This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

Volumetric Isosurface Rendering with Deep Learning-Based Super-Resolution

Sebastian Weiss*, Mengyu Chu†, Nils Thuerey‡ and Rüdiger Westermann§
Technical University of Munich.
Email: *sebastian13.weiss@tum.de, †mengyu.chu@tum.de, ‡nils.thuerey@tum.de, §westermann@tum.de

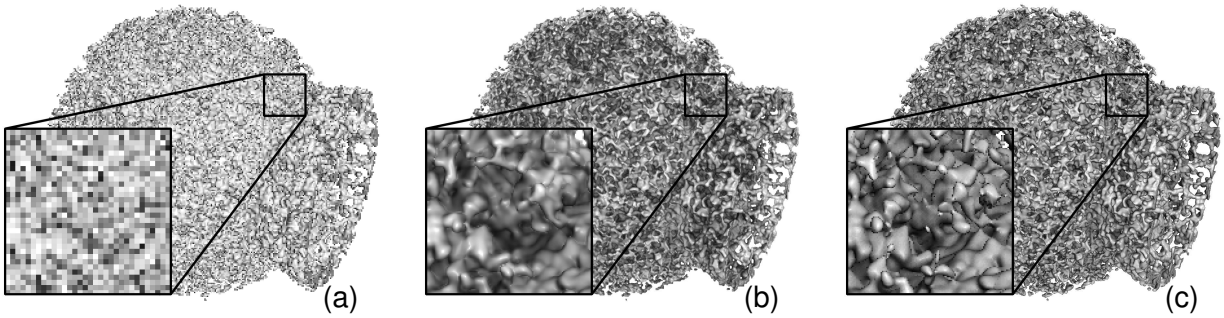


Fig. 1: Our super-resolution network can upscale (a) an input sampling of isosurface depths and normals at low resolution (i.e., 320x240), to (b) a high resolution depth and normal map (i.e., 1280x960) with ambient occlusion. For ease of interpretation, only the shaded output is shown. (c) The ground truth is rendered at 1280x960. Samples are from a 1024^3 grid, ground truth renders at 0.16 and 18.6 secs w/ and w/o ambient occlusion, super-resolution takes 0.07 sec.

Abstract—Rendering an accurate image of an isosurface in a volumetric field typically requires large numbers of data samples. Reducing this number lies at the core of research in volume rendering. With the advent of deep learning networks, a number of architectures have been proposed recently to infer missing samples in multi-dimensional fields, for applications such as image super-resolution. In this paper, we investigate the use of such architectures for learning the upscaling of a low resolution sampling of an isosurface to a higher resolution, with reconstruction of spatial detail and shading. We introduce a fully convolutional neural network, to learn a latent representation generating smooth, edge-aware depth and normal fields as well as ambient occlusions from a low resolution depth and normal field. By adding a frame-to-frame motion loss into the learning stage, upscaling can consider temporal variations and achieves improved frame-to-frame coherence. We assess the quality of inferred results and compare it to bi-linear and -cubic upscaling. We do this for isosurfaces which were never seen during training, and investigate the improvements when the network can train on the same or similar isosurfaces. We discuss remote visualization and foveated rendering as potential applications.

Index Terms—Machine Learning ; Extraction of Surfaces (Isosurfaces, Material Boundaries) ; Volume Rendering.

1 Introduction

Much of the research in isosurface volume ray-casting has been devoted to the development of efficient search structures, i.e., data structures that can reduce the number of data samples required to determine where a view ray intersects the surface. Despite their high degree of sophistication, for large volumes with heterogeneous composition the traversal of these data structures becomes increasingly costly. Since the workload of ray-casting is linear in the number of pixels, frame rates can drop significantly when isosurfaces in large volumes are rendered on high resolution display systems.

This effect is intensified if global illumination effects are considered. An important global illumination effects for isosurfaces is ambient occlusion (AO). AO estimates for every surface point the attenuation of ambient light from the surrounding, and uses this information to enhance cavities and locations closely surrounded by other surface parts. AO is simulated by testing along many secondary rays per surface point whether the isosurface is hit, requiring so many data samples, in general, that interactive frame rates cannot be maintained.

In this work, we investigate the potential of convolutional neural networks to further reduce the number of samples in isosurface ray-casting, for both the reconstruction of the surfaces' geometry and ambient occlusions on it. This strategy works in tandem with an acceleration structure to even more aggressively reduce the number of samples. First, we shed light on the question whether an accurate high resolution image of the surface—a super-resolution image—can be inferred from only the surface points at a far lower image resolution. Second, we aim at investigating whether ambient occlusions can be inferred from the surface geometry without the need to explicitly compute occlusions on that geometry.

From a signal theoretical point of view, it can be argued that new structures—beyond what can be predicted from multiple frames of low resolution inputs by classical up-scaling filters like bi-linear or -cubic interpolation—cannot be inferred without any further assumptions about their occurrence. Recent works in deep learning have demonstrated that such assumptions can be learned by an artificial neural network. Learning-based image and video super-resolution have achieved remarkable results, by training networks

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

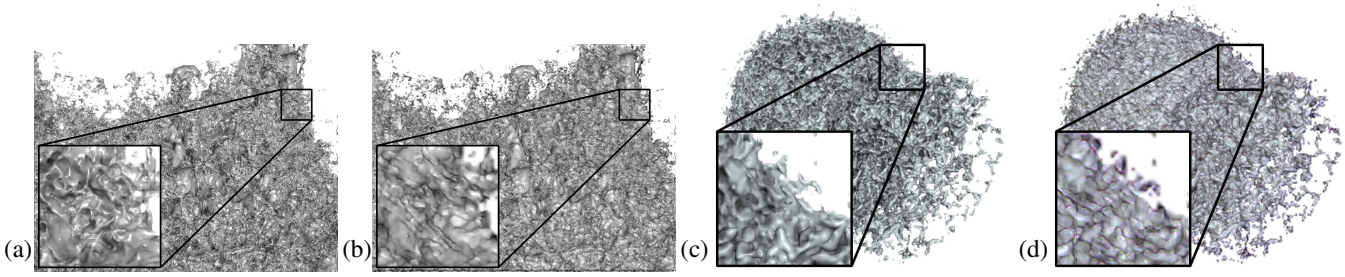


Fig. 2: Super-resolution on depth and normal maps with screen-space shading (a,c) leads to superior reconstruction quality compared to super-resolution on color images (b,d). These images exhibit color bleeding and shifts, while our screen-space shading approach successfully prevents these artifacts. (b) was converted to gray-scale to emphasize geometric differences.

using corresponding pairs of low- and high resolution color images [1], [2]. Learned assumptions can then be transferred to a new low resolution input, to generate a high resolution variant that adheres to the structures seen at training time.

Similar observations have been reported in current works in visualization, which demonstrate the use of neural-network-based inference of data samples for volume upscaling and in-situ visualization, as well as parameter-space exploration. Networks are used to infer high resolution scalar fields and missing time-steps between 3D simulation results [3], [4], and they learn the dependencies between simulation results and the simulation parameters to infer the results for new parameter settings [5]. These works demonstrate that a network can learn to infer new data from given samples, by learning either some type of interpolation, or the local effects of parameter value modifications constraint by the global data distribution.

1.1 Contribution

We present an artificial neural network that learns to upscale a sampled representation of geometric properties of an isosurface at low resolution to a higher resolution. We introduce a fully convolutional neural network, using the FRVSR-Net [6] as a basis, to learn a latent representation that generates a smooth, edge-aware depth and normal field, as well as ambient occlusions, from a low resolution depth and normal field. To support user navigation, we integrate a loss function into the training pass that penalizes frame-to-frame inconsistencies and achieves improved temporal coherence in the reconstruction step.

Even though similar in spirit to classical super-resolution techniques, we strive for a conceptually different approach in this work: Instead of using color images and down-scaled ground truth images for training, we aim at incorporating 3D scene information in the form of per-frame depth and normal fields into the training and reconstruction process. It is our goal to let the network learn the relations between the isosurface geometry sampled at a low and a high resolution, and to infer on the relations between the geometry and shading.

We use the neural network to infer images of isosurfaces with four times the resolution of the input images. Figure 1 demonstrates the result of the upscaling process. Since the network is designed to learn high resolution ambient occlusions from low resolution depth and normal images, computations of ambient occlusions at runtime are entirely avoided. Thus, compared to volumetric ray-casting at full resolution, the number of samples from the volumetric field can be reduced drastically.

To analyse the pixel-wise reconstruction error of the network, we compare reconstructed images to ground truth renderings and reconstructions using bi-linear and bi-cubic upscaling. These comparison are performed using the peak signal-to-noise ratio (PSNR) between ground truth and reconstructed results, as well as the structure-similarity metric (SSIM) [7] that gives more weight to the perceived quality of the results. We demonstrate very good reconstruction quality even for isosurfaces that were never shown to the network during training, and that the network's accuracy can be even improved by retraining on isosurfaces of shapes similar to the ones used in the inference step.

Our specific contributions are:

- We show that it is beneficial to train the network based on depth and normal images instead of color. Our results indicate that this training process results in an improved learning of geometric surface properties, as illustrated in Figure 2.
- Instead of letting the network learn to infer AO in the high resolution output from AO in the low resolution inputs, our networks only receive low resolution depth and normal maps as input. Thus, AO does not need to be simulated at the samples of the low resolution input, which would significantly increase the rendering time.
- To let the network learn to maintain frame-to-frame coherence, we additionally add a motion loss for the generated image content. In this way, the network achieves improved reconstruction quality and becomes well suited for interactive exploration tasks.
- We perform a quality evaluation to shed light on the reconstruction accuracy of learning-based isosurface reconstruction. This evaluation shows the strengths and weaknesses of this type of reconstruction, and helps to better understand its specific properties and possible application scenarios.

For a number of isosurfaces with vastly different geometric properties, we demonstrate the potential of learning-based upscaling. Furthermore, we discuss several use cases where isosurface inference can significantly improve and accelerate existing approaches, e.g., during interactive navigation in remote visualization environments, in focus+context visualization, and in foveated rendering.

2 Related Work

Our approach works in combination with established acceleration techniques for volumetric ray-casting of isosurfaces, and builds upon recent developments in image and video super-resolution

via artificial neural networks to further reduce the number of data access operations.

Volumetric Ray-Casting of Isosurfaces: Over the last decades, considerable effort has been put into the development of acceleration techniques for isosurface ray-casting in 3D scalar fields. Direct volume ray-casting of isosurfaces was proposed by Levoy [8]. Classical fixed-step ray-casting traverses the volume along a ray using equidistant steps in the order of the voxel size. Acceleration structures for isosurface ray-casting encode larger areas were the surface cannot occur, and ray-casting uses this information to skip these areas with few steps. One of the most often used acceleration structure is the min-max pyramid [9], a tree data structure that stores at every interior node the interval of data values in the corresponding part of the volume. Pyramidal data structures are at the core of most volumetric ray-casting techniques to effectively reduce the number of data samples that need to be accessed during ray traversal.

For selected isosurfaces, bounding cells or simple geometries were introduced to restrict ray-traversal to a surface's interior [10], [11]. Adaptive step-size control according to pre-computed distance information aimed at accelerating first-hit determination [12]. Recently, SparseLeap [13] introduced pyramidal occupancy histograms to generate geometric structures representing non-empty regions. They are then rasterized into per-pixel fragment lists to obtain those segments that need to be traversed.

Significant performance improvements have been achieved by approaches which exploit high memory bandwidth and texture mapping hardware on GPUs for sampling and interpolation in 3D scalar fields [14], [15]. For isosurface ray-casting, frame to frame depth buffer coherence on the GPU was employed to speed up first-hit determination [16], [17]. A number of approaches have shown the efficiency of GPU volume ray-casting when paired with compact isosurface representations, brick-based or octree subdivision, and out-of-core strategies for handling data sets too large to be stored on the GPU [18], [19], [20], [21]. For a thorough overview of GPU approaches for large-scale volume rendering, let us refer to the report by Beyer *et al.* [22].

Related to isosurface rendering is the simulation of realistic surface shading effects. AO estimates for every surface point the integral of the visibility function over the hemisphere [23]. AO can greatly improve isosurface visualization, by enhancing the perception of small surface details. A number of approximations for AO simulation in volumetric data sets have been proposed, for instance, local and moment-based approximations of occluding voxels [24], [25] or pre-computed visibility information [26]. The survey by Ropinski *et al.* [27] provides a thorough overview of the use of global illumination in volume visualization. Even though very efficient screen-space approximations of AO exist [28], [29], we decided to consider ray-traced AO in object-space to achieve high quality.

Deep Learning of Super-Resolution and Shading: For super-resolution of natural images, deep learning based methods have progressed rapidly since the very first method [1] surpassed traditional techniques in terms of peak signal-to-noise ratio (PSNR). Regarding network architectures, Kim et al. introduced a very deep network [30], Lai et al. designed the Laplacian pyramid network [31], and advanced network structures have been applied, such as the ResNet [32], [33] and DenseNet [34], [35] architectures. Regarding loss formulations, realistic high-frequency detail is significantly improved by using adversarial and perceptual losses based on pretrained networks [33], [36]. Compared to single-image methods, video super-resolution tasks introduce the time dimensions, and as such require temporal coherence and consistent image content across multiple frames. While many methods use multiple low resolution frames [37], [38], [39], the FRVSR-Net [6] reuses the previously generated high resolution image to achieve better temporal coherence. By using a spatio-temporal discriminator, the TecoGAN [40] network produced results with spatial detail without sacrificing temporal coherence. Overall, motion compensation represents a critical component when taking into account multiple input frames. Methods either use explicit motion estimation and rely on its accuracy [6], [40], [41], [42], or spend extra efforts implicitly such as detail fusion [37] and dynamic upsampling [39]. In our setting, we can instead leverage the computation of reliable screen-space motions via raytracing.

In a different scenario, neural networks were trained to infer images from a noisy input generated via path-tracing with low number of paths, of the same resolution as the target, but with significantly reduced variance in the color samples [43], [44]. Deep shading [45] utilized a neural network to infer shading from rendered images, targeting attributes like position, normals, and reflections for color images of the same resolution. None of these techniques used neural networks for upscaling as we do, yet they are related in that they use additional parameter buffers to improve reconstruction quality of global illumination.

Deep Learning of Volumetric Fields: For volume visualization, Zhou et al. [3] presented a learning-based approach for volume upscaling which better preserves structural details and volume quality than linear upscaling. Berger et al. [46] proposed a deep image synthesis approach to assist transfer function design using generative adversarial networks (GANs). Recently, the use of convolutional neural networks for temporal upscaling has been introduced [4]. The authors demonstrate the application of learning-based reconstruction for in-situ visualization, by letting a network learn to infer the time evolution of a physical field in-between a pair of simulated time steps. In another work it has been demonstrated that a neural network can learn the relationships between simulation parameters and the simulation results [5]. By using training samples consisting of parameter sets and corresponding results, the network can infer the parameter dependencies and use the build latent representation to generate results for new input values.

3 Isosurface Learning

Our method consists of a pre-process in which an artifical neural network is trained, and the upscaling process which receives a new low resolution isosurface image and uses the trained network to perform the upscaling of this image. Our network is designed to perform $4\times$ upscaling, i.e. from input images of size $H \times W$ to output images of size $4H \times 4W$. Note, however, that other upscaling factors can be realized by straight forward adaptations and network re-training.

The network is trained on unshaded surface points. It receives the low resolution input image in the form of a depth and normal map for a selected view, as well as corresponding high resolution maps with an additional AO map that is generated for that view. A low- and high resolution binary mask indicate those pixels where the surface is hit. Once a new low resolution input image is upscaled, i.e., high resolution depth, normal and AO maps are reconstructed, screen-space shading is computed and added to AO in a post-process to generate the final color.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

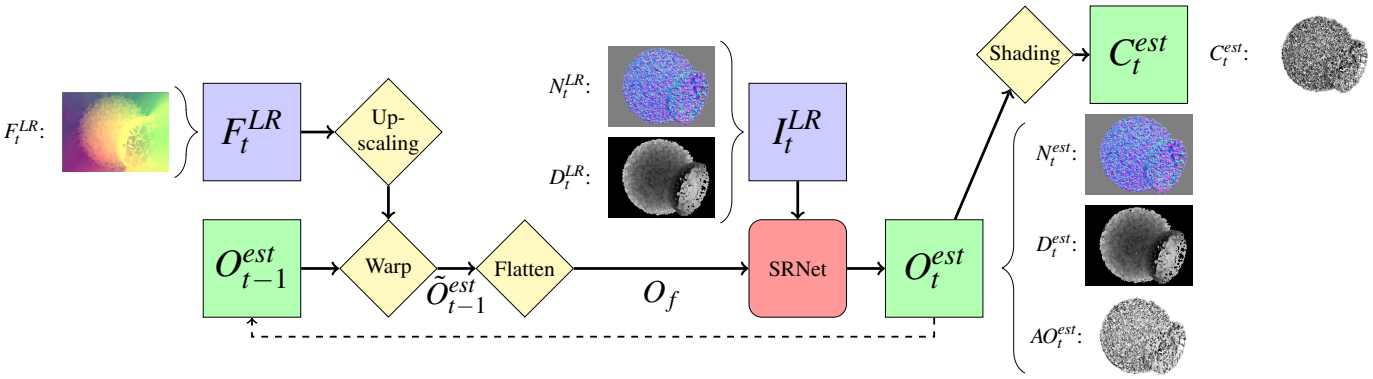


Fig. 3: Overview of network-based learning of isosurface image upscaling. Blue: low resolution inputs, green: high resolution outputs, yellow: fixed processing steps, red: trained network. From left to right: The current optical flow F_t^{LR} is used to warp the output of the previous time step O_{t-1}^{est}. The current low resolution input I_t^{LR} and the warped previous output is given as input to the network. The network produces the output O_t^{est} including estimated high resolution mask (M_t^{est}, not shown in the figure), normal (N_t^{est}), depth (D_t^{est}), and AO (AO_t^{est}) maps . Deferred shading is used to generate the final color (C_t^{est}).

The network, given many low- and high resolution pairs of input maps for different isosurfaces and views, internally builds a so-called latent representation that aims at mapping the low resolution inputs to their high resolution counterparts. A loss function is used to penalize differences between the high resolution learned and ground-truth variants. We investigate different networks, trained with collections of randomly sampled views from a small number of exemplary datasets. Images of isosurfaces at full resolution are used as ground-truth training data. We analyze the upscaling quality of these networks on new views for the training data, as well as views of isosurfaces in datasets the networks have never seen during training.

3.1 Input Data

Both the low- and high resolution input maps are generated via volumetric ray-casting. AO in the high resolution image is simulated by spawning 512 additional secondary rays per surface point, and testing each of them for an intersection with the surface. Since we aim at supporting temporally coherent super-resolution, all images have a time subscript t, starting with $t = 1$ at the first frame.

The following low resolution input maps of size $H \times W$ are used in the training step:

- $M_t^{LR} \in [-1,+1]^{H\times W}$: The binary input mask that specifies for every pixel whether the isosurface is hit (mask=1) or not (mask=-1). Internally, the network learns continuous values, and uses these values to smoothly blend the final color over the background.
- $N_t^{LR} \in [-1,+1]^{3\times H\times W}$: The normal map with the normal vectors in screen-space.
- $D_t^{LR} \in [0,1]^{H\times W}$: The depth map, in which 0 indicates that no hit was determined.

The low resolution input to the network is then given by $I_t^{LR} := \{M_t^{LR}, N_t^{LR}, D_t^{LR}\} \in \mathbb{R}^{5\times H\times W}$. We subsequently call this the low resolution input image.

In addition, the following map is generated during ray-casting:

- $F_t^{LR} \in [-1,+1]^{2\times H\times W}$: A map of 2D displacement vectors, indicating the screen-space flow from the previous view to the current view.

The screen-space flow is used to align the previous high resolution maps with the current low resolution input maps. Under the assumption of temporal coherence, the network can then minimize for the deviation of the currently inferred high resolution map from the temporally extrapolated previous one in the training process. To compute the screen-space flow, assume that in the low resolution view the current ray hits the isosurface at world position x_t. Since during rendering the current and previous model-view-projection matrices are known, the current and previous screen-space coordinates of the point x_t, i.e., x_t' and x_{t-1}', can be computed. The flow is then computed as $f_t := x_t' - x_{t-1}'$, indicating how to displace the previous mask, depth and normal maps at time $t-1$ to align them with the frame at time t. Since the described method provides the displacement vectors only at locations in the low resolution input image where the isosurface is hit in the current frame, we use a Navier-Stokes-based image inpainting [47] via OpenCV [48] to obtain a dense displacement field F_t^{LR}. The inpainting algorithm fills the empty regions in such a way that the resulting flow is as incompressible as possible.

For aligning the previous maps, the current flow field is first upscaled via bi-linear interpolation to the high resolution. In a semi-Lagrangian fashion, we generate new high resolution maps where every pixel in the upscaled maps retrieves the value in the corresponding high resolution map from the previous frame, by using the inverse flow vector to determine the target location.

The high resolution input data, which is used as ground truth in the training process, is comprised of the same maps as the low resolution input, plus an AO map $AO_t^{GT} \in [0,1]^{4H\times 4W}$. Here, values of one or zero indicate no or full occlusion, respectively. Thus, the ground truth image can be written as $O_t^{GT} := \{M_t^{GT}, N_t^{GT}, D_t^{GT}, AO_t^{GT}\} \in \mathbb{R}^{6\times 4H\times 4W}$. Once the network is trained with I_t^{LR} and O_t^{GT}, it can infer a new high resolution output image $O_t^{est} := \{M_t^{est}, N_t^{est}, D_t^{est}, AO_t^{est}\} \in \mathbb{R}^{6\times 4H\times 4W}$ from a given low resolution image and the high resolution output of the previous frame.

3.2 Super-Resolution Surface Inference

Once the network has been trained, new low resolution input images are given to the network to infer the corresponding high resolution output images. For the inference step, we build upon the frame-recurrent neural network architecture proposed by Sajjadi *et al.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

[6]. At the current timestep t, the network is given the input I_t^{LR} and the previous high resolution prediction O_{t-1}^{est}, warped using the image-space flow, for temporal coherence. It produces the current prediction O_t^{est}, and after a post-processing step also the final color $C_t^{est} \in [0,1]^{3\times 4H\times 4W}$.

Figure 3 shows all data that are used and inferred by the network, together with the different processing stages an inference step is comprised of. These processing stages are:

1. Upscaling and Warping: After upscaling the screen-space flow F_t^{LR}, it is used as described to warp all previous estimated maps O_{t-1}^{est}, leading to $\tilde{O}_{t-1}^{est} \in \mathbb{R}^{6\times 4H\times 4H}$.

2. Flattening: Next, the warped previous maps $\tilde{O}_{t-1}^{est}$ are flattened into the low resolution by applying a space-to-depth transformation [6]

$$S_s : \mathbb{R}^{6\times 4H\times 4W} \rightarrow \mathbb{R}^{4^2 6\times H\times W}. \tag{1}$$

I.e., every 4×4 block of the high resolution image is mapped to a single pixel in the low resolution image. The channels of these $4\times 4 = 16$ pixels are concatenated, resulting in a new low resolution image O_f with 16-times the number of channels.

3. Super-Resolution: The super-resolution network then receives the current low resolution input I_t^{LR} (5 channels) and the flattened, warped prediction from the previous frame O_f (i.e., $16\cdot 6$ channels). The network then estimates the six channels of the output O_t^{est}, the high resolution mask, normal, depth, and AO maps.

4. Shading: To generate a color image, screen-space Phong shading with AO is applied as a post-processing step, i.e.,

$$C_{rgb} = \text{Phong}(c_a, c_d, c_s, c_m, N_t^{est}) * AO_t^{est}, \tag{2}$$

with the ambient color c_a, diffuse color c_d, specular color c_s and material color c_m as parameters.

The network also produces a high resolution mask M_t^{est} as output. While the input mask M_t^{LR} is comprised only of values -1 (outside) and +1 (inside), M_t^{est} can take on any value. Hence, M_t^{est} is clamped first to $[-1,+1]$ and then rescaled to $[0,1]$, leading to $M_t'^{est}$. This map shows a smooth fall-off of values across edges and and allows the network to smooth out edges via

$$C_t^{est} = \text{lerp}(c_{\text{bg}}, C_{rgb}, M_t'^{est}), \tag{3}$$

with c_{bg} being the background color.

3.3 Loss Functions

In the following, we describe the loss functions we have used during training to calculate the model error in the optimization process. Via the loss functions, the importance of certain features—and thus the fidelity by which they can be inferred—can be controlled. The single loss functions we use are common in artificial neural networks, yet in our case they are applied separately to different channels of the inferred and ground truth images. The total loss function used for training the network is a weighted sum of the loss functions below. In section 5, we analyze the effects of different loss functions on the reconstruction quality.

1. Spatial loss: As a baseline, we employ losses with regular vector norms, i.e. L_1 or L_2, on the different outputs of the network. Let X be either the mask M, the normal map N, the depth map D, the AO map AO or the shaded output C. Then the L_1 and L_2 losses are given by:

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

2. Perceptual loss: Perceptual losses, as proposed by Gatys *et al.* [49], Dosovitskiy and Brox [50], and Johnson *et al.* [51] have been widely adopted to guide learning tasks towards detailed outputs instead of smoothed mean values. The idea is that two images are similar if they have similar activations in the latent space of a pre-trained network. Let ϕ be the function that extracts the layer activations when feeding the image into the feature network. Then the distance is computed by

$$\mathscr{L}_{X,P} = ||\phi(X_t^{est}) - \phi(X_t^{GT})||_2^2. \tag{4}$$

As feature network ϕ, the pretrained VGG-19 network [52] is used. We used all convolution layers in all spatial dimensions as features, with weights scaled so that each layer has the same average activation when evaluated over all input images.

Since the VGG network was trained to recognize objects in color images in the space $[0,1]^3$, the shaded output C can be directly used. This perceptual loss on the color space can be backpropagated to the network outputs, i.e. normals and ambient occlusions, with the help of the differentiable Phong shading. This shading is part of the loss function, and is implemented such that gradients can flow from the loss evaluation into the weight update of the neural network during training. Hence, with our architecture the network receives a gradient so that it can learn how the output, e.g., the generated normals, should be modified such that the shaded color matches the look of the target image. When applying the perceptual loss on other entries, the input has to be transformed first. The normal map is rescaled from scale $[-1,+1]^3$ to $[0,1]^3$, and depth and masking maps are converted to grayscale RGB images. We did not use additional texture or style loss terms [36], [49], since these introduce artificial details and roughness in the image which is not desired in smooth isosurface renderings.

3. Temporal loss: All previous loss functions worked only on the current image. To strengthen the temporal coherence and reduce flickering, we employ a temporal L_2 loss [53]. We penalize differences between the current high resolution image O_t^{est} and the previous, warped high resolution image $\tilde{O}_{t-1}^{est}$ with

$$\mathscr{L}_{X,\text{temp}} = ||X_t^{est} - \tilde{X}_{t-1}^{est}||_2^2, \tag{5}$$

where X can be M, N, D, AO or C.

In the literature, more sophisticated approaches to improve the temporal coherence are available, e.g. using temporal discriminators [40]. These architectures give impressive result, but are quite hard to train. We found that already with the proposed simple temporal loss, good temporal coherence can be achieved in our current application. We refer the readers to the accompanying video for a sequence of a reconstruction over time.

4. Loss masking: During screen-space shading in the post-process (see section 3.2), the output color is modulated with the mask indicating hits with surface points. Pixels where the mask is -1 are set to the background color. Hence, the normal and AO values produced by the network in these areas are irrelevant for the final color.

To reflect this in the loss function, loss terms that do not act on the mask (i.e. normals, ambient occlusions, colors) are itself modulated with the mask so that areas that are masked out don't contribute to the loss. We found this to be a crucial step that simplifies the network's task: In empty regions, the ground truth images are filled with default values in the non-mask channels, while with loss masking, the network does not have to match these values.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

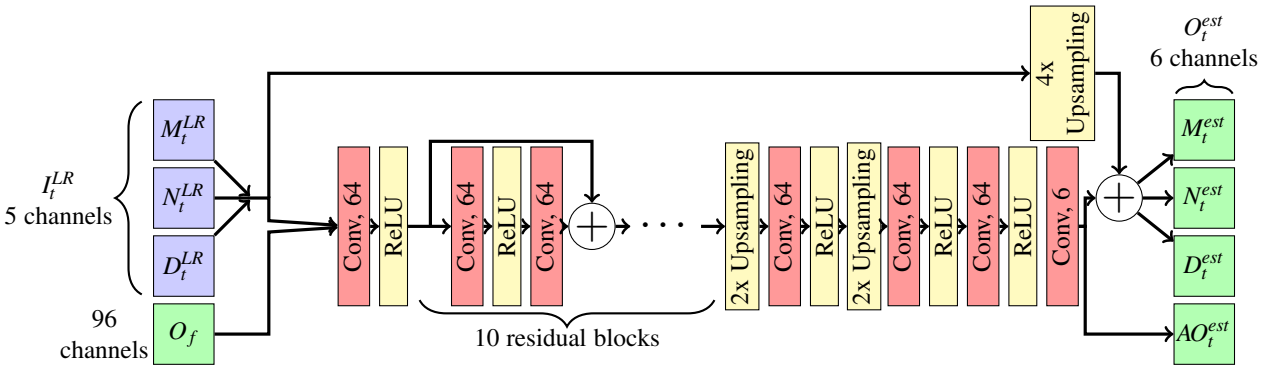


Fig. 4: Network architecture for the SRNet. Within the network, $\oplus$ indicates component-wise addition of the residual. All convolutions use 3x3 kernels with stride 1. Bilinear interpolation was used for the upsampling layers.

5. Adversarial Training: Lastly, we also employed an adversarial loss as inspired by Chu *et al.* [40]. In adversarial training, a discriminator network is trained parallel to the super-resolution network generator. The discriminator receives ground truth images and predicted images, and is trained to classify whether the input was ground truth or not. This discriminator is then used in the loss function of the generator network (see, e.g., Goodfellow *et al.* [54] for further details).

In our scenario, for evaluating the predicted images the discriminator is provided with

- the high resolution output O_t^{est}, and optionally the color C_t^{est},
- the input image I_t^{LR} as a conditional input to learn and penalize the mismatching between input and output,
- the previous frames $I_{t-1}^{LR}, O_{t-1}^{est}$, and optionally C_{t-1}^{est} to learn to penalize for temporal coherence.

To evaluate the discriminator score of the ground truth images, the predicted images O^{est} are replaced by O^{GT}.

As a loss function of the discriminator, we use the binary cross entropy loss. More concretely, let z be the input over all timesteps and $G(z)$ the generated results, i.e. the application of the super-resolution prediction on all timesteps. Let $D(x)$ be the discriminator that takes the high resolution outputs as input and produces a single scalar score. Then the discriminator is trained to distinguish fake from real structures by minimizing

$$\mathscr{L}_{\text{GAN},D} = -\log(D(x)) - \log(1 - D(G(z))). \tag{6}$$

The generator is trained to minimize

$$\mathscr{L}_{\text{GAN},G} = -\log(D(G(z))) \tag{7}$$

4 Learning Methodology

In this chapter, we provide a detailed description of the used network architecture, as well as the training and inference steps. We also shed light on the dependency of the reconstruction quality on the used loss functions.

4.1 Network Architecture

The network architecture is a fully convolutional frame-recurrent neural network (FRVSR-Net) consisting of a series of residual blocks [6]. An illustration of the network's building blocks and its topology is given in Figure 4. The modifications we have performed are with respect to the number of input and output channels, the other parts of the network are kept unchanged. The generator network starts with one convolutional layer to reduce the 101 input channels (5 from I_t^{LR}, $6*4^2$ from O_f) into 64 channels. Next, 10 residual blocks are used, each of which contains 2 convolutional layers. These are followed by two upscaling blocks ($2\times$ bilinear upscaling, a convolution and a ReLU) arriving at a $4\times$ resolution, still with 64 channels. In a final step, two convolutions process these channels to reduce the latent feature space to the desired 6 output channels. All layers use 3x3 kernels with stride 1.

The network is a residual network, i.e. it learns changes to the input. As shown in previous work [32], this improves the network's capability to generalize to new data, as it can focus on generating the residual content. Hence, the 5 channels of the input are bi-linearly upsampled and added to the first five channels of the output, producing M_t^{est}, N_t^{est} and D_t^{est}. The only exception is AO_t^{est}, which is inferred from scratch, as there is no low resolution input AO map.

4.2 Training Data

The training and validation data consists of images of isosurfaces from different timesteps and multi-resolution versions of the Ejecta dataset. This dataset stems from a particle-based simulation of a supernova that was resampled to a grid with resolutions from 256^3 to 1024^3. We choose this test suite because it contains isosurfaces showing many different geometric structures, ranging from very small details to rather smooth low-frequency parts. Of these, we rendered 500 sequences, each consisting of 10 frames. For each sequence, two views are selected at random and used as start view and end view of a smooth camera path. Eight additional in-between views are then computed along the path, and used to render corresponding frames at a low image resolution of 128^2, see Figure 5 for examples. 5000 sub-regions with a spatial size of 32^2 (for the low image resolution) and a temporal length of 10 are randomly cropped from the initial sequences, and are split into training (80%) and validation (20%) data. By rejection sampling we ensure that in each sub-region at least 50% of the pixels show a surface hit. The smaller spatial size is needed to fit multiple inputs at once into the memory during training and benefit from batch processing in the optimizer. The number of timesteps is kept the same. On a single Nvidia GeForce GTX 1080 Ti, the networks are trained for 100 to 500 epochs with training times from 3 to 18 hours, depending on the used cost function.

At this point, it is worth noting that the low resolution input images are directly generated by the raycaster. This is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

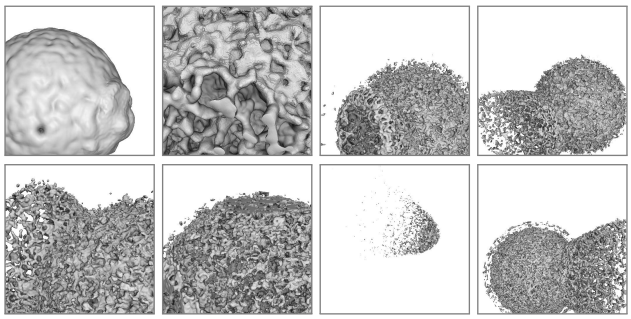


Fig. 5: Example images that are used to train our networks.

substantially different from common practice in image and video super-resolution [36], [40], where the inputs are low-pass filtered and downscaled versions of the original high resolution images. Thus, in our case the input images contain a huge amount of aliasing due to sub-sampling of the volumetric field, which poses a challenging task for the network.

The ground truth AO at a surface point is computed by sampling random directions on the hemisphere and testing for intersections between rays along these directions and the isosurface. This gives a much higher visual quality than screen-space AO, which tests samples against surface points based on screen-space depth.

To infer the current frame, the network takes the previous high resolution prediction O^{est}_{t-1} as input. Since the previous frame is not available in the first frame of a sequence, we evaluated different options to initialize the first previous high resolution input: Zeroing all entries, default setting to mask=0, normal=$[0,0,1]$, AO=1, and an upscaled version of the current input. Since we did not experience any noticeable difference between the three variants, we used the first and most simple option to train the network.

4.3 Loss Function Characteristics

The losses for training different super-resolution networks are obtained by using different weighted combinations of the individual losses described in subsection 3.3. Table 1 shows the specific weight combinations that are used for the networks we have analysed in our work.

For the networks trained with the losses in Table 1, Figure 6 shows a visual comparison of the surface structures (without AO) they infer from a given low resolution input image. In a number of tests we have confirmed the representativeness of these results, regardless of the type of isosurface and variations in the loss function weights. In subsection 5.2, we provide a quantitative evaluation that supports these findings.

All networks make use of a temporal loss $\mathscr{L}_{X,\text{temp}}$ to reduce flickering between successive frames. Due to the warping of the previous image, however, the use of a temporal loss can introduce smoothing. By changing the weighting between the temporal loss and the other losses, more focus can be put on either sharp details or improved temporal coherence.

As a baseline for comparison, we use a network that performs super-resolution on the low resolution color images, only including Phong shading but no AO. This network—"Shaded"—receives an RGB color image and a mask, and outputs the upscaled versions. In particular, this is different from our proposed upscaling process, where the inputs to the network comprise geometry, i.e., a depth and a normal field, and the final shading is performed in a deferred pass on the inferred high resolution normal field. Our experiments

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.

have shown that the network "Shaded" infers the best results if it utilizes a combination of perceptual and adversarial loss. The exact loss function configuration is given in Table 1.

Figure 6, however shows that the "Shaded" network produces color distortions and over-blurring (see also Figure 2). The visual quality of this network falls consistently below the quality of the networks trained on geometry. This result supports our strategy to let the network learn upscaling the depth and normal fields, and shade the image in a deferred pass. The following networks all follow this strategy.

The second network we evaluate is "L_1-color", which is trained with L_1 loss on colors. It acts on depths and normals, yet it lets the loss function consider the colors after deferred shading. Apparently, this deteriorates the quality of the inferred output and produces rather washed out results.

The networks "L_1-geometry", "Perceptual' and "GAN" also train on depths and normals, yet they work with the mask, depth, normal and AO maps in the loss function. Thus, all three networks are capable of focusing more on the geometric properties of isosurfaces rather than their appearance. This is confirmed by our results, which show that these networks can infer far more details from the low resolution inputs.

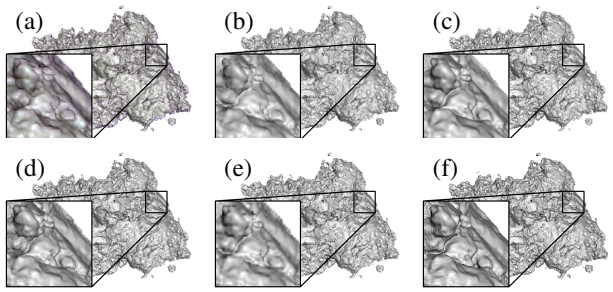


Fig. 6: Visual comparison of networks with different loss function configurations: (a) Shaded, (b) L_1-color, (c) L_1-geometry (our final model), (d) Perceptual, (e) GAN, (f) ground truth

The network "L_1-Geometry" trains only with L_1-losses on all input channels, i.e., mask, depth, normal, and AO. Adding a perceptual loss on the normal and AO fields, i.e., network "Perceptual", doesn't lead to any visual differences (Figure 6). We attribute this to the fact that the VGG-19 network was trained on color images and does not explicitly consider the relationships between geometric variations—given by the depth and normal fields—and the shaded output. We also noticed that the training time increases about a factor of six when using the VGG-19 network. Even though this drawback can probably be weakened by using fewer layers of the VGG-19 network, we still expect a significant performance loss if quality is maintained.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

With network "GAN", we evaluate the influence of an adversarial loss on reconstruction fidelity. In our experiments, however, the "GAN" network produces more high-frequent details that actually decrease the quality of the reconstruction, and it significantly increases both training time and memory requirements by the discriminator. As a consequence, we decided to focus on directly supervised networks instead of GAN variants in this work.

In summary, of all tested networks the "L_1-geometry" network with minor objective on temporal coherence shows superior performance, both in terms of training time and reconstruction quality. This network does only see shaded colors in the temporal coherence loss during the training process, and is thus forced to focus primarily on the reconstruction of geometry.

5 Evaluation

We compare the "L_1-geometry" network with bi-linear and bi-cubic filtering as baseline methods on unseen test data. To validate how good the trained networks generalize to new isosurfaces, we let the network also upscale low resolution isosurface images of volumes which were never shown during training (see Figure 7): A CT scan of a human skull with a resolution of 256^3, a CT scan of a human thorax at 256^3, and a numerical simulation of a Richtmyer-Meshkov instability at 1024^3.

5.1 Qualitative Evaluation

To analyse the visual quality of network-based upsampling, we used the "L_1-geometry" network to upsample images of isosurfaces in the Ejecta dataset from perspectives that were never seen during training (see Figure 1 and Figure 7). The results indicate that the network can effectively infer the surface structures from the structures it has learned during training. In particular, compared to bi-linear and bi-cubic upsampling the network infers meaningful details in line with the geometric surface properties.

To demonstrate the reconstruction quality even for isosurfaces in datasets that were never seen by the network during training, we compare the results of bi-linear and network-based upscaling to the ground truth images using the datasets introduced before. The accompanying video shows the results when the network considers frame-to-frame coherence during animations. For the human skull, which exhibits smooth surfaces similar to Ejecta, the inference results are very close to the ground truth. A similar result is obtained when upscaling images of isosurfaces in the Richtmyer-Meshkov dataset. Despite the many fine-grained geometric details, the network can reconstruct the isosurface in a fairly accurate way. The network, however, faces difficulties when applied to images of an isosurface exhibiting geometric details at a higher frequency than it was trained on, as for example in the Thorax dataset. In this case, the network cannot reconstruct all fine-scale details accurately and rather blurs out the missing surface structures.

5.2 Quantitative Evaluation

To quantify the error that is introduced by learning-based isosurface reconstruction, all datasets are rendered 500 times from different views, and for each upscaled image the PSNR and SSIM are computed between the high resolution ground truth rendering and the reconstruction. The results, in the form of the medians, the n% quantiles, and the range of outliers, are shown in Figure 8 and Figure 9.

The PSNR is computed as

$$\mathrm{PSNR}(O_t^{est}, O_t^{GT}) = -10\log_{10}(||O_t^{est} - O_t^{GT}||_2^2), \qquad (8)$$

where O_t^{est} and O_t^{GT} are the reconstructed and ground truth images, respectively. The SSIM is defined as

$$\mathrm{SSIM}(O_t^{est}, O_t^{GT}) = \frac{(2\mu_{est}\mu_{GT} + c_1)(2\sigma_{est,GT} + c_2)}{(\mu_{est}^2 + \mu_{GT}^2 + c_1)(\sigma_{est}^2 + \sigma_{GT}^2 + c_2)}, \qquad (9)$$

where μ_{est} and μ_{GT} are the average values of O_t^{est} and O_t^{GT}, σ_{est}^2 and σ_{GT}^2 are the variances of O_t^{est} and O_t^{GT}, $\sigma_{est,GT}$ is the covariance between O_t^{est} and O_t^{GT}, and c_1 and c_2 are two small constants to avoid division by zero.

The first quantitative evaluation sheds light on the reconstruction quality of the networks that were trained using different loss functions. Figure 8 confirms the strength of the "L_1-geometry" network compared to the alternative variants presented in subsection 4.3. These results back up our decision to chose the "L_1-geometry" network as our preferred model, and to use it in the following quantitative analysis of the reconstruction quality.

In Figure 9, we analyze the reconstruction quality for both the normal and depth field, and further asses how well local illumination values and AO values can be inferred from the normal fields. Note here that AO values are not available at the low resolution input samples, because their simulation would be far too costly to maintain interactive frame rates. Thus, bi-linear and bi-cubic upscaling cannot generate high resolution AO. Therefore, the error measures were applied on the shaded color output, once without AO and including measures for bi-linear and bi-cubic, and once with AO and excluding bi-linear and bi-cubic upscaling. As can be seen, the PSNR and SSIM always slightly decrease when AO is added. This, however, is not particularly surprising, since one more quantity is inferred by the network that could introduce some error.

It can be seen that the "L_1-geometry" network always achieves better results than the other alternatives, even for isosurfaces of volumes that were not seen during training. This indicates the principal capability of neural networks to generalize to new data. Since the PSNR cannot capture the "sharpness" of the image very well [7], the differences are rather moderate when using the PSNR as quality metric. However, the differences become significant when using SSIM as quality metric. This is further confirmed by the images in subsection 5.1, which also show far better perceptual quality of network-based reconstruction compared to bi-linear and bi-cubic upscaling.

As described in subsection 4.2, the "L_1-geometry" network is trained solely on Ejecta (see Figure 5). This dataset does not contain the specific structures and rather smooth AO distribution observed in the Cloud dataset, a dataset of 12 clouds by Kallweit *et al.* [55]. An example of a volume typical for this dataset can be seen in Figure 10. We therefore re-trained the network for 600 epochs on images of isosurfaces in the Cloud dataset. The statistics in Figure 9 indicate that the re-trained network performs substantially better on the Cloud dataset, both in terms of PSNR and SSIM. This is also confirmed by Figure 10, which shows the inference results of the "L_1-geometry" network, once trained on Ejecta and once on Cloud. The comparison to the ground truth image indicates strong improvement of the reconstruction accuracy when specializing the network on a specific dataset.

The evaluation so far shows that network-based upscaling outperforms bi-linear and bi-cubic upscaling, yet it does not provide information about the number of erroneously inferred pixels and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

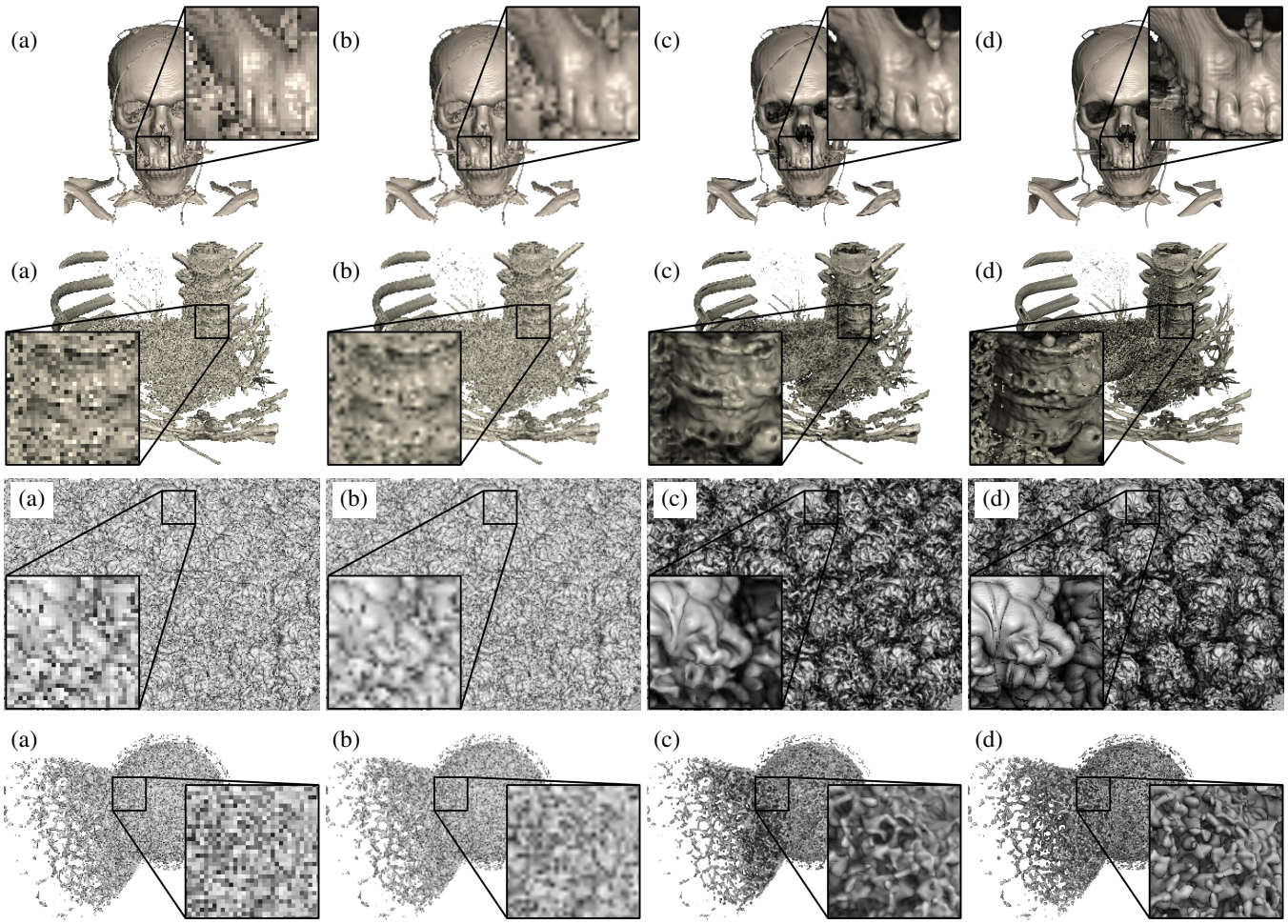


Fig. 7: Comparison of upscaling quality: (a) input, (b) bi-linear, (c) our network, (d) ground truth on the Skull, Thorax, Richtmyer-Meshkov and Ejecta dataset (top to bottom).

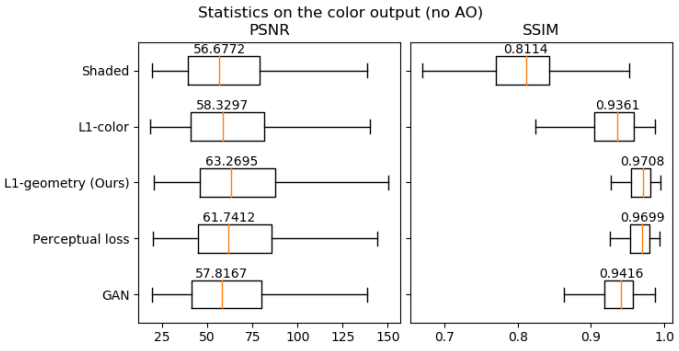


Fig. 8: Reconstruction quality of networks trained on the Cloud dataset using different loss functions. The orange line shows the median and the box outlines the 25% and 75% quantile.

their deviation from the ground truth. To investigate this aspect, we further analyze how many "good pixels" and "bad pixels" exist using the Regression Error Characteristic (REC) curves [56]. Given a certain error tolerance, REC curves show the percentage of pixels that are accurate according to the ground truth, i.e. $REC(\tau) = P(|x^{est} - x^{GT}| \leq \tau)$. It is widely used as a better performance description of a predictive model comparing to error statistics like PSNR, because the performance is illustrated across the range of errors. In our cases, we evaluate the REC curves on the normal, depth, AO and the gray-scale pixel intensity after shading without AO for our model in together with bi-linear and bi-cubic upscaling in Figure 11(a). First and foremost, the measurements indicate that erroneously inferred pixels cannot be avoided by any of the methods The pixel-wise difference images in Figure 11(b) and (c) indicate that these errors are predominantly introduced along the silhouettes and cavities, where sometimes the network cannot accurately extrapolate the sharp transitions. Second, it is shown that network-based inference can reduce the number of pixels with a certain deviation significantly. With the Ejecta dataset as example, bi-linear upscaling generates an image in which 16% of the pixels have an absolute error to the ground truth intensity value larger than 0.1, with a maximal error of 1 between completely dark and completely bright. When using network-based upscaling, this number is reduced to 12.4%. Similar results hold for all other test datasets, demonstrating the superior quality of network-based upscaling.

5.3 Timings

To evaluate the performance of isosurface super-resolution, we compare it to volumetric ray-casting on the GPU using an empty-space acceleration structure. Rendering times for the shown isosurfaces in the four test datasets are given in Table 2, for a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

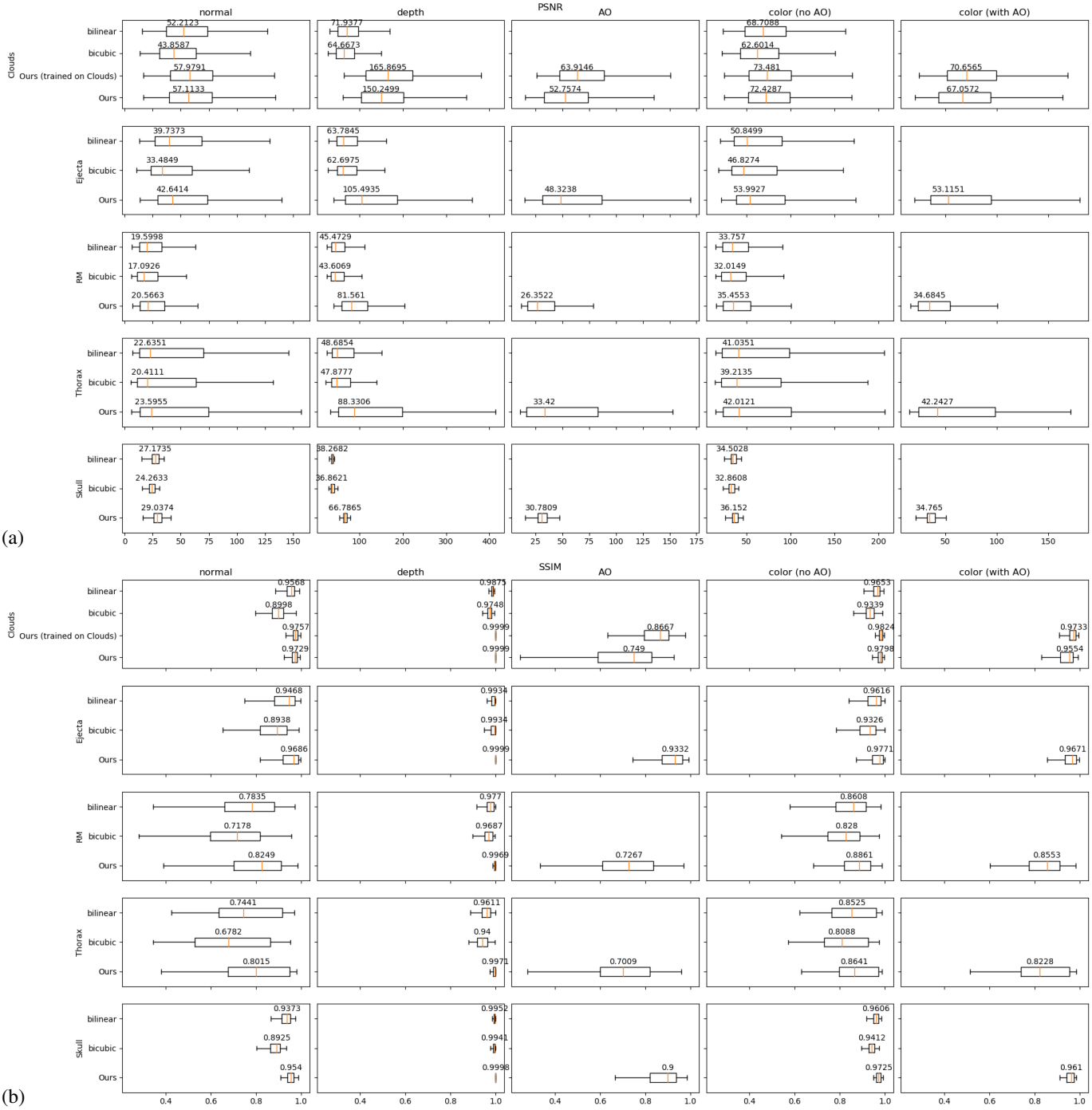


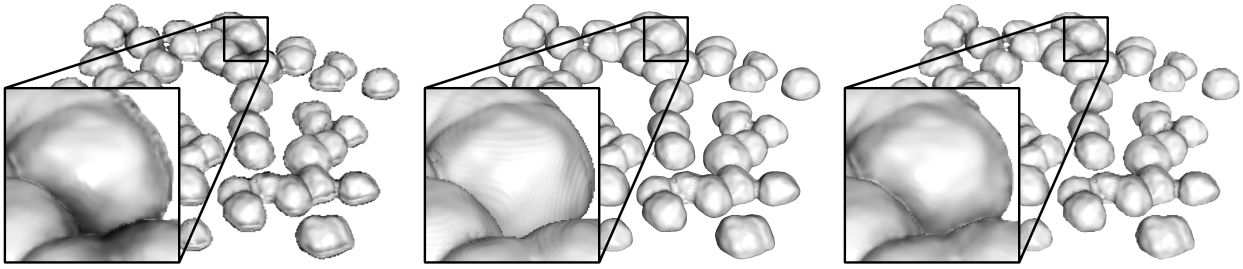
Fig. 9: (a) PSNR and (b) SSIM on the training and test examples for different upscaling approaches. Box plots show medians, quantiles and outliers. Since bi-linear and bi-cubic upsampling cannot infer AO, error measures are not available for these fields.

viewport size of 1920x1080. Each dataset was rendered from a number of different views along a pre-recorded path, so that the dataset covers the entire viewport. The isosurface renderer is implemented with Nvidia's GVDB library [57], an optimized GPU raytracer written in CUDA. The super-resolution network uses Pytorch. The timings were performed on a workstation running Windows 10, equipped with an Intel Xeon W-2123, 3.60Ghz, 8 logical cores, 64GB RAM, and a Nvidia RTX Titan GPU.

The table shows the time to render the ground truth image at full resolution with AO, the rendering times for the low resolution input without AO, and the time to perform super-resolution upscaling of the input using the "L_1-geometry" network. As the computation times for all three different quantities do not differ significantly from frame to frame, the average time is reported. The time to warp the previous image, perform screen-space shading, and IO between the renderer and the network are not included. For rendering the ground truth AO, 128 samples were taken. This gives reasonable results, but noise is still visible.

As one can see from Table 2, the time to simulate AO increases the computational cost significantly. Because the Ejecta dataset

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics



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

Fig. 10: (a) Our network trained on the Ejecta dataset cannot faithfully reproduce the smooth geometry and silhouettes in the Cloud dataset, as shown in the ground truth rendering (b). It tends to "overshoot" the AO values and produces artefacts at the boundaries. (c) By re-training the network on images of isosurfaces in the Cloud dataset, the reconstruction quality is improved.

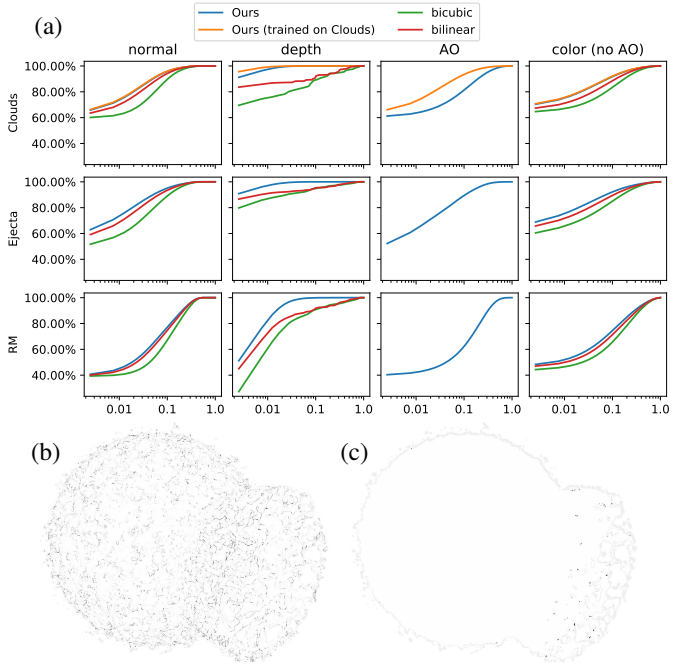


Fig. 11: (a) REC curves, the cumulative accuracy over the error tolerance on L_1-distance $|x^{est} - x^{GT}|$. A higher value is better. The spatial distribution of the pixel-wise error for the (b) normal and (c) depth map.

contains less empty blocks that can be skipped during rendering than the Richtmyer-Meshkov dataset, the computation time for the first hit (high resolution image without AO) is twice as high as that for the Richtmyer-Meshkov dataset. As expected, the time to evaluate the super-resolution network stays constant for all four datasets, as it only depends on the viewport size.

As an example, the total time to render the input and 4 times upscale images of isosurfaces in the Richtmyer-Meshkov dataset is $0.014s + 0.072s = 0.086s$. Hence, the network approximately takes the same time it takes to render the full resolution without AO ($0.088s$), but in this time also produces a smooth AO map. More prominently, for the Ejecta dataset, a high resolution rendering without AO takes $0.163s$, approximately 50% longer than rendering the low resolution version and upscaling it, which requires $0.103s$ in total. The latter version also provides AO "for free". Once AO

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

TABLE 2: Timings in seconds for rendering an isosurface in FullHD (1920x1080) resolution, averaged over 10 frames.

is included in the high resolution rendering, the rendering time increases to $18.6s$, hence the super-resolution outperforms the high resolution renderer by two orders of magnitudes.

6 Discussion

Our results demonstrate that deep learning-based inference has potential for upscaling tasks beyond classical image-based approaches. The trained network seems to infer well the geometric properties of isosurfaces in volumetric scalar fields. We believe this result is of theoretical interest on its own, and at the same time opens up new perspectives in a number of practical use cases. Even though it is not possible, in general, to predict the error that is introduced by the network, we believe there are two classes of applications were learning-based isosurface inference is eligible: Into the first class fall applications were a high resolution surface does not exist, for instance, because due to time and memory constraints only a low resolution volumetric field can be acquired. In such scenarios, learning-based inference might be able to predict where certain features can occur and, thus, can guide refined simulations or measurements. Regarding this use case, it will be important to investigate whether artificial neural networks can infer from a given high resolution image of an isosurface in a low resolution volume the isosurface rendering from the corresponding high resolution volume.

Into the second class fall applications were the user is willing to make tradeoffs in fidelity and speed, i.e., were reconstruction quality can be sacrificed for speed, at least temporarily or locally. In the following, we shed light on two such applications and demonstrate the practical usefulness of isosurface inference.

6.1 Use Cases

One application is the utilization of isosurface inference to support an interactive exploration of high resolution volume datasets. When the high resolution dataset cannot be rendered at interactive rates, it is common practice in many visualization tools to render a low

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

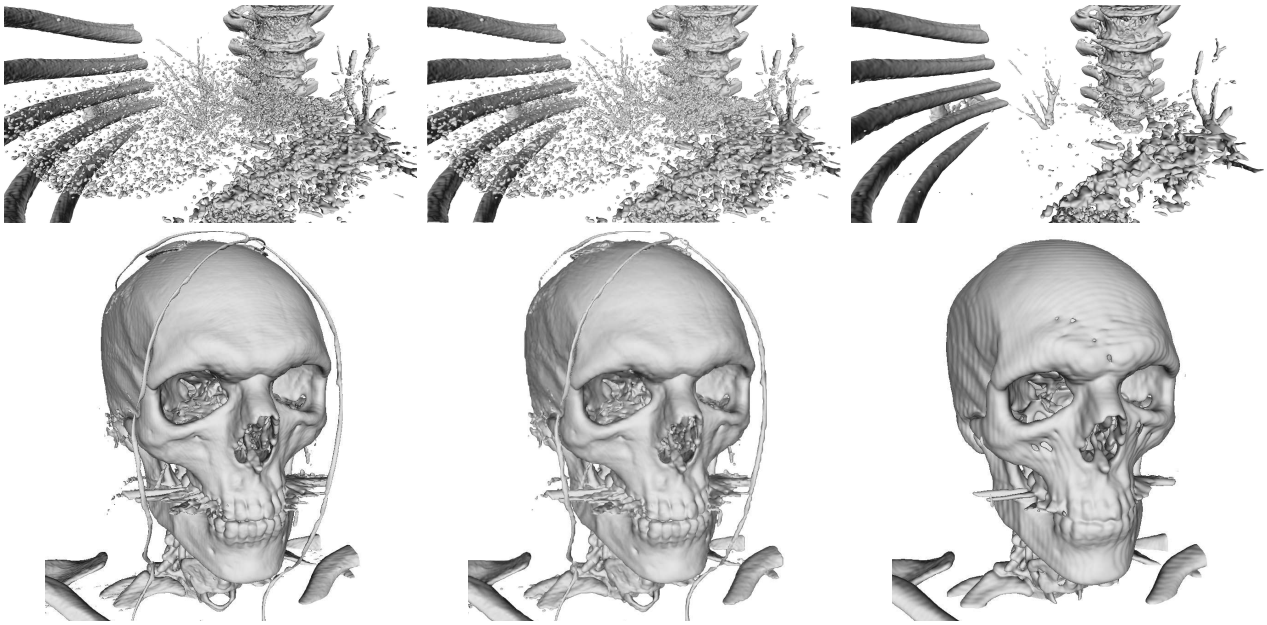


Fig. 12: Level of detail rendering for interactive exploration. Left: Rendering of isosurfaces in the original datasets at full image resolution. Middle: Rendering the isosurfaces in the original datasets at $1/4$ the image resolution and upscaling to full image resolution. Right: Rendering the isosurfaces in the low-pass filtered and down-sampled datasets with half the resolution in each dimension at full image resolution.

resolution version at full image resolution during interaction, and to switch back to the high resolution dataset once the camera stands still. This enables an interactive exploration, yet provides views in which many details are lost and even the topology of the isosurface can be corrupted due to the downsampling process that is used to generate the low resolution version (see 3rd column in Figure 12). In contrast, upscaling a low resolution image of the high resolution dataset (2nd column in Figure 12) generates a far more detailed view and preserves topology to a large extend. Notably, upscaling the low resolution image of the high resolution isosurface requires roughly the same time as rendering the low resolution isosurface at full image resolution.

The same principle can be applied in remote visualization, where in practice the bandwidth of the communication channel across which rendered images are transmitted often limits the streaming performance. Thus, the degree of interactivity often falls below what a user expects. To weaken this limitation, during interaction low resolution images of the dataset can be streamed to the client-side and upscaled using trained networks.

As a second use case we have integrated isosurface inference into foveated rendering. In foveated rendering, a focus region in the image is given by the falloff of acuity in the visual periphery. Since fine details can only be sensed within a small portion (5°) of the visual field, with increasing angular distance from the central region of visual stimulus, the number of samples can be reduced accordingly. This is exploited in foveated rendering to reduce the number of samples rendered in the peripheral region, by either upscaling low resolution renderings [58] or interpolating between a sparse set of initial samples in this region [59].

To use network-based inference in foveated rendering, we utilize renderings at different image resolution. While the image is rendered at full resolution in the region of highest acuity, it is rendered at $1/4$ this resolution in the exterior (taking over the full resolution samples in the focus region) and upscaled by the super-resolution network. The two images are then smoothly blended together. As shown in Figure 13a,b, the difference between the full resolution rendering in the focus regions and the upscaled low resolution version is almost indistinguishable, yet a significant lower number of samples is required to generate the final image. This number can be further reduced by generating and blending multiple images at ever lower resolution, i.e., by rendering images using $1/2$ and $1/4$ the full resolution and using networks to upscale to the full resolution.

In foveated rendering, the reconstruction error that may be introduced by the network is fully acceptable, since it only occurs in the region outside the users central region of visual stimulus. Yet as we have shown in this work, learning-based upscaling can far better maintain geometric and topological features than other techniques like bi-linear upscaling. Thus, transitions between multiple resolutions are far less pronounced and can be removed more effectively.

6.2 Conclusion and Future Work

We have introduced and analyzed a deep learning technique for isosurface super-resolution with AO. The proposed recurrent network architecture with temporally coherent adversarial training makes it possible to infer highly detailed images from low resolution input renderings. The network reconstructs high resolution images of isosurfaces including ambient occlusions at a performance that is significantly faster than that of an optimized ray-caster at full resolution. We have published the source code for training and inference as well as the trained networks and datasets on https://github.com/shamanDevel/IsosurfaceSuperresolution.

The quality of upscaling seems to indicate that not a specific isosurface is learned, but rather that the network is able to generalize, i.e., to infer the geometric properties of isosurfaces in volumetric scalar fields. Yet especially when the network's learned

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2956697, IEEE Transactions on Visualization and Computer Graphics

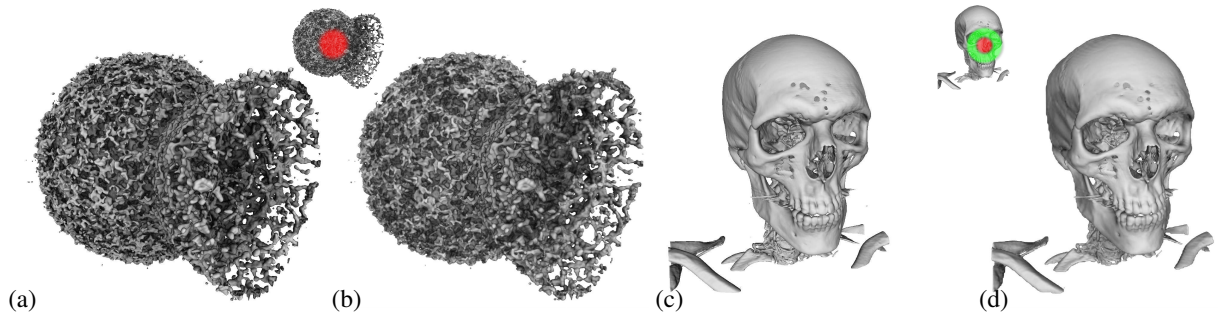


Fig. 13: Foveated rendering. Rendering of the original datasets in (a) and (c). (b) In the region of highest acuity (indicated by the red circle in the inlay), the isosurface is sampled at full resolution and smoothly blended with an upscaled image at $1/4$ the resolution. (d) Same as (b), but region of highest acuity is decreased and blended with upscaled images at $1/2$ (green area) and $1/4$ (exterior) the resolution. In (b) and (d), respectively, 16% and 11% of the of the samples in (a) and (c) are used.

representation is not sufficient due to limited training sample variation, network-based reconstruction can lead to distortions in the inferred structures. We have demonstrated that this can be counteracted by specializing the network on certain types of isosurfaces, such as they occur in certain types of simulated or measured physical fields. The quality of the reconstruction can be substantially improved if the network is given the chance to see the type of isosurface it is used to infer. Nevertheless, it is arguable that network-based inference should be used carefully in applications were highest accuracy is required, e.g., in medical imaging.

On the other hand, we have shown applications where a reconstruction error is tolerable, e.g., during interactive navigation in large datasets and in foveated rendering. In such applications, network-based inference provides a very effective means to balance between reconstruction quality and performance.

The proposed method only represents a first step towards learning-based data inference, and we see numerous promising and interesting avenues for future research. Among others, it will be important to analyze how sparse the input data can be so that a network can still infer on the geometry of the underlying structures. Furthermore, we will shed light on the inference of additional rendering effects such as soft shadows. Finally, we will investigate the extension of our approach to support transparency and multiple-scattering effects, by going beyond image-based inference and integrating volumetric representations in the training and inference steps.

Acknowledgments

This work is supported by the ERC Starting Grant *realFlow* (StG-2015-637014).

References

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[2] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos, "Video super-resolution with convolutional neural networks," *IEEE Transactions on Computational Imaging*, vol. 2, no. 2, pp. 109–122, 2016.

[3] Z. Zhou, Y. Hou, Q. Wang, G. Chen, J. Lu, Y. Tao, and H. Lin, "Volume upscaling with convolutional neural networks," in *Proceedings of the Computer Graphics International Conference*, ser. CGI '17. New York, NY, USA: ACM, 2017, pp. 38:1–38:6. [Online]. Available: http://doi.acm.org/10.1145/3095140.3095178

[4] J. Han and C. Wang, "Tsr-tvd: Temporal super-resolution for time-varying data analysis and visualization," *IEEE Transactions on Visualization and Computer Graphics (to appear)*, 2019.

[5] W. He, J. Wang, H. Guo, K.-C. Wang, H.-W. Shen, M. Raj, Y. S. G. Nashed, and T. Peterka, "Insitunet: Deep image synthesis for parameter space exploration of ensemble simulations," *IEEE Transactions on Visualization and Computer Graphics (to appear)*, 2019.

[6] M. S. Sajjadi, R. Vemulapalli, and M. Brown, "Frame-recurrent video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6626–6634.

[7] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[8] M. Levoy, "Display of surfaces from volume data," *IEEE Comput. Graph. Appl.*, vol. 8, no. 3, pp. 29–37, May 1988. [Online]. Available: https://doi.org/10.1109/38.511

[9] J. Danskin and P. Hanrahan, "Fast algorithms for volume ray tracing," in *Proceedings of the 1992 Workshop on Volume Visualization*, ser. VVS '92. New York, NY, USA: ACM, 1992, pp. 91–98. [Online]. Available: http://doi.acm.org/10.1145/147130.147155

[10] L. M. Sobierajski and A. E. Kaufman, "Volumetric ray tracing," in *Proceedings of the 1994 Symposium on Volume Visualization*, ser. VVS '94. New York, NY, USA: ACM, 1994, pp. 11–18. [Online]. Available: http://doi.acm.org/10.1145/197938.197949

[11] M. Wan, A. Kaufman, and S. Bryson, "High performance presence-accelerated ray casting," in *Proceedings of the conference on Visualization '99*, 1999, pp. 379–386.

[12] M. Sramek, "Fast surface rendering from raster data by voxel traversal using chessboard distance," in *Proceedings Visualization '94*, Oct 1994, pp. 188–195.

[13] M. Hadwiger, A. K. Al-Awami, J. Beyer, M. Agus, and H. Pfister, "Sparseleap: Efficient empty space skipping for large-scale volume rendering," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 974–983, Jan 2018.

[14] J. Kruger and R. Westermann, "Acceleration techniques for gpu-based volume rendering," in *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, ser. VIS '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 38–. [Online]. Available: https://doi.org/10.1109/VIS.2003.10001

[15] M. Hadwiger, C. Sigg, H. Scharsach, K. Bühler, and M. H. Gross, "Real-time ray-casting and advanced shading of discrete isosurfaces," *Comput. Graph. Forum*, vol. 24, pp. 303–312, 2005.

[16] T. Klein, M. Strengert, S. Stegmaier, and T. Ertl, "Exploiting frame-to-frame coherence for accelerating high-quality volume raycasting on graphics hardware," in *IN: PROCEEDINGS OF IEEE VISUALIZATION '05*. IEEE, 2005, pp. 223–230.

[17] C. Braley, R. Hagan, Y. Cao, and D. Gračanin, "Gpu accelerated isosurface volume rendering using depth-based coherence," in *ACM SIGGRAPH ASIA 2009 Posters*, 2009, pp. 42:1–42:1.

[18] E. Gobbetti, F. Marton, and J. A. Iglesias Guitián, "A single-pass gpu ray casting framework for interactive out-of-core rendering of massive volumetric datasets," *The Visual Computer*, vol. 24, no. 7, pp. 797–806, Jul 2008. [Online]. Available: https://doi.org/10.1007/s00371-008-0261-9

[19] M. Treib, K. Bürger, F. Reichl, C. Meneveau, A. Szalay, and R. Westermann, "Turbulence visualization at the terascale on desktop pcs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2169–2177, Dec 2012.

[20] F. Reichl, M. G. Chajdas, K. Bürger, and R. Westermann, "Hybrid Sample-based Surface Rendering," in *Vision, Modeling and Visualization*,

M. Goesele, T. Grosch, H. Theisel, K. Toennies, and B. Preim, Eds. The Eurographics Association, 2012.

[21] T. Fogal, A. Schiewe, and J. Kruger, "An analysis of scalable gpu-based ray-guided volume rendering," in *2013 IEEE Symposium on Large-Scale Data Analysis and Visualization (LDAV)*, vol. 2013, 10 2013, pp. 43–51.

[22] J. Beyer, M. Hadwiger, and H. Pfister, "State-of-the-art in gpu-based large-scale volume visualization," *Computer Graphics Forum*, vol. 34, no. 8, pp. 13–37, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12605

[23] S. Zhukov, A. Iones, and G. Kronin, "An ambient light illumination model," in *Rendering Techniques '98*, G. Drettakis and N. Max, Eds. Vienna: Springer Vienna, 1998, pp. 45–55.

[24] E. Penner and R. Mitchell, "Isosurface ambient occlusion and soft shadows with filterable occlusion maps," in *Proceedings of the Fifth Eurographics / IEEE VGTC Conference on Point-Based Graphics*, ser. SPBG'08. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2008, pp. 57–64. [Online]. Available: http://dx.doi.org/10.2312/VG/VG-PBG08/057-064

[25] F. Hernell, P. Ljung, and A. Ynnerman, "Efficient ambient and emissive tissue illumination using local occlusion in multiresolution volume rendering," in *Proceedings of the Sixth Eurographics / Ieee VGTC Conference on Volume Graphics*, ser. VG'07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 1–8. [Online]. Available: http://dx.doi.org/10.2312/VG/VG07/001-008

[26] T. Ropinski, J. Meyer-Spradow, S. Diepenbrock, J. Mensmann, and K. Hinrichs, "Interactive volume rendering with dynamic ambient occlusion and color bleeding," *Computer Graphics Forum*, vol. 27, no. 2, pp. 567–576, 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2008.01154.x

[27] D. Jönsson, E. Sundén, A. Ynnerman, and T. Ropinski, "A survey of volumetric illumination techniques for interactive volume rendering," *Comput. Graph. Forum*, vol. 33, pp. 27–51, 2014.

[28] M. Mittring, "Finding next gen: Cryengine 2," in *ACM SIGGRAPH 2007 Courses*, ser. SIGGRAPH '07. New York, NY, USA: ACM, 2007, pp. 97–121. [Online]. Available: http://doi.acm.org/10.1145/1281500.1281671

[29] L. Bavoil, M. Sainz, and R. Dimitrov, "Image-space horizon-based ambient occlusion," in *ACM SIGGRAPH 2008 Talks*, ser. SIGGRAPH '08. New York, NY, USA: ACM, 2008, pp. 22:1–22:1. [Online]. Available: http://doi.acm.org/10.1145/1401032.1401061

[30] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1646–1654.

[31] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate superresolution," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2017, p. 5.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[33] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[34] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[35] T. Tong, G. Li, X. Liu, and Q. Gao, "Image super-resolution using dense skip connections," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4799–4807.

[36] M. S. Sajjadi, B. Scholkopf, and M. Hirsch, "Enhancenet: Single image super-resolution through automated texture synthesis," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4491–4500.

[37] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia, "Detail-revealing deep video super-resolution," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[38] D. Liu, Z. Wang, Y. Fan, X. Liu, Z. Wang, S. Chang, and T. Huang, "Robust video super-resolution with learned temporal dynamics," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2526–2534.

[39] Y. Jo, S. W. Oh, J. Kang, and S. J. Kim, "Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3224–3232.

[40] M. Chu, Y. Xie, L. Leal-Taixé, and N. Thuerey, "Temporally coherent gans for video super-resolution (tecogan)," *arXiv preprint arXiv:1811.09393*, 2018.

[41] R. Liao, X. Tao, R. Li, Z. Ma, and J. Jia, "Video super-resolution via deep draft-ensemble learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 531–539.

[42] Y. Xie, E. Franz, M. Chu, and N. Thuerey, "tempoGAN: A temporally coherent, volumetric GAN for super-resolution fluid flow," *ACM Trans. Graph.*, vol. 37, no. 4, 2018.

[43] C. R. Alla Chaitanya, A. S. Kaplanyan, C. Schied, M. Salvi, A. Lefohn, D. Nowrouzezahrai, and T. Aila, "Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder," *ACM Transactions on Graphics*, vol. 36, pp. 1–12, 07 2017.

[44] M. Mara, M. McGuire, B. Bitterli, and W. Jarosz, "An efficient denoising algorithm for global illumination," in *Proceedings of High Performance Graphics*. New York, NY, USA: ACM, jul 2017.

[45] O. Nalbach, E. Arabadzhiyska, D. Mehta, H.-P. Seidel, and T. Ritschel, "Deep shading: Convolutional neural networks for screen space shading," *Comput. Graph. Forum*, vol. 36, no. 4, pp. 65–78, Jul. 2017. [Online]. Available: https://doi.org/10.1111/cgf.13225

[46] M. Berger, J. Li, and J. A. Levine, "A generative model for volume rendering," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, pp. 1636–1650, 2017.

[47] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1. IEEE, 2001, pp. I–I.

[48] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[49] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2414–2423.

[50] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," in *Advances in neural information processing systems*, 2016, pp. 658–666.

[51] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711.

[52] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[53] D. Chen, J. Liao, L. Yuan, N. Yu, and G. Hua, "Coherent online video style transfer," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1105–1114.

[54] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[55] S. Kallweit, T. Müller, B. McWilliams, M. Gross, and J. Novák, "Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks," *ACM Trans. Graph. (Proc. of Siggraph Asia)*, vol. 36, no. 6, Nov. 2017. [Online]. Available: http://doi.acm.org/10.1145/3130800.3130880

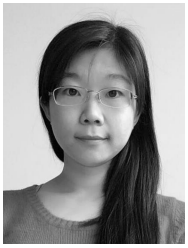
[56] J. Bi and K. P. Bennett, "Regression error characteristic curves," in *ICML*, 2003.

[57] R. K. Hoetzlein, "GVDB: Raytracing Sparse Voxel Database Structures on the GPU," in *Eurographics/ ACM SIGGRAPH Symposium on High Performance Graphics*, U. Assarsson and W. Hunt, Eds. The Eurographics Association, 2016.

[58] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder, "Foveated 3d graphics," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 164, 2012.

[59] M. Stengel, S. Grogorick, M. Eisemann, and M. A. Magnor, "Adaptive image-space sampling for gaze-contingent real-time rendering," *Comput. Graph. Forum*, vol. 35, pp. 129–139, 2016.

Sebastian Weiss received the M.Sc. degree from the Technical University of Munich in 2018. Currently, he is a doctoral student of computer science, Technical University of Munich. His research interests include volume visualization, deep learning and high performance GPU programming.



Mengyu Chu received the M.Eng. degree from Zhejiang University, China in 2014. Currently, she is a doctoral student of computer science, Technical University of Munich. Her research interests include fluid simulations and deep learning.

Nils Thuerey is an Associate-Professor at the Technical University of Munich (TUM). He works in the field of computer graphics, where a central theme of his research are physics simulations and deep learning algorithms. He received a tech-Oscar from the AMPAS in 2013 for his research on controllable smoke effects. He worked for three years as a post-doc at ETH Zurich and as R&D lead at ScanlineVFX, before starting at TUM in October 2013.

Rüdiger Westermann studied computer science at the Technical University Darmstadt and received his Ph.D. in computer science from the University of Dortmund, both in Germany. In 2002, he was appointed the chair of Computer Graphics and Visualization at TUM. His research interests include scalable data visualization and simulation algorithms, GPU computing, real-time rendering of large data, and uncertainty visualization.