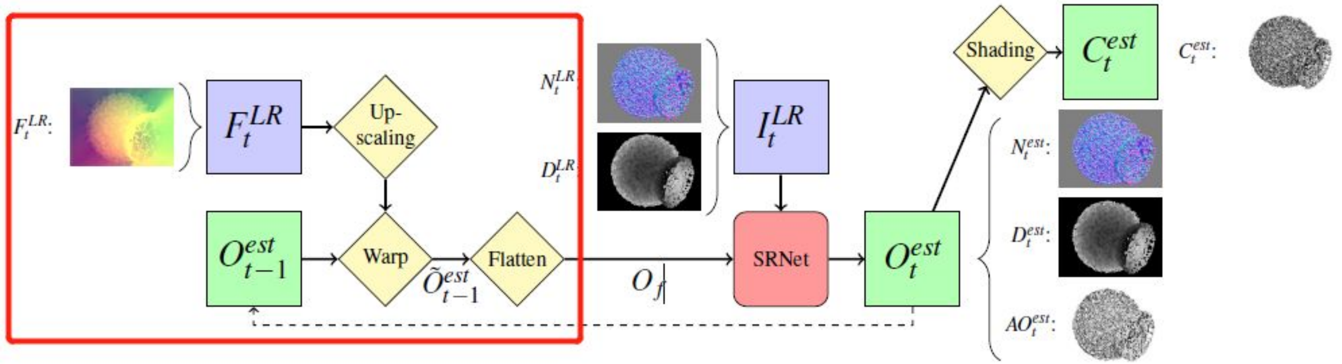
- Goal: perform 4 upscaling
  - i.e. from input images of size HxW to output images of size 4Hx4W
  - Input: LR depth and normal maps
  - Output: HR depth and normal maps, additional AO map
  - screen-space shading used for generating final color

Input data



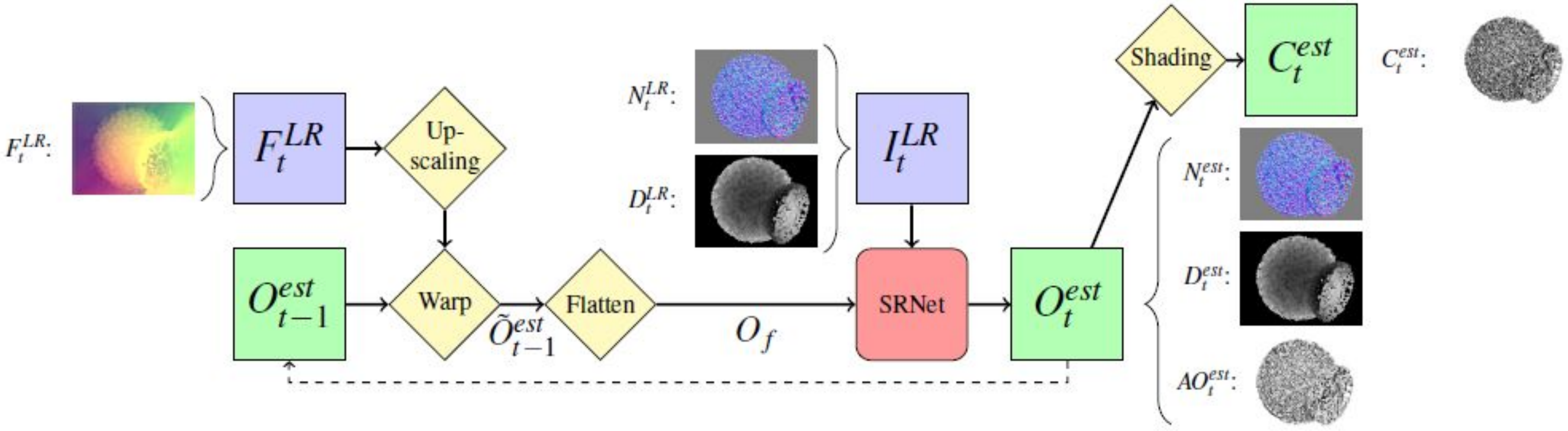
- Input maps generated via volumetric ray-casting
- low resolution input maps (size HxW)
  - $M_t^{LR} \in [-1, +1]^{H\,x\,W}$: binary input mask
    - Whether the isosurface hit or not
    - network learn continuous value, use these to smoothly blend the final color
  - $N_t^{LR} \in [-1, +1]^{3\,x\,H\,x\,W}$: normal maps
  - $D_t^{LR} \in [0, 1]^{H\,x\,W}$: depth maps
  - LR input to the network $I_t^{LR} := \{M_t^{LR}, N_t^{LR}, D_t^{LR}\} \in R^{5\,x\,H\,x\,W}$

Input data



- $F_t^{LR} \in$ [-1, +1]$^{2\ x\ H\ x\ W}$: A map of displacement vectors
  - The screen space-flow from the previous view to current view
  - Goal: minimize the deviation of current inferred HR map from the extrapolated previous one
  - Isosurface from different viewpoints: the same coordinate at **world space**, different at **screen space**, denote as x_t' and x_{t-1}'
  - Flow computed by $f_t \; := \; x_t' - x_{t-1}'$
  - Current flow field, up-scaling via bi-linear interpolation
  - Then use inverse flow vector to determine the target location

Input data



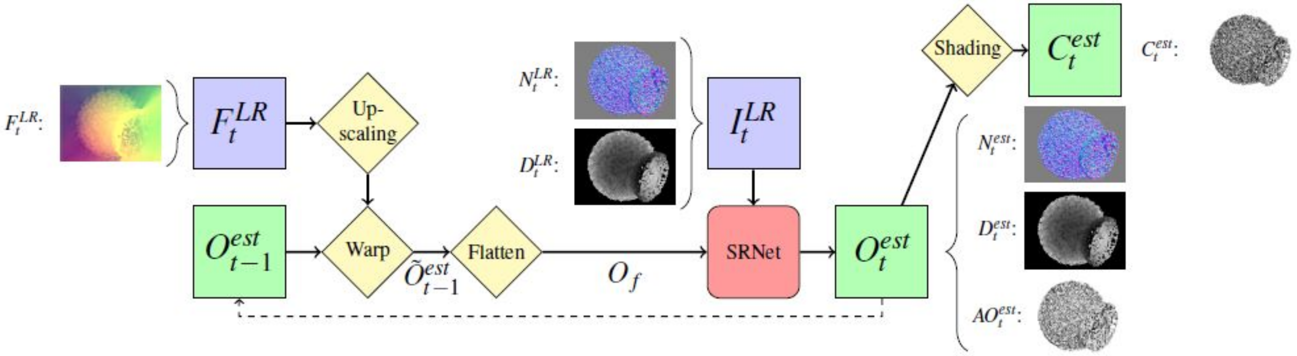
- High resolution input data
  - Used as GT in training process
  - The same as low resolution input + AO map $AO_t^{LR} \in [0, 1]^{4H\ x\ 4W}$
    - in AO map, 0 or 1 -> no or full occlusion
  - GT image
    - $O_t^{GT} := \{M_t^{GT}, N_t^{GT},\ D_t^{GT},\ AO_t^{GT}\} \in R^{6\ x\ 4H\ x\ 4W}$

Processing stage

- 1. upscaling & warping
- 2. flatten previous map to low resolution by apply space-to-depth transformation

$$S_s : \mathbb{R}^{6\times 4H\times 4W} \rightarrow \mathbb{R}^{4^2 6\times H\times W}.$$

- 3. Super-Resolution:
  - Network receive I_t^{LR} (5 channels) and the flattened, warped prediction from the previous frame O_f (16 * 6 channels), estimate the six-channel output O_t^{est}, (HR mask, normal, depth and AO maps)



- 4. Shading: generate a color image
  - screen-space Phong shading with AO

$$C_{rgb} = \mathrm{Phong}(c_a, c_d, c_s, c_m, N_t^{est}) * AO_t^{est},$$

  - Output mask M_t^{est}
    - Clamp first to [-1, +1], rescale to [0, 1], shows smooth fall-of values across edges

$$C_t^{est} = \mathrm{lerp}(c_{bg}, C_{rgb}, M_t'^{est}),$$

Loss functions

- 1. Spatial Loss

$$\mathcal{L}_{X,L_1} = ||X_t^{est} - X_t^{GT}||_1, \quad \mathcal{L}_{X,L_2} = ||X_t^{est} - X_t^{GT}||_2^2.$$

- 2. Perceptual loss
  - detailed outputs instead of smoothed mean values
  - two images are similar <- have similar activations in the latent space of a pre-trained network

$$\mathcal{L}_{X,P} = ||\phi(X_t^{est}) - \phi(X_t^{GT})||_2^2.$$

  - VGG-19 can used on shaded images
  - Differentiable Phong shading
    - Gradient can flow from loss function to update weight of neural networks
  - VGG on other entries, input transformed first
- 3. temporal loss
  - a temporal L2 loss
  - Penalize differences between current HR and previous warped one

$$\mathcal{L}_{X,\mathrm{temp}} = ||X_t^{est} - \tilde{X}_{t-1}^{est}||_2^2,$$

Loss function (cont'd)

- 4. Loss masking
  - Pixels where the mask is -1 set to the background color
  - Area are masked do not contribute to the final result
  - crucial step simplifies the network's task: network do not need to match masked areas

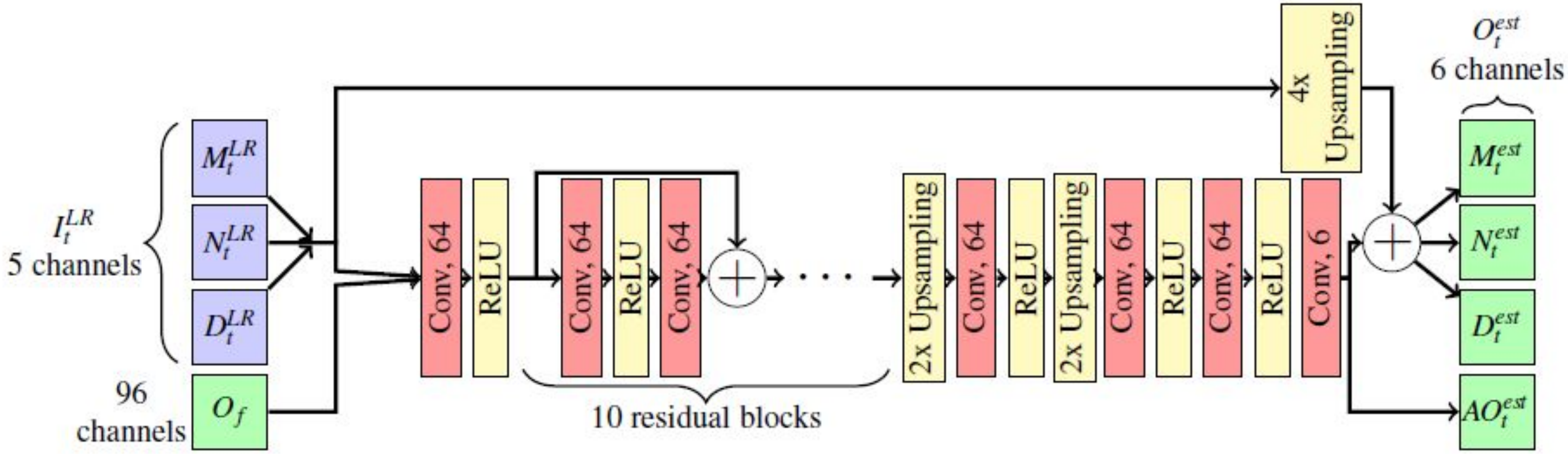
- 5. Adversarial Training
  - the discriminator is provided with:
    - the high resolution output O_t^{est}
    - The input I_t^{LR} as a conditional input
    - the previous frame I_{t-1}^{LR}, O_{t-1}^{est}, to learn to penalize for temporal coherence

$$\mathscr{L}_{\mathrm{GAN},D} = -\log(D(x)) - \log(1 - D(G(z))).$$

$$\mathscr{L}_{\mathrm{GAN},G} = -\log(D(G(z)))$$

Network architecture

- A fully convolutional frame-recurrent neural network (FRVSR-Net) consisting of a series of residual blocks



Training data

- images of isosurfaces from different timesteps and multi-resolution versions of the Ejecta dataset.
- render 500 sequences, each consisting of 10 frames
  - Each sequence, random select start and end viewpoints
  - Construct a smooth view path
  - Get resolution 128x128
- Crop sub-regions of 32x32
  - R: benefit batch processing

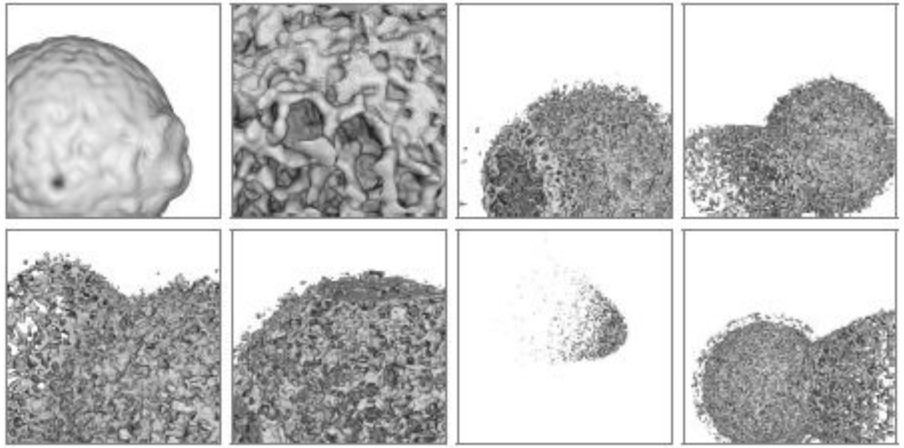
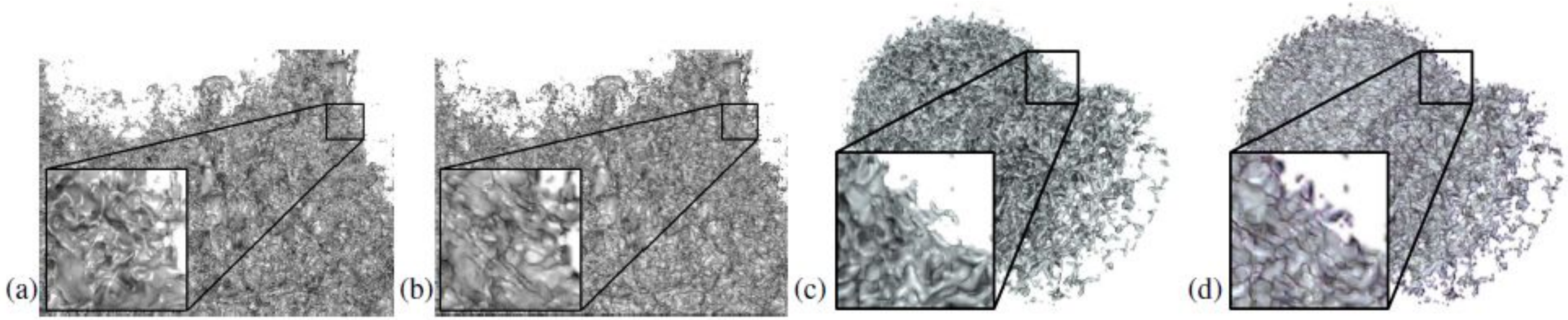


Fig. 5: Example images that are used to train our networks.

- Both LR&HR image directed generated by raycaster
- Initialize previous frame of 1^{st} frame
  - Zeroing all entries

Loss Function Characteristics

- geometric surface properties > color images



(a)(c) super-resolusion on depth and normal maps with screen-shading
(b)(d) super-resolusion on color images

Loss Function Characteristics

Network	Losses
Shaded	$\mathscr{L}_{\text{GAN},G} + 0.5\mathscr{L}_{C,P} + 50\mathscr{L}_{C,\text{temp}}$ network acts on shaded colors
L_1-color	$\mathscr{L}_{M,L_1} + \mathscr{L}_{AO,L_1} + 10\mathscr{L}_{C,L_1} + 0.1\mathscr{L}_{C,\text{temp}}$
L_1-geometry	$\mathscr{L}_{M,L_1} + \mathscr{L}_{AO,L_1} + 10\mathscr{L}_{N,L_1} + 100\mathscr{L}_{D,L_1} + 0.1\mathscr{L}_{C,\text{temp}}$
Perceptual	$\mathscr{L}_{M,L_1} + \mathscr{L}_{AO,L_1} + \mathscr{L}_{N,L_1} + \mathscr{L}_{D,L_1} + 0.1\mathscr{L}_{C,\text{temp}} + 5\mathscr{L}_{N,P} + \mathscr{L}_{AO,P}$
GAN	$\mathscr{L}_{M,L_1} + \mathscr{L}_{AO,L_1} + \mathscr{L}_{N,L_1} + \mathscr{L}_{D,L_1} + 0.1\mathscr{L}_{C,\text{temp}} + \mathscr{L}_{\text{GAN},G}$

TABLE 1: Networks and their specific loss function configurations.

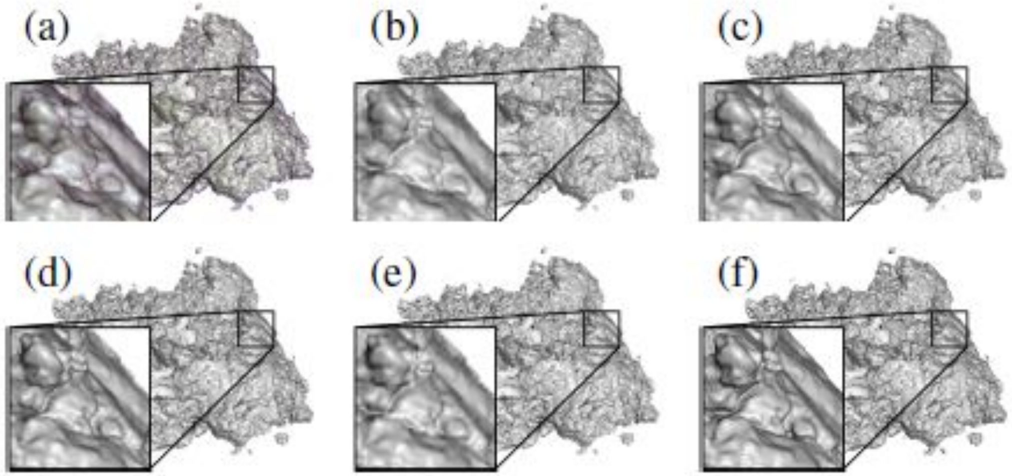


Fig. 6: Visual comparison of networks with different loss function configurations: (a) Shaded, (b) L_1-color, (c) L_1-geometry (our final model), (d) Perceptual, (e) GAN, (f) ground truth

Qualitative Evaluation

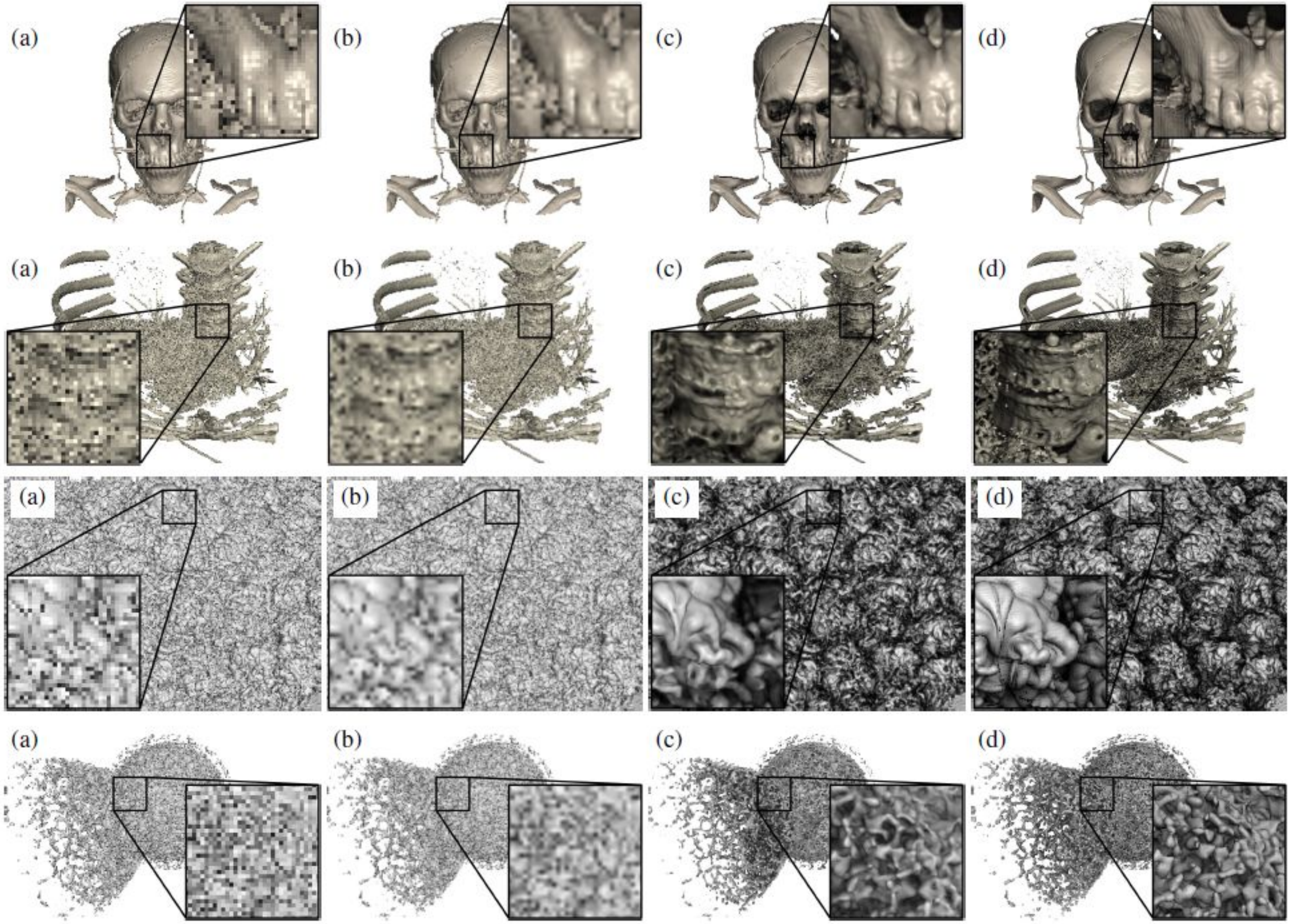


Fig. 7: Comparison of upscaling quality: (a) input, (b) bi-linear, (c) our network, (d) ground truth on the Skull, Thorax, Richtmyer-Meshkov and Ejecta dataset (top to bottom).

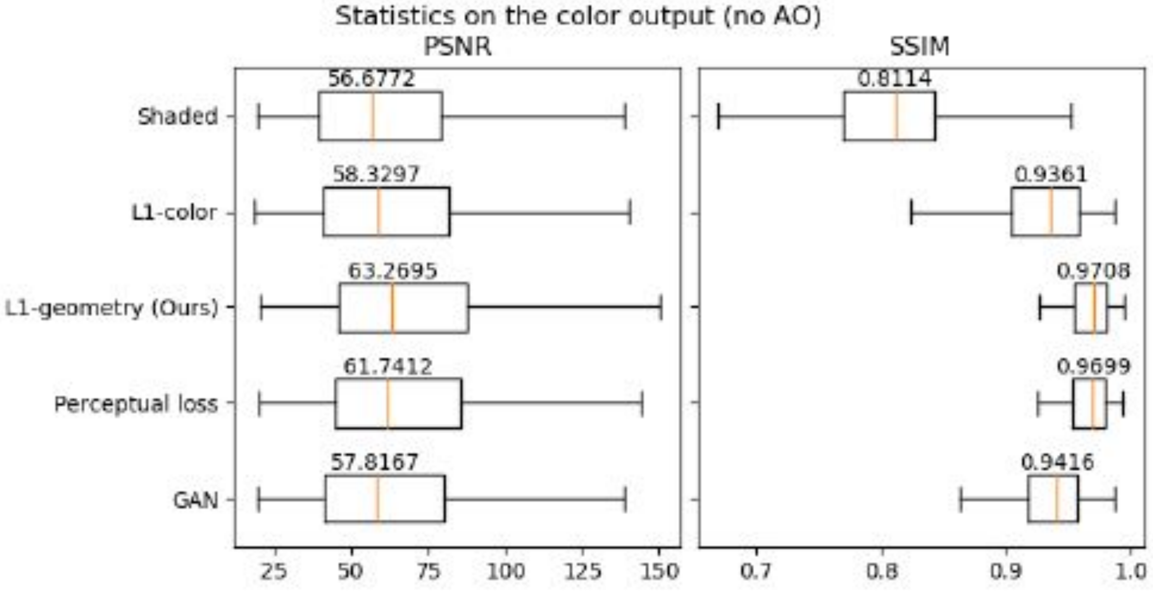
Quantitative Evaluatior

- peak signal-to-noise ratio (PSNR)

$$\mathrm{PSNR}(O_t^{est}, O_t^{GT}) = -10\log_{10}(||O_t^{est} - O_t^{GT}||_2^2),$$

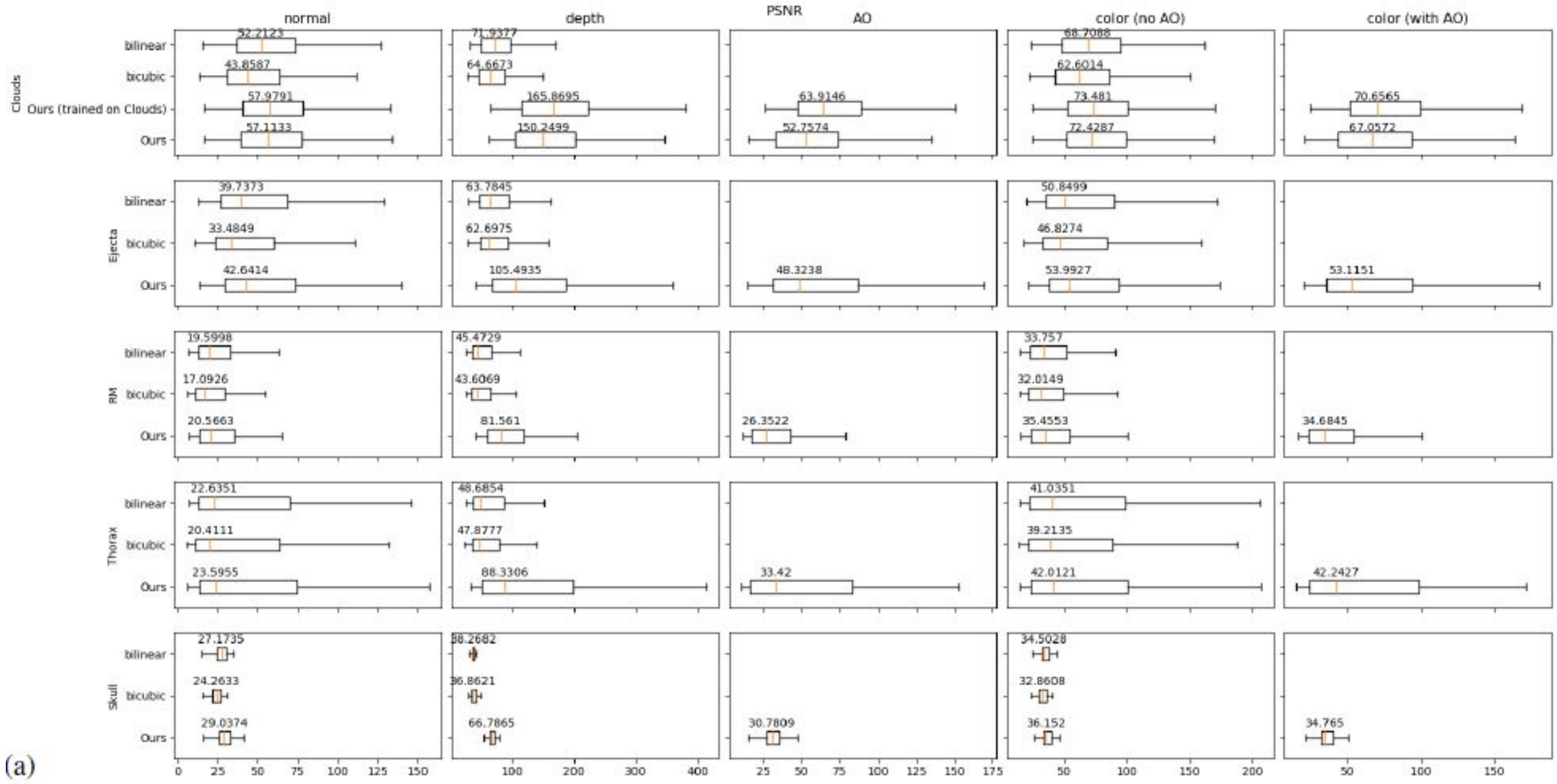
- Structural similarity index (SSIM)

$$\mathrm{SSIM}(O_t^{est}, O_t^{GT}) = \frac{(2\mu_{est}\mu_{GT} + c_1)(2\sigma_{est,GT} + c_2)}{(\mu_{est}^2 + \mu_{GT}^2 + c_1)(\sigma_{est}^2 + \sigma_{GT}^2 + c_2)},$$

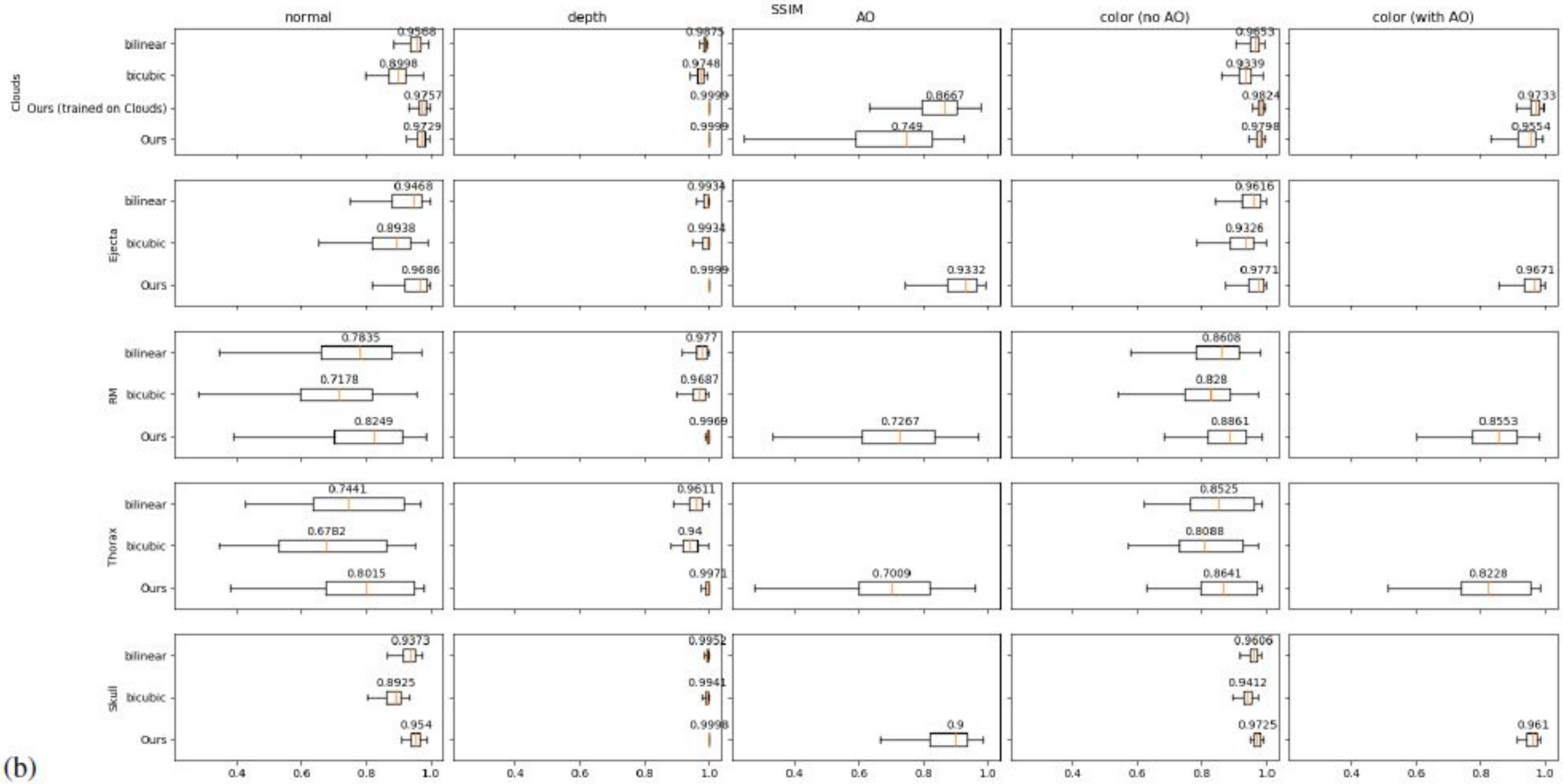


Different loss functions

Quantitative Evaluation (PSNR)

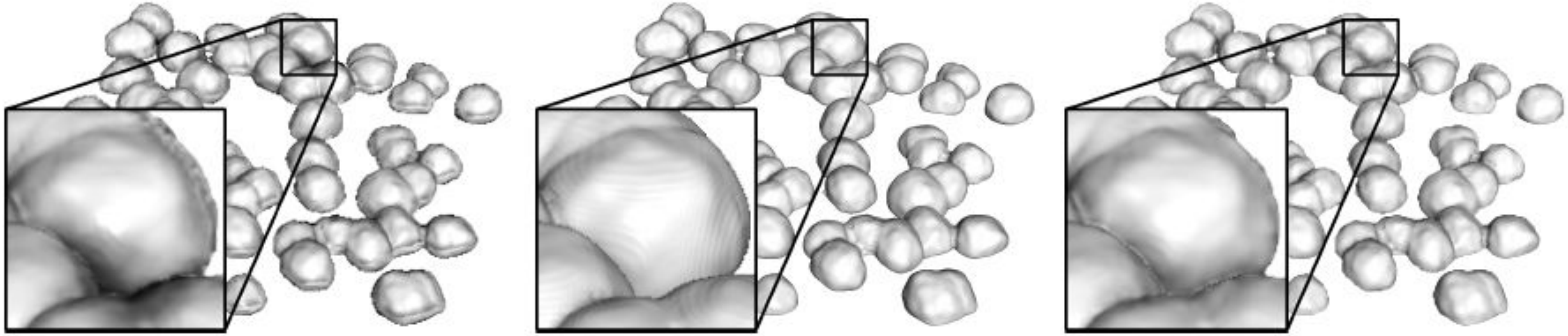


Quantitative Evaluation (SSIM)



(b)

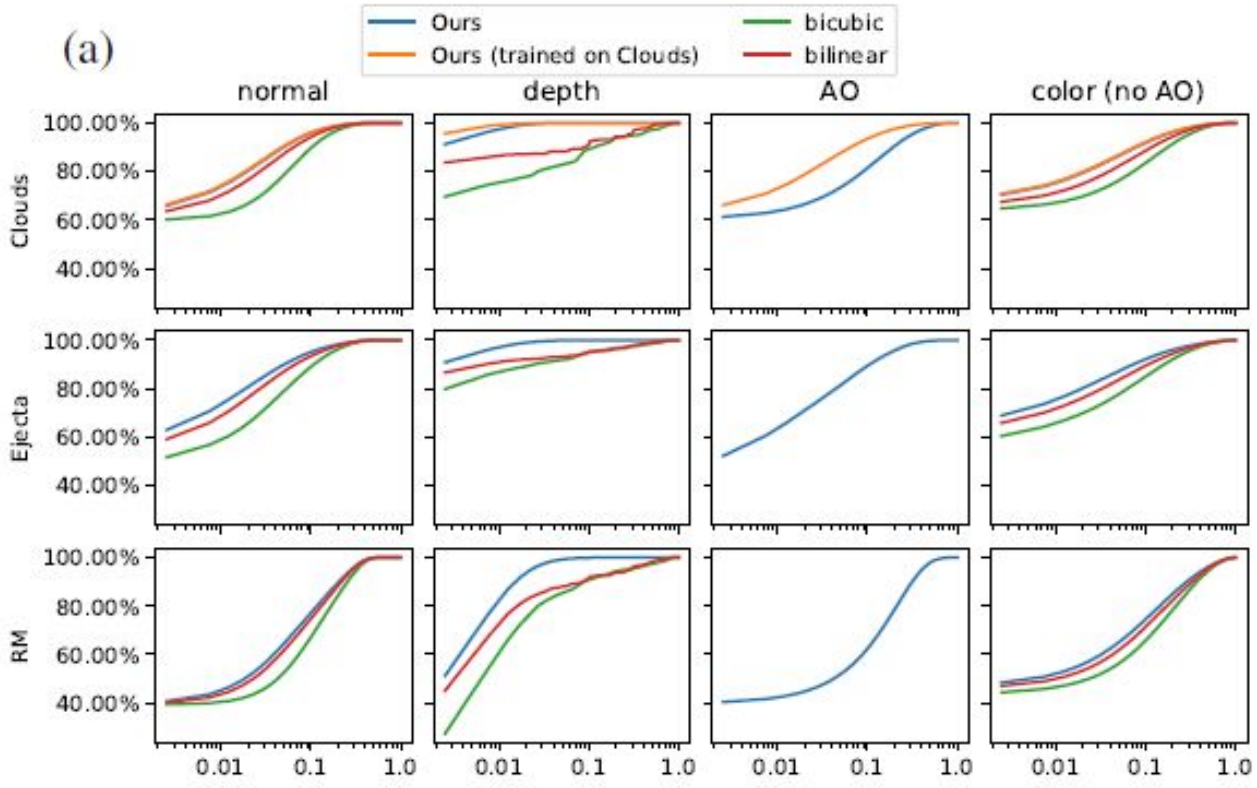
Evaluation



(a) Network trained on w/o Clouds. (b) Ground truth rendering of Cloud. (c) Network re-trained only on Clouds.

Evaluation

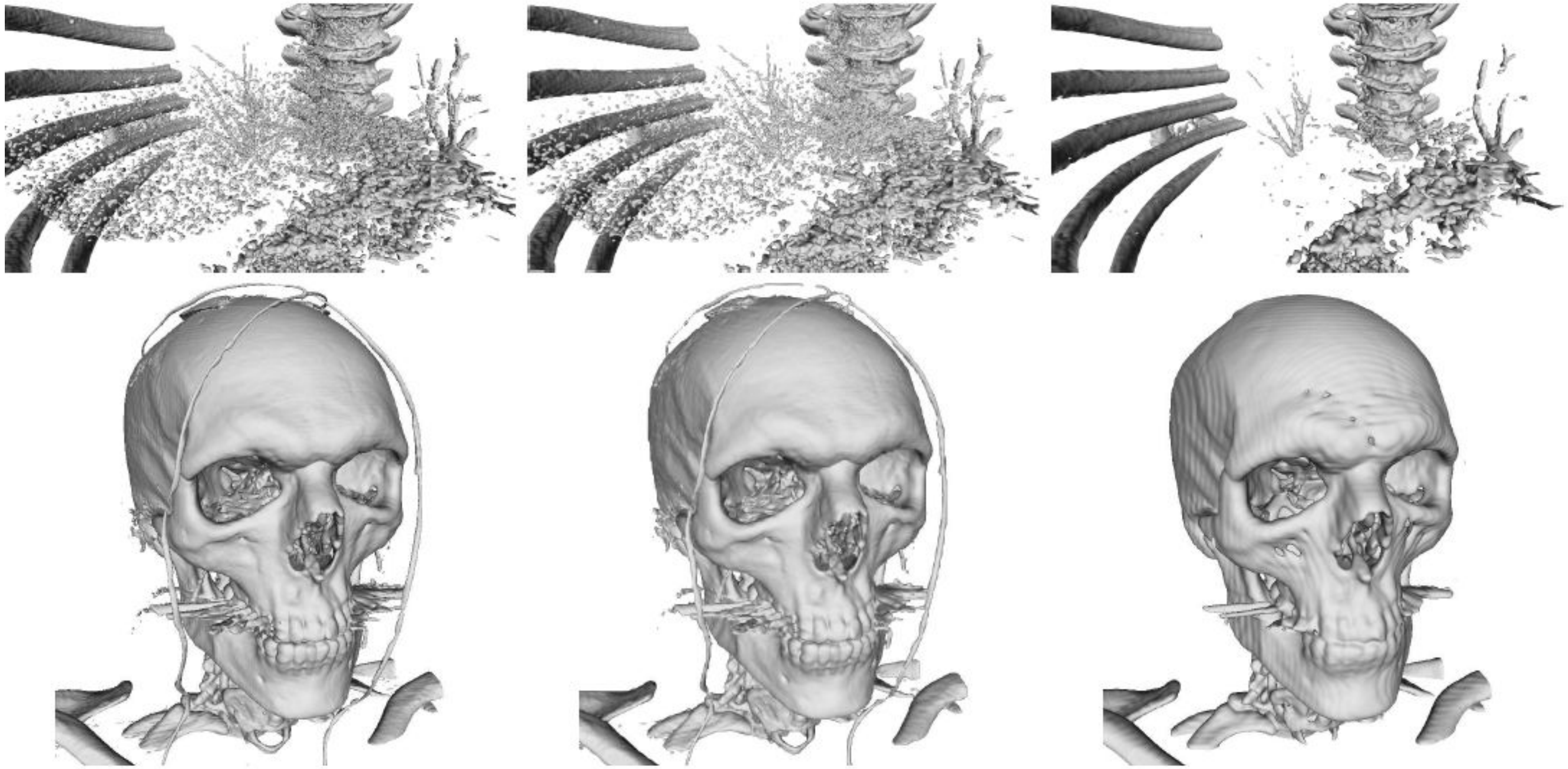
- Regression Error Characteristic (REC) curves
  - $REC(\tau) = P(|x^{est} - x^{GT}| \leq \tau)$



Times for rendering

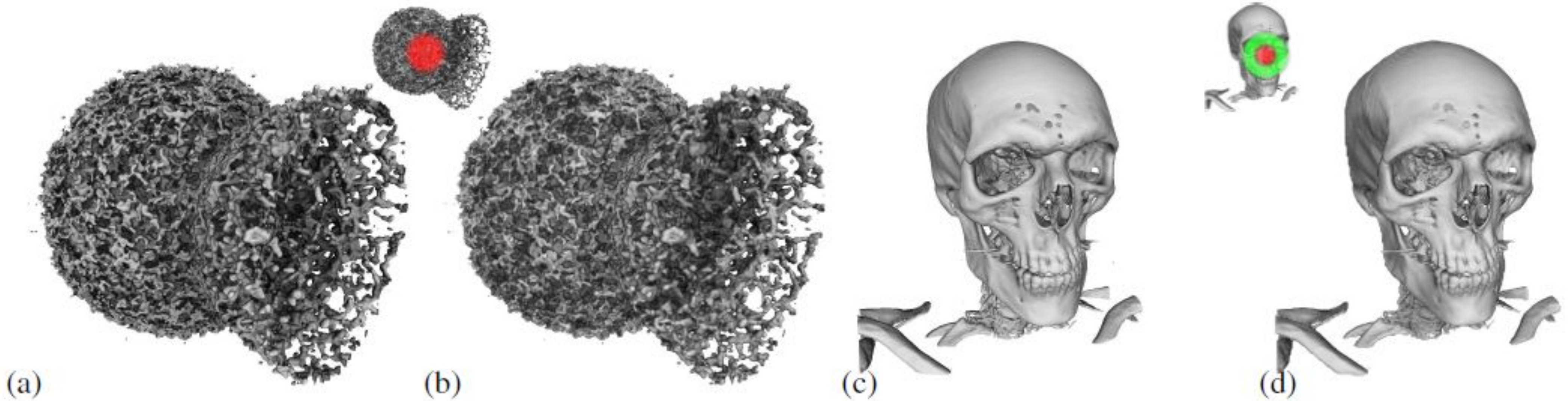
Dataset	High-res (no AO)	High-res (with AO)	Low-res	Super-res
Skull 256^3	0.057	4.2	0.0077	0.071
Thorax 256^3	0.069	9.1	0.010	0.071
R.-M. 1024^3	0.088	14.5	0.014	0.072
Ejecta 1024^3	0.163	18.6	0.031	0.072

Application (interactive visualization)



Left: full image resolution Middle: 1/4 resolution then upscale Right: half resolution

Application (Foveated rendering)



Conclusion

- deep learning technique for isosurface super-resolution with AO
- For me, learning with depth/normal maps is interesting, can be applied to other generative problems too