EUROGRAPHICS 2022 / R. Chaine and M. H. Kim
(Guest Editors)

Volume 41 (2022), Number 2

Deep Reconstruction of 3D Smoke Densities from Artist Sketches

Byungsoo Kim[1], Xingchang Huang[1,2], Laura Wuelfroth[1], Jingwei Tang[1], Guillaume Cordonnier[1,3], Markus Gross[1], Barbara Solenthaler[1]

[1]ETH Zurich [2]Max Planck Institute for Informatics [3]Inria, Université Côte d'Azur, France

Figure 1: *Our method reconstructs volumetric density fields from sketch inputs using an updater CNN. The efficient reconstruction at test time enables interactive authoring and editing of arbitrary smoke shapes. From left to right we show results for the* smoke jet*,* puppy *and* dissolving character *examples.*

Abstract

Creative processes of artists often start with hand-drawn sketches illustrating an object. Pre-visualizing these keyframes is especially challenging when applied to volumetric materials such as smoke. The authored 3D density volumes must capture realistic flow details and turbulent structures, which is highly non-trivial and remains a manual and time-consuming process. We therefore present a method to compute a 3D smoke density field directly from 2D artist sketches, bridging the gap between early-stage prototyping of smoke keyframes and pre-visualization. From the sketch inputs, we compute an initial volume estimate and optimize the density iteratively with an updater CNN. Our differentiable sketcher is embedded into the end-to-end training, which results in robust reconstructions. Our training data set and sketch augmentation strategy are designed such that it enables general applicability. We evaluate the method on synthetic inputs and sketches from artists depicting both realistic smoke volumes and highly non-physical smoke shapes. The high computational performance and robustness of our method at test time allows interactive authoring sessions of volumetric density fields for rapid prototyping of ideas by novice users.

CCS Concepts

• ***Computing methodologies*** → ***Shape modeling; Neural networks;***

1. Introduction

Digital content creation requires a constant and iterative interplay between artists and the generated data. At the core of these workflows, digital artists first manually generate concept sketches in the pre-visualization stage, which are then transferred into 3D models. Creating content in this manner requires a significant and costly authoring effort. To bridge the gap between early-stage prototyping and visual realization, sketch-based modeling methods have been developed that can reconstruct 3D geometries from 2D input sketches.

Many existing approaches target a particular class of objects and leverage domain-specific geometric constraints. Accordingly, they make use of properties such as symmetry, smoothness, or adjacency [EBC*15, MHZ*15]. To reconstruct non-flat silhouettes or higher-frequency information, various sketching paradigms have been explored to inject curvature variation across surfaces or sharp features [JHR*15, LPL*17, LSGV18]. Particularly challenging is the integration of domain knowledge for creating volumetric density fields, such that intrinsic flow details are captured. Existing sketch-based cloud modeling techniques circumvent this by adding noise to the shape boundary [SBRS10] or by deferring the synthe-

sis of high-frequency details to the rendering stage [WBC08], at the cost of reduced control.

We address this limitation by integrating low- and mid-frequency flow structures already in the modeling step. We focus on the general class of turbulent density volumes, including physically plausible smoke keyframes and artistic smoke creatures, and target the method to early-stage prototyping of effects and communication of ideas. We impose a shading-based sketching style that follows the principles of effects drawing depicted in [Gil12]. Our data-driven method then infers 3D densities from the sketch inputs in real-time (Figure 1 and Figure 2), allowing on-the-fly testing and editing. Such a sketch-based reconstruction has to handle a set of unique challenges, as the problem is ill-posed due to the sparsity of sketch data and the ambiguity when inferring another dimension [DAI*18, WCPM18, LPL*18]. Previous work indicates that the use of convolutional neural networks is especially powerful in such tasks [DAI*18, LSS*19, MST*20].

In our work, we present a neural network architecture that keeps the artist in the loop. The method uses an iterative strategy [DAI*18, LTJ18], which alternately refines the shape for each input viewpoint (e.g., front and side). This design choice further allows an artist to dynamically add sketches from further viewpoints if needed, and hence enables fine-grained control over the final shape. As opposed to previous CNN-based reconstruction methods (such as [DAI*18]), our method can reconstruct richer flow details and is more robust to variations of the input. This is attributed to the following key contributions that are essential for the robust reconstruction of smoke volumes:

- A real-time, iterative refinement method that can handle arbitrary viewpoints and preserves previous views in the reconstruction.
- A differentiable sketch generator for smoke volumes that mimics the principles of effects drawing, and corresponding loss function.
- A data augmentation technique accounting for sketch style variations that significantly increases the robustness of the reconstruction.
- A post-processing network based on multi-pass GAN [WXCT19] that extends the sketched low- and mid-frequency details with fine structures.

We evaluate the method on synthetically generated sketches, validate the quality of the results and ease of use by accompanying user studies and comparisons with previous work, and demonstrate the following demo applications:

- 3D results reconstructed from artist sketches.
- Real-time authoring sessions of artists displaying the ease-of-use creation and editing of volumetric density fields, which demonstrates a gentle learning curve even for novices and non-artists.
- Density keyframe generation for smoke animation control.

2. Related Work

We focus our discussion on deep learning based 3D modeling from sketches. For a more general overview of sketch-based systems we refer the interested reader to [CSGC16, KYZ14].

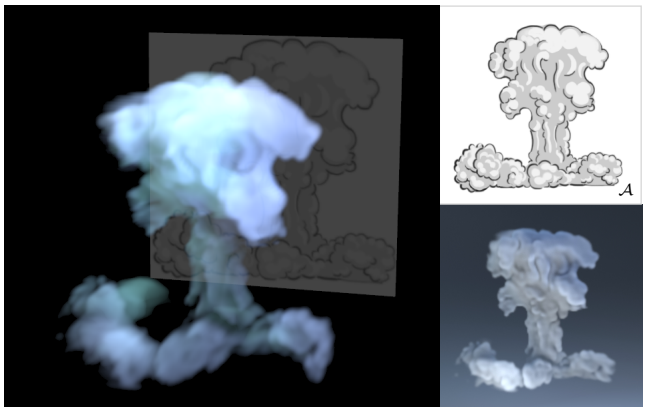


Figure 2: *Screen capture of an interactive authoring session of a volcanic plume, and the corresponding sketch and rendered result.*

Learning-based sketch modeling. For geometric shape modeling, Convolutional Neural Networks (CNNs) were used to build a direct mapping from sketches to procedural 3D shape models [HKYM16, NGDA*16, LGK*17], mostly restricted to pre-defined object classes and fixed viewpoints. Delanoy et al. [DAI*18] trained a CNN to predict occupancy in a voxel grid based on a single or multiple contour drawings as input. The method first generates an initial reconstruction with a single-view network, and then uses a so-called updater network to iteratively refine the prediction as new drawings from additional views are provided. We follow their idea of using subsequent refinement steps, but propose significant changes to the network components, loss functions and training process for increased robustness and multi-view consistency.

Li et al. [LPL*18] use a CNN to infer the depth and normal maps representing the surface of an object. To reduce ambiguity, the network additionally considers the flow field of the surface to generate a confidence map. The input to the network consists of sketches, silhouette mask, and optional depth sample points and curvature hints. Depth and normal maps were also used to reconstruct a dense point cloud from sketches [LGK*17]. The decoder captures the object's surface from several viewpoints, which are then fused into a single 3D point cloud through optimization. To eliminate the need of well-labeled hand-drawn sketch data during the training process and hence allow sketch style variations, Wang et al. [WQWF18] introduced an unsupervised learning model. A shape representation was also learned in a fully self-supervised system in [SBS21] to prevent non-manifold artifacts in the parametric shapes.

A deep learning based sketching system has also been used for 3D face and caricature modeling [HGY17]. 2D lines representing the contours of facial features represent the input to a CNN. An initial sketching mode is followed by sketch and gesture based refinement steps. Deep learning was also used for an interactive modeling tool of 3D hair from 2D sketches [SZF*20]. Hair contour and a few strokes indicating the hair growing direction are used by a first network to generate a 2D hair orientation field, which is then processed by a second network to output a 3D vector field. For sketch-based garment design, a joint latent shape space is learnt

across different modalities representing the draped garment, body shape parameters and garment parameters [WCPM18].

Noteworthy is also the non-photorealistic rendering system *SynDraw* that facilitates the generation of synthetic drawings to train sketch-based modeling systems [WB19], and the work of Zhong et al. [ZGZS20] that identified the main differences between sketch- and image-based modeling with respect to style variance, imprecise perspective and sparsity.

The above discussed CNN-based 3D shape reconstruction methods fail *by design* to reconstruct realistic flow details, as they target different classes of objects compared to our work. In contrast, our reconstruction is driven by fluid simulation data that captures turbulent structures realistically and maps them to the sketched cues.

Modeling density volumes. To our knowledge, no method exists that generates 3D reconstructions of density volumes with realistic flow structures from 2D artist sketches. Classical, non-learning based approaches typically require a mesh or surface as input and fill the volume with density values or particles. Turbulent structures can be mimicked by adding noise to the boundary particles [SBRS10]. The lack of physical plausibility is manifested in the results, which we address by driving our reconstruction with simulated flow data. Clouds have also been authored by a skeleton approach to model the surface shape and by adding details during rendering [WBC08]. This is opposed to our method, where such details are included in the 3D representation. While it would be possible to couple the above mentioned methods with a fluid solver to achieve physically realistic structures, this would inherently prevent interactive frame rates during authoring and restrict the tool to artists with expert knowledge in animation and effects.

Less related to our work but noteworthy are reconstruction methods developed in fluid animation. 3D fluid densities and motion have been reconstructed from image sequences [EHT18, EUT19, FST21, QLWQ21] and 3D smoke has been stylized based on 2D example images [KAGS19, KAGS20]. Sketches have only been used in form of strokes on selected keyframes to control the motion of liquids [PHT*13]. Local editing is computed with an efficient optimization and propagated spatially and temporally. Sketched strokes were also used in interactive environments to define shapes and connections of fluid circuits primarily applied in medicine [ZIH*11] and for 2D flow field design using a generative adversarial network [HXF*19].

3. Overview

The design principles of effects drawing, and in particular of smoke, provide us a solid foundation for the design of an algorithm that progressively refines a 3D density volume at arbitrary viewpoints based on artist sketches.

3.1. Fluid Sketch Principles

Sketching fluid objects, such as smoke, requires a specific style and workflow. The first step towards a tool designed to interpret such drawings is to understand their specificity. We analyzed smoke drawings from multiple sources, such as the book of

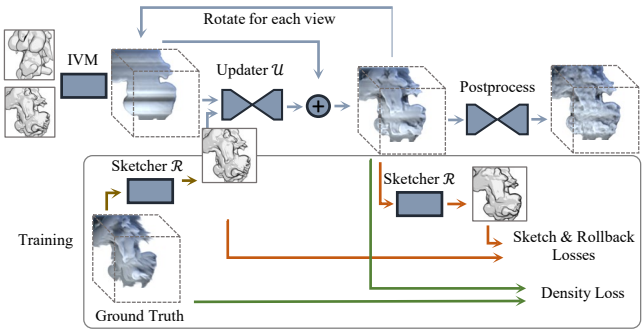


Figure 3: *Overview of our method. At inference time, the updater network $\mathcal{U}$ takes as input a sketch and a density volume (initialized by IVM) and computes a* residual *to correct the volume. By rotating it to arbitrary viewpoints the result can be iteratively refined. A subsequent post-processing GAN synthesizes additional fine structures. At training time, our sketcher generates drawings of our smoke data set. The sketcher is also a key component of the unsupervised sketch and roll-back losses.*

Gilland [Gil12], and summarize the findings that have driven the development of our tool and that explain the differences to drawings of solid objects.

Strokes. Common to many sketch-based modeling techniques, the input to our method is a user-drawn sketch. Even with modern digital tools, the workflow is still close to pen and paper drawing. Our style is thus made of several strokes with varying shades of grey and thickness (both properties respectively linked to the pressure and inclination of the pencil or piece of chalk). The main dark strokes express the silhouette and the contours of additional inner details.

Generality. Key to the plausibility of natural systems is the chaotic nature, which forbids any kind of problem simplification with structures, symmetries, hard edges or parallel strokes. Therefore, our method does not have to carefully reproduce such geometric features, but cannot either rely on them to better reconstruct the object, as commonly done by previous work [XCS*14].

Simplicity. Intrinsic to effects drawings are natural design principles. Instead of filling the drawing with random details, every detail represents a pattern of energy. Fluid energy propagates through different scales, the so-called Kolmogorov cascade, which results in turbulence patterns that can visually be clearly distinguished. It is often advised to keep the drawing simple and to focus on a few frequencies rather than drawing all intricate details - not least to reduce the sketching time that would otherwise grow exponentially. Our sketch style follows this principle, which at the same time is also beneficial for machine learning.

Volume. Inner contours are too sparse to fully express the many variations of the curvature of smoke objects. Additional shading is used by artists to encode this information, which also gives a better intuition of the relative placement of smoke structures. To comply with the simplicity of the strokes guideline, we impose a two-color

toon shading with a light source placed such that on average 2/3 of the smoke is in the light. We see this requirement for shading as a trade-off between a too simplistic sketch that would fail to capture the intricate details of the smoke and a more complex representation that might prevent novice users from using our tool. We show in Section 8.1 that this additional color information, even painted carelessly and unrealistically, provides a powerful control hint to solve curvature and depth ambiguities in the reconstructed density.

3.2. Pipeline

One major challenge of sketch-based modeling lies in the depth ambiguity and occlusions that cannot be captured from a single viewpoint. We follow [DAI*18] and allow a user to progressively refine sketches at different angles and hence iteratively update and relocate smoke structures. Our pipeline is depicted in Figure 3. The core of our method is an updater network (Section 4.2), which takes as input a 3D voxel grid containing smoke densities and a sketch of the desired output from the target view. The output is a corrected volume that respects two key properties:

Accuracy The output density volume seen from front view should correspond to the input sketch.

Stability The density parts that are not visible from the front view should be preserved to ensure multi-view continuity.

The initial density can be user-provided, but we also propose a simple method to create an initial guess from two sketches in canonical views (Initial Volume Modeling, or IVM, Section 4.1). We eliminate the need for artists to sketch high-frequency flow structures by using a variation of multi-pass GAN [WXCT19] that synthesizes details in a post-processing step (Section 4.4).

We use a semi-supervised training strategy (Figure 3, bottom), designed to enforce the two properties of the updater detailed above. The key ingredient is a new synthetic sketcher for smoke volumes (Section 5), aiming at producing sketches that are compliant with the guidelines of Section 3.1. It has two main functions: First, to automatically annotate a data set of smoke models (Section 6). Second, the differentiability property of the sketcher allows us to incorporate it in the loss functions. This enforces the *accuracy* property, which we reformulate as the minimization of the difference between the input sketch and the synthetic sketch of the output. Generalizing this to all previous viewpoints leads to the rollback loss (Section 4.3), which enforces the *stability* property of the updater.

4. 2D Sketch to 3D Density Prediction

Sketches are the primary control tool for the artist during the authoring process. They represent the input to our convolutional neural network (CNN) that computes the corresponding volumetric density fields. To allow for multi-view sketching, we use a refinement strategy: the volume is rotated to face the sketch viewpoint and is then corrected by the network.

4.1. Initial Volume Modeling

The initial volume modeling (IVM) computes an initial guess for the density field $\hat{d}_0$ from the provided input sketches s, i.e.

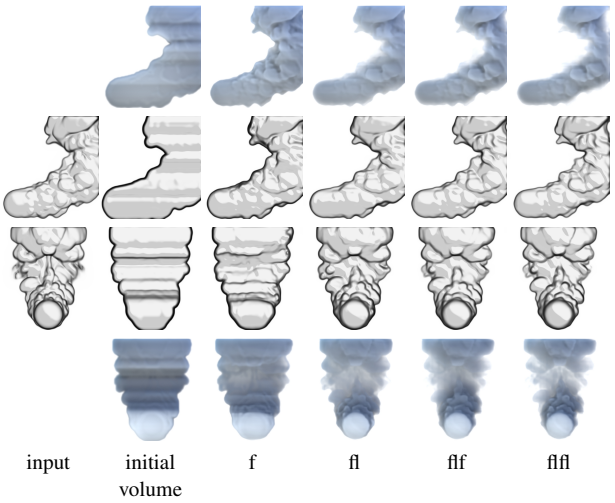


Figure 4: *From two input sketches depicting the front and left views, we compute the initial volume and then alternately optimize the front (f) and left (l) view density reconstructions with an updater CNN. The intermediate density fields and corresponding sketches illustrate the convergence.*

$\hat{d}_0 = \mathcal{V}(s_1, .., s_n)$. For each sketch, we first extract the contours and foreground with thresholding and filling operations (note that our sketching style imposes a different color for the background). From the inputs of the different viewpoints, we compute a visual hull and an initial guess of the density distribution. We calculate the intersection of these volumes from the different views, blend them and apply Gaussian blur to smooth the boundaries. The output represents the initial guess for the subsequent optimization steps, i.e., the input to the neural network. The initial volumes are depicted in Figure 3 and Figure 4. Note that IVM requires at least two viewpoints (possibly duplicates of the front view). In all our illustrations and most examples we use two input sketches indicating the front and left views, i.e., $\hat{d}_0 = \mathcal{V}(s_f, s_l)$. In practice, arbitrary additional views can be added if needed.

4.2. Updater CNN

We adopt an updater CNN $\mathcal{U}$ that alternately optimizes information from the different input views. We refer to each of these iterations as pass p, associated with a camera orientation θ_p.

Each pass p takes as input an artist sketch s_p and a previously reconstructed density field $\hat{d}_{p-1}^{\theta_p} = \mathcal{T}(\hat{d}_{p-1}^{\theta_{p-1}})$. Note that we need to rotate (i.e., $\mathcal{T}(.)$) the output of pass $p-1$ computed at an angle θ_{p-1} to obtain the input of the next pass p at an angle θ_p. The reconstructed density after p passes, seen from an angle θ_p, is then given as $\hat{d}_p^{\theta_p} = \mathcal{U}(\hat{d}_{p-1}^{\theta_p}, s_p)$. In the first iteration, the predicted density field corresponds to the initial density $\hat{d}_0^{\theta_0} = \hat{d}_0$ computed with IVM and no rotation is performed $\hat{d}_0^{\theta_1} = \hat{d}_0^{\theta_0}$. During training, we pick a random view from one of the 24 canonical views in each pass, as multi-pass/view training results in higher robustness for the reconstruction when the density field is optimized from multi-

ple viewpoints. An overview of the training pipeline is shown in Figure 3.

Figure 4 shows the output of the individual passes at test time for an example where two sketches are provided. The updater CNN alternately uses the front and left view sketch in the optimization. In this example, a total of 4 passes are computed (front - left - front - left). The intermediate density reconstructions and corresponding sketch representations illustrate the convergence after multiple passes.

The concept of using an updater network is similar to Delanoy et al. [DAI*18], but there are significant differences in the network components, loss functions, and training process that impact reconstruction quality and robustness of predictions. We train the network to predict the residual ($d^{\theta_p} - \hat{d}_p^{\theta_{p-1}}$) [HZRS16] for correction instead of predicting the density field directly. Due to the ambiguity of line drawings, [DAI*18] focuses on a few informative viewpoints for training. Although we also use canonical viewpoints, our differentiable sketcher allows not only less ambiguity on depth with its shading, but also easy extension to arbitrary viewpoints on the fly, without any loss of generality.

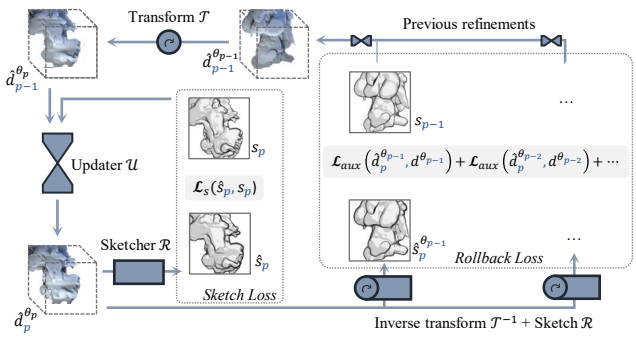


Figure 5: *In addition to the density loss, our training process uses the sketch loss (left), which compares the input sketch s_p to a synthetic sketch created from the output density $\hat{s}_p$. The rollback loss (right) further rotates the output density back to face the previous viewpoints to match the corresponding target sketches. In practice, this enforces that the unseen details are not changed by the updater.*

4.3. Loss Functions

In each pass, we calculate view-dependent losses (density loss, sketch loss, depth variation loss) and a loss over all preceding passes (rollback loss) for preserving reconstructions of previous input views.

Density Loss. We define our density loss $\mathcal{L}_d$ in terms of $L1$ loss on the reconstructed field, similar to [KAT*19]:

$$\mathcal{L}_d(\hat{d}, d) = ||\hat{d} - d||_1, \tag{1}$$

where d refers to the ground truth density field and $\hat{d}$ is the reconstructed field from our updater network.

Sketch Loss. An important difference to previous work is the integration of a differentiable sketcher $\mathcal{R}$ (Section 5) into the network, which allows us to generate a target sketch $s_p = \mathcal{R}(d_p)$ and a sketch of the reconstructed density $\hat{s}_p = \mathcal{R}(\hat{d}_p)$ at each pass p during training. We use this to minimize the distance to the input sketch with a sketch loss $\mathcal{L}_s$ (Figure 5, left) given as

$$\mathcal{L}_s(\hat{s}, s) = ||\hat{s} - s||_1, \tag{2}$$

where s is the input sketch and $\hat{s}$ is the rendered sketch from the reconstructed density $\hat{d}$.

Depth Variation Loss. As the sketch loss only focuses on the rendering of the 3D volume, there is no constraint in the depth direction during training. Although the sketch shading encodes some depth information, artifacts in depth are still noticeable even with the density loss (Figure 20). We, therefore, use the total variation loss in the depth direction as a regularizer to enforce smoothness:

$$\mathcal{L}_{tv}(\hat{d}) = ||\frac{\partial \hat{d}}{\partial z}||_1. \tag{3}$$

Full Objective. We define an auxiliary loss $\mathcal{L}_{aux}$ combining sketch loss and total variation loss in depth as following:

$$\mathcal{L}_{aux}(\hat{d}, d) = w_s \mathcal{L}_s(\mathcal{R}(\hat{d}), \mathcal{R}(d)) + w_{tv} \mathcal{L}_{tv}(\hat{d}), \tag{4}$$

where w_s and w_{tv} are weights for the sketch loss and total variation loss set to 0.03 and 0.1, respectively. We can then define a pass loss $\mathcal{L}_p$ as

$$\mathcal{L}_p = p\mathcal{L}_d\left(\hat{d}_p^{\theta_p}, d^{\theta_p}\right) + \mathcal{L}_{aux}\left(\hat{d}_p^{\theta_p}, d^{\theta_p}\right) + w_{rb} \sum_{n=1}^{p-1} \mathcal{L}_{aux}\left(\hat{d}_p^{\theta_n}, d^{\theta_n}\right). \tag{5}$$

We refer to the last term as the rollback loss (Figure 5, right) weighted by w_{rb}. The rollback loss compares the synthetic sketches of the generated density and the one of the ground truth from all previously reconstructed viewpoints. This enforces the updater to preserve the density in the areas unseen by the current view as much as possible, which would otherwise be degraded. The rollback weight defines how much the updater network preserves the reconstructions of previous viewpoints. While at the beginning of the optimization a large degree of freedom is necessary and hence a lower value for w_{rb} is advantageous, a larger value is important in later steps to preserve structures reconstructed from other viewpoints. Therefore, we progressively increase w_{rb} from 1 to 100 during the first epoch.

The total loss is then defined as

$$\mathcal{L}_{total} = \frac{\sum_p \mathcal{L}_p}{\sum_p p}. \tag{6}$$

4.4. Post-processing Network

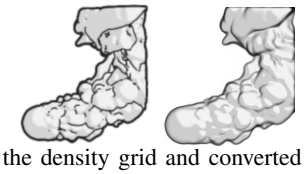
To alleviate the need for an artist to sketch small-scale structures, we use a second network to synthesize flow details onto the reconstructed density. We use the architecture of tempoGAN [XFCT18] but restrict the model to the spatial discriminator. For the training, we first add the ground truth velocity to the input data, which during inference is the output of the density prediction $\hat{d}_p$. Similar to multi-pass GAN [WXCT19] we train the network in two passes. In

Figure 6: *Details can be synthesized with multi-pass GAN onto a density field (left) and controlled by the curl-noise parameters (middle, right).*

each pass we decompose the 3D data into a stack of 2D slices along an axis, using the z-axis in the first pass and the x-axis in the second. The passes are trained separately with the same architecture. At test time, since we have no velocity information in the sketches nor the reconstructions, we synthesize turbulent flow fields with curl-noise [BHN07] onto the reconstructed density fields. The different parameters - octaves for frequency, seed, and velocity scale - allow control over the visual result as depicted in Figure 6.

5. Differentiable sketcher

Our sketcher aims to generate a stylized representation of a smoke keyframe that respects the guidelines observed in Section 3.1.



Previous techniques typically extract contours from meshes [DFRS03] (inset image, left) and cannot be directly applied to amorphous shapes such as smoke densities. Although isosurfaces can be extracted from the density grid and converted into a mesh, the choice of the isovalue is not obvious and typically low-density details are lost in the process. Therefore, for cartoon rendering of smoke, particles have been used and traced through the underlying fluid grid, and then rendered as textured billboards without [SMC04] and with [MF06] shading and shadow effects.

In contrast to using tracer particles, we work directly with volumetric data and adapt the mesh contour extraction to amorphous smoke shapes (inset image, right). Our method is inspired by the absorption in smoke rendering and estimates a normal map. We average normals for a range of density isosurfaces at the visible 'boundary' of the smoke (Equation (8)). At each position $\mathbf{p}$ in the volume, the normal of the isosurface is the gradient of the density field: $\nabla d(\mathbf{p})$. We integrate the normals along the viewing rays while weighting them such that the normals closer to the boundary have higher importance. For a ray r, the normal $\mathbf{n}_r$ is given as

$$\mathbf{n}_r = \int_0^\infty \nabla d(r(s))\, w(r(s))\, \mathrm{d}s, \tag{7}$$

where s is the distance to the camera and w the weighting factor.

A possible choice for the weights is to use the transmittance of the smoke. Computing the averaged normal would then be equivalent to the rendering of the smoke, where the density gradient is set as the light emission of each of the voxels. We found, however, that this strategy results in too smooth variations in normals, leading to blurred sketches. Therefore, we choose a stricter filtering strategy:

Let $\tau(s)$ be the accumulated density along the ray: $\tau(s) = \int_0^s d(t)\,\mathrm{d}t$. We choose our weight as

$$w(s) = \xi^2 \cdot \tau(s)\, e^{-\xi\cdot\tau(s)}, \tag{8}$$

where $\xi = 5$ is a scale coefficient. This ensures that a small band of smoke close to the boundary contributes to the final normal while discarding the low-density voxels that are first encountered by the ray and which produce noisy normals.

We make two assumptions to discretize Equation 7. First, we assume that the gradient of the density is constant between two voxels , and second, that τ varies linearly in a small neighborhood. The normal can then be defined as

$$\begin{aligned}\mathbf{n}_r &= \sum_{i=0}^{i=n-1} \frac{\nabla d(i) + \nabla d(i+1)}{2} \int_i^{i+1} w(\tau(s))\,\mathrm{d}s + \mathbf{n}_\infty \\ &= \sum_{i=0}^{i=n-1} \frac{\nabla d(i) + \nabla d(i+1)}{2} \frac{\left[-(\xi\cdot\tau+1)\cdot e^{-\xi\cdot\tau}\right]_{\tau(i)}^{\tau(i+1)}}{\tau(i+1)-\tau(i)} + \mathbf{n}_\infty,\end{aligned} \tag{9}$$

where n is the number of cells along the ray (here it is assumed to have a cell size of 1, without loss of generality). Similarly, n_∞ is the background normal, that we set as opposed to the view direction $\mathbf{v}$:

$$\mathbf{n}_\infty = -\mathbf{v}\int_n^\infty w(\tau(s))\,\mathrm{d}s = -\mathbf{v}\,(\xi\cdot\tau(n)+1)\cdot e^{-\xi\cdot\tau(n)}. \tag{10}$$

The integral appearing in Equation 10 is also used as a mask that indicates if the ray intersected the smoke.

We normalize $\mathbf{n}_r$, which is then used to extract the contours c as:

$$c = max(0, min(-\mathbf{n}_r\cdot\mathbf{v}, \delta))/\delta, \tag{11}$$

where δ is a threshold set to 0.8. We then add a two-colors toon shade t to provide more information about the volumes and we combine all to compute the sketch s as

$$s = (1-c)\cdot c + c\cdot t. \tag{12}$$

The individual steps of the sketcher $\mathcal{R}$ are visualized in Figure 7.

Figure 7: *The differentiable sketcher computes normals, contours, and toon shading (from left to right), and combines them into the final sketch (right).*

6. Training Data Generation

The generality of our data set (*data set and code will be released after acceptance*) is key for robust and high-quality reconstruction of diverse scenes sketched by artists, that can include both physically inspired and non-physical shapes and motions of smoke. Therefore, we build a new data set consisting of smoke densities, which captures a large variety of volumetric shapes. We also augment the output of our synthetic sketcher with several strategies to increase

the robustness of our network with respect to small variations in sketching styles.

6.1. Simulation Training Data

Our simulation data set consists of 39,380 density fields (from 1641 simulations) used for training and 600 density fields (25 simulations) for validation. Example snapshots from the training set are shown in Figure 9 and in the Supplementary Information (*SI*). Each simulation contains 20 frames without source and 30 frames with source computed at a resolution of 129^3. We run the simulation in *Houdini* with randomly changing temperature diffusion factor, cooling rate, viscosity, buoyancy strength, and direction, as well as sharpening and turbulence factors.

For simulations without a source, the pre-generated source is set as the initial density of the simulation. For the ones with source, we selected one of the regenerated sources and combined it with a max operator on the ongoing simulation. The number of sources for each simulation are [1,2,3,4] with probability [10,5,2,1]/18. The shapes of the initial sources are generated by computing a random shape from a set of union or difference of cube, cylinder, and sphere, and setting the resulting shape as the initial density of a fluid simulation that we run for 10 frames. We added this supplementary operation to remove sharp edges and corners found in the initial shapes and to capture a more diverse range of densities than the original binary ones.

Based on the density histogram, we find that both simulation strategies are complementary to each other. No source data are dominant by small density values, while with source data higher density values can be observed. In this way, our neural network model is not biased to learn to output small or large values only. It also helps to increase variation in the data set so that we can handle various unseen data, such as simulation with and without sourcing.

6.2. Sketch Training Data and Augmentation

The synthetic sketcher allows us to inexpensively build a large data set of smoke simulations coupled with sketches. We used sketch sizes of 258^2. Using a larger size compared to the density fields improves quality and is also more artist-friendly. To increase the robustness against variations in artist style, we augmented the sketches used in the training to cover different values for brightness, contrast, contour strength, toon shading color, blurring, and slurring [SSISI16] in x and y directions as noise addition, and variations in the light direction in x and y (see *SI*). Figure 8 illustrates the different properties.

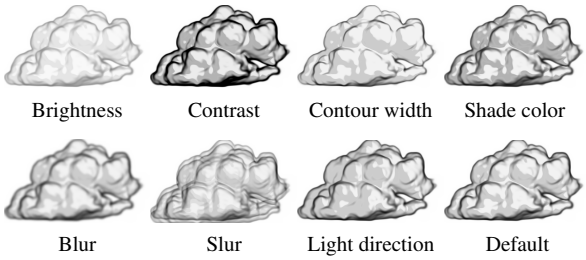


Figure 8: *Sketches are augmented to consider variations in sketching styles to increase robustness of the reconstruction.*

7. Implementation and Synthetic Results

We first provide additional details on the implementation and then evaluate our reconstructions on the training and validation data set and on synthetic scenes.

7.1. Implementation and Performance

We implemented our system in PyTorch and used the Adam optimizer [KB15] with a learning rate of 0.0001. Our network follows a U-Net architecture. The details of the architecture can be found in the Supplementary Information (*SI*). In the decoder, we use nearest neighbor upsampling with flat 2D convolution instead of transposed convolution to avoid checkerboard artifacts [ODO16]. The network is trained on density fields in the range $[-1,1]$, using the full objective. In all our examples we used 3 passes in the training and two subsequent refinement steps (front-left (fl)) at test time. Different settings are evaluated in the ablation study in Section 9.3.

Our model converges in 10 epochs (see *SI*), and the training time with our data set takes up to 3 days using 4 NVIDIA GTX 1080 GPUs. At test time, the IVM computation with front and side input sketches takes $\sim$ 6.5ms per frame on a single GPU. Similarly, the update step of our trained updater CNN takes $\sim$ 3ms per frame. With refinements on front and left views (fl), the total inference time is $\sim$ 12.5ms per frame and hence our method is able to reconstruct 3D densities in real-time.

The post-processing model is trained with 20 epochs and takes up to 8 hours to train the first pass and another 8 hours for the second pass using an NVIDIA RTX 3080 GPU. At test time the network takes $\sim$ 2s per frame for the first pass and $\sim$ 0.5s for the second pass.

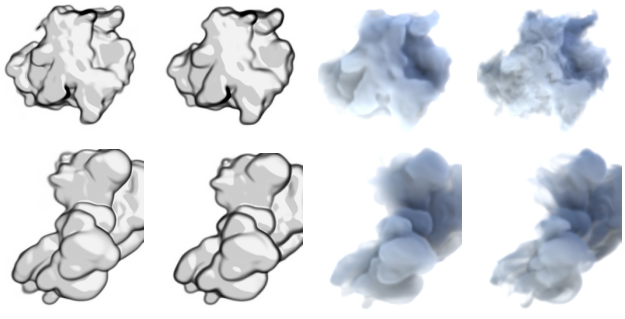


Figure 9: *Density reconstructions of seen (top) and unseen (bottom) examples of our data set. From left to right: input sketch, sketch of the reconstructed density, reconstructed density, ground truth density.*

7.2. Reconstruction of Training and Validation Data

Figure 9 shows the front view result of a reconstruction of the training data set ('seen', top row) and validation set ('unseen', bottom

row). From left to right we show the sketch of the ground truth (input sketch), the sketch of the reconstructed density, the reconstructed density (without post-processing), and the ground truth density. It can be seen that only minimal differences are visible between the input sketch and the sketch of the output. The reconstructed density is - as expected - smoother than the ground truth. This is because the sketch lacks complete information about the original density. This implies the ambiguity of several possible densities corresponding to a given sketch. While an alternative could be to use a sketch style that also encodes fine flow structures, we preferred to follow the sketching principles outlined in Section 3.1 and, if desired, to synthesize high-frequency details in the post-processing.

7.3. Reconstruction of Synthetic Scenes

We show that our method generalizes well to various scenes by applying it to inputs from physically-based simulations and selected frames from animation sequences including a character, cloud, and puppy animation. For these examples, we generated synthetic sketch inputs by applying our sketcher to the given density field. It is important to note that the given density cannot be directly compared to the reconstructed density: our method infers a density such that its sketch is close to the input sketch.

7.3.1. Simulated Smoke

We used a selected frame of the smoke jet data set of [KAGS19] and applied our sketcher to the existing density field to generate sketch input keyframes for front and left views. Figure 10 shows the synthetically generated sketch from the given density (input to our method), the sketch of the reconstructed density field, and the reconstructed density without and with post-processing. The reconstructed density closely matches the footprint of the input sketch. We measure the quality of the inner details by comparing the synthetic sketches of the input with the one of our output, where we see that the shading and inner contours are well captured.

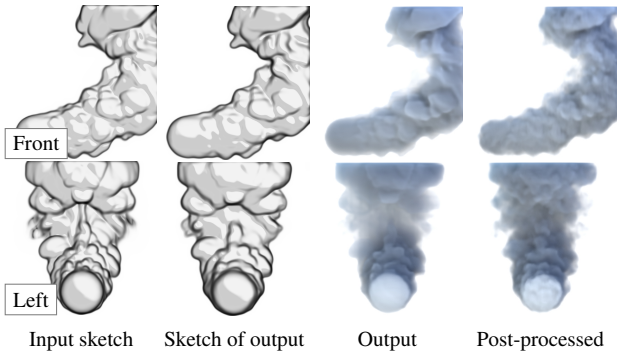


Figure 10: *Result using a physics-based simulation data set (smoke jet) of [KAGS19] showing front and left views. From left to right: input sketch, sketch of the reconstructed density, reconstructed density, post-processed result.*

7.3.2. Animation Data

To evaluate the model on unseen (non-smoke) shapes, we used selected keyframes of animation sequences as input and show the corresponding reconstructions in Figure 11, without and with post-processing. The top row shows the results using the dancing character animation data set from Adobe's Mixamo Gallery [MG]. This data is not only difficult because no comparable examples were part of the training data set, but also because of thin extremities and occlusions. The remaining rows show results using the cloud data [CL] and a keyframe of a running puppy animation [Nit]. For the character and puppy examples, we first converted the original meshes into volumes before computing the input sketches. Only minimal differences are visible when comparing the input sketch to the reconstruction sketch, demonstrating that our method is not only able to reliably reconstruct unseen shapes, but especially also handle non-physical (non-smoke) inputs.

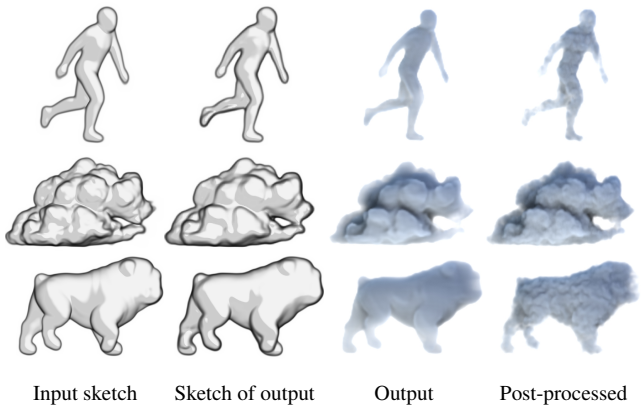


Figure 11: *Results using keyframes of three unseen animation sequences: dancing character, clouds, and puppy. From left to right: input sketch, sketch of reconstructed density, reconstructed density, post-processed result.*

7.3.3. Arbitrary Viewpoints

In most of our examples, we show results where the sketches are prescribed from canonical viewpoints (front and left views). We believe that this is one of the most complicated cases for density reconstruction because no information is shared between the different views. But we also show that our method can be applied efficiently to sketches from arbitrary viewpoints. The main - classical - issue is that the corners of the domain are cropped during rotation. We chose not to increase the size of the volume, but instead we reduce the domain to a sphere inscribed inside the 129^3 cube. A quick computation shows that proportionally less volume is wasted with this strategy. Figure 12 shows an example where the *puppy* model was reconstructed from sketches drawn from arbitrary viewpoints. The last row demonstrates again the effect of the neural post-processing to synthesize turbulence details.

8. Application Demos

We demonstrate our approach on three real-world applications. We first present 3D smoke results generated by artists, introduce our in-

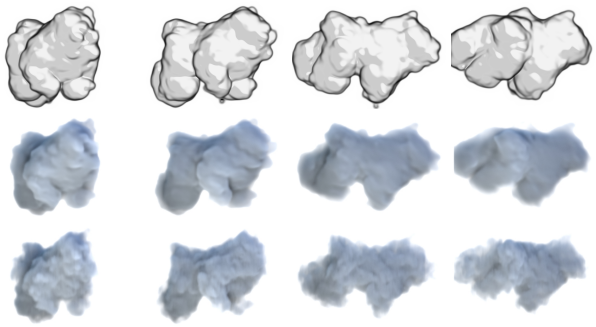


Figure 12: *Our method can seamlessly handle progressive generation from arbitrary viewpoints despite training with canonical views. From top to bottom: input sketches, reconstructed densities, and post-processed densities.*

teractive authoring tool for smoke keyframe prototyping and show how the 3D models could be used in animation.

8.1. Results on Artist Sketches

We used sketches drawn by multiple different artists to evaluate our method at test time. We instructed the artists to sketch keyframes of smoke or an imaginary smoke-like object. For each example, we requested a front and side view sketch. We provided specific instructions for image size (256x256), light direction (front-right, slightly up), shading (toon, white, and grey), colors (grey-black lines, white background), and line width and style (up to 5px with Gaussian falloff).

Figure 14 depicts different keyframes drawn by artists and the corresponding reconstructions, without (left) and with (right) post-processing. The scenes illustrate a dissolving character that transforms into a rising smoke cloud, a shape of a lion that transforms into a rhino, and a smoke plume that transforms into a bird-like object. A sequence depicting the dissolving character can be seen in Figure 13. Further sequences, as well as sketches and renderings from other viewpoints, are provided in the *SI*.

Our method works surprisingly well for such highly non-physical examples. The reconstructed density and sketches thereof closely match the input sketches from the artists. Note that the reconstruction is also robust to variations from the prescribed sketch style and can handle inconsistencies of shape and shading between the different views.

We observed that artists took advantage of our sketch style to draw in a coarse-to-fine manner (Figure 15). First, they specify the silhouette, giving a global sense of the resulting volume. They subsequently refine with sketches detailing the inner contours, which result in interior volumes where the direction of curvature or relative depth is not the one desired by the artists. Fortunately, this can be effectively corrected by a last stroke with shade information. Interestingly, our system proved robust to unrealistic shading, and even non-artists intuitively used this tool as an extra hint for the network to solve ambiguities in the reconstruction.

8.2. Real-time Authoring in Houdini

We have implemented a Houdini plugin for interactive smoke field authoring. For two selected authoring sessions with different artists, we show a screen capture, the generated sketch, and final rendering in Figure 16 and Figure 2. It took the untrained artists only a couple of minutes to create, iteratively edit the shape, and render the final result. Noteworthy is that our sketcher provides the artist with an estimate of sketches as guidance from unseen viewpoints. The accompanying video shows additional examples by novice users. It can be seen that the sketched front and side view inputs do not perfectly correspond and that some artists even varied the sketch style or introduced imperfections such as gaps between strokes. Even in such challenging cases, our method could reliably infer the 3D shape. In Figure 17 we used an image from [Gil12] as input to our tool. We duplicated the front view, edited the resulting side view, computed the 3D reconstruction, and rendered the result with a liquid texture.

8.3. Keyframe Control of Smoke Simulations

A further application of our method is the authoring of density volumes that can then be used as target keyframes in fluid simulation control methods. We use the most recent approach for keyframe control [TACS21] that employs a differentiable fluid solver to compute derivatives of the objective indicating target density matching quality with respect to control parameters, to generate animations that match the targets. We take the reconstructed densities as target keyframes from the dancing character (Figure 11), top row) and dissolving character (Figure 14, top row) scenes, and use the earliest frame as the initial state and the later two frames as targets. Selected frames from the keyframe control animation can be seen in Figure 18. The resulting animation sequences show that the simulation matches the target density keyframes (inset images) while generating intrinsic flow structures and filaments due to the dynamics computed with the underlying physics solver (see also the accompanying video).

9. Evaluation

We compare our method to previous work and evaluate the results in a user study with viewers. We additionally discuss the results of a user study with artists to answer questions related to our authoring tool and sketch style. Lastly, we evaluate the design choices of our network and method in an ablation study.

9.1. Comparison with Previous Work

We compare our approach (without post-processing) to a previous CNN-based shape reconstruction method [DAI*18] that predicts occupancy in a voxel grid based on a single or multiple contour drawings as input. Similar to our method, an updater network is used to iteratively refine the result. Since the previous work targets solid geometry reconstruction, we retrained the model on contour drawings of our data set with the density loss (Equation (1)) to make the results comparable. Figure 19 shows the superiority of our method in terms of reconstruction quality of outer and inner structures, which is attributed to the network design and differentiable sketcher.

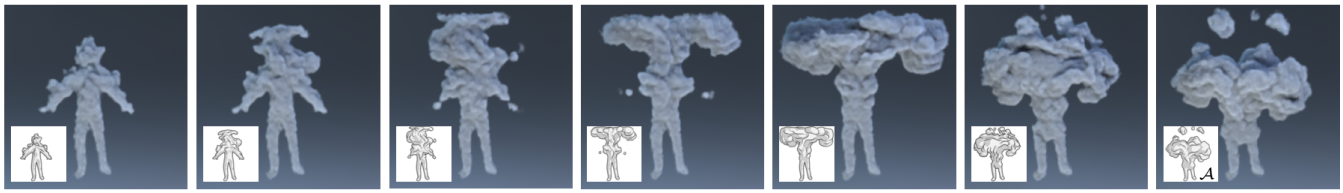


Figure 13: *A sequence depicting a dissolving character sketched by an artist and reconstructed with our method. Our efficient sketch-to-density computation is especially useful for rapid prototyping and communication of ideas.*

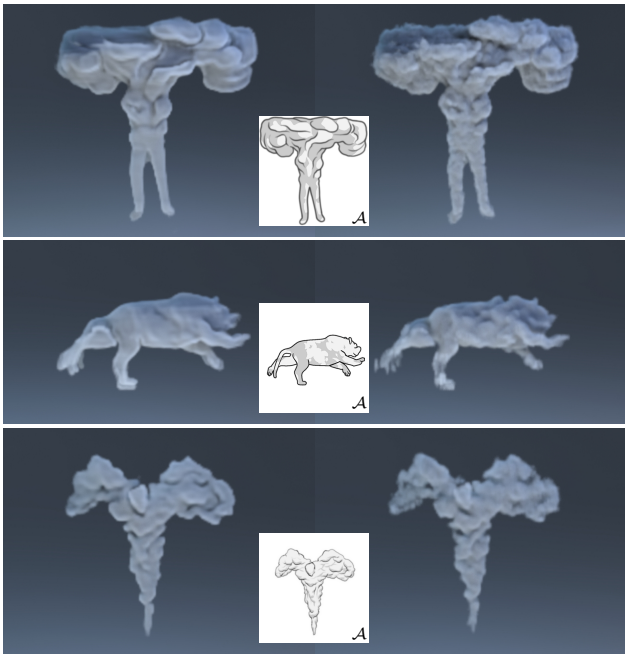


Figure 14: *Reconstructed results without (left) and with (right) post-processing using sketches from three different artists. From top to bottom: dissolving character, lion to rhino, and bird.*

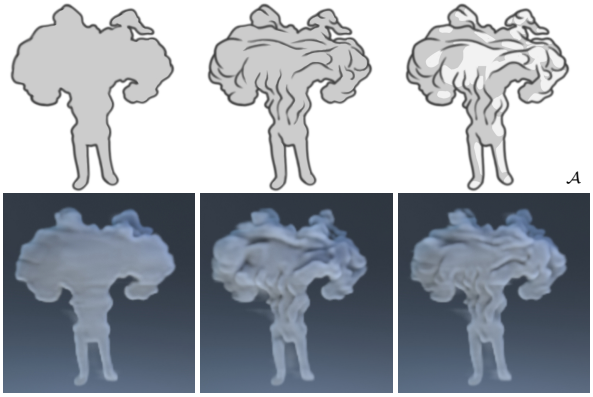


Figure 15: *Our sketch style naturally guides the users to a global-local approach, where the silhouette (left) defines the boundaries of the objects, inner contours (middle) specifies some details, and shading provides an extra level of control on the direction of curvature.*

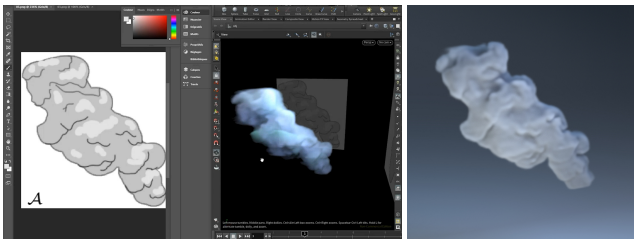


Figure 16: *Interactive authoring session with an artist. The front and side views of a smoke volume are sketched and edited in our Houdini plugin with a real-time preview of the density field (left) created in only a few minutes. The final rendering (without post-processing) is shown on the right.*

The same figure shows the results of a classical, non-learning-based approach for density authoring. The cloud modeling method [SBRS10] first computes a surface mesh, which is then filled with particles. By adding noise to the particles, finer structures are generated. The results indicate that our approach can reconstruct flow details more realistically and accordingly suggest the advantage of driving the modeling by simulated flow structures rather than noise.

9.2. User Study

9.2.1. Evaluation with Viewers

We conducted a user study with 53 viewers to evaluate how they perceive our results in comparison to previous work (A) [DAI*18] and (B) [SBRS10] (Section 9.1). For 8 different scenes, we showed the rendered images of the three methods side-by-side in random order. Table 1 summarizes the statistics for the three scenes shown in Figure 19. We asked the users to rate the image with the best overall similarity to the input sketch, outer contours, inner contours, shading (normal information), and similarity to real-world smoke. We choose that specific order for the questions to minimize the bias induced in the subsequent analyses. We observed another bias in the appearance of the results, especially of method (B), for which we rescaled the densities so that they resemble ours. The remaining difference in appearance comes from the structure of the details and the thicker boundaries inherent to method (B). In all questions our method clearly outperforms the previous work, except for one

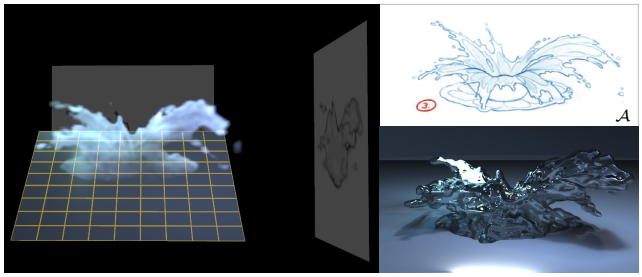


Figure 17: *Sketched splash of [Gil12] reconstructed with our method and rendered with a liquid texture.*

Figure 18: *Selected frames of five different smoke simulations where our reconstructed densities are used as keyframes (inset images) to artistically control the motion of the smoke [TACS21]. Due to the underlying fluid solver, intrinsic flow details appear.*

where the users found that method (A) better resembles real smoke (Figure 19, top row).

User study (Ours [%], A [%], B [%])			
	Smoke data	Cloud	Character
Similarity	(**86.8**, 13.2, 0)	(**75.4**, 20.8, 3.8)	(**84.9**, 11.3, 3.8)
Outer	(**98.1**, 1.9, 0)	(**85**, 7.5, 7.5)	(**86.8**, 3.8, 9.4)
Inner	(**100**, 0, 0)	(**94.3**, 5.7, 0)	(**90.6**, 7.5, 1.9)
Shading	(**90.6**, 7.5, 1.9)	(**86.8**, 7.5, 5.7)	(**69.8**, 9.4, 20.8)
Reality	(34.6, **40.4**, 25)	(**54.8**, 35.8, 9.4)	(**52.8**, 32.1, 15.1)

Table 1: *User study statistics for the three scenes of Figure 19. We asked 53 participants to rate the best reconstruction (Ours, A [DAI*18], B [SBRS10]) with respect to overall similarity, outer contours, inner contours, shading (normals), and similarity to real smoke structures.*

9.2.2. Evaluation with Users

We have evaluated the authoring tool with 14 users (7 non-artists, 5 hobbyists, 2 professional artists), who either tested the authoring tool or contributed with sketches to our work. All users have at least basic experience with image editing and 3D modeling tools. They read our instructions on the sketch style (see *SI*) and were able to adopt the style already in their very first design tests.

100% found that the method is targeting artists that need to prototype and communicate ideas, while 57% and 42% could see the tool also for high-end products and non-artist use, respectively. 62% reported that the tool is useful for early-stage prototyping, while the remaining 38% find it somewhat useful. 85% rated the reconstruction quality as good, while 60% found that the result matches the expected intent either good or very good. They all rated the time to generate the result as average to fast. Only 1 user reported that the tool is unsatisfactory. The 2 users who had experience with alternative strategies to create density keyframes both reported that they prefer our method over alternatives.

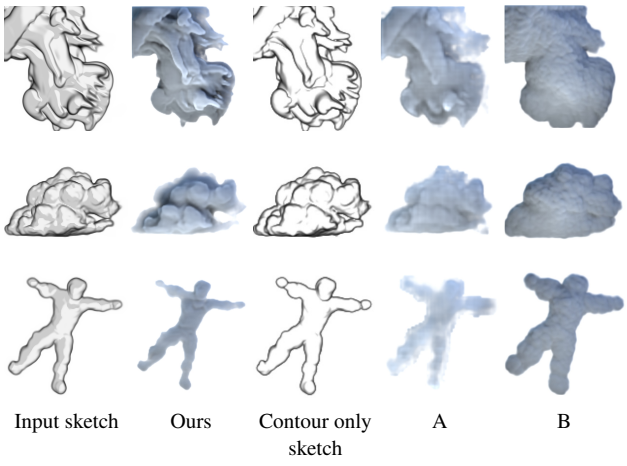


Figure 19: *Comparison with (A) a previous CNN-based reconstruction method [DAI*18] that uses contours as input and (B) a surface-based cloud modeling approach [SBRS10].*

The question of whether the sketch style is intuitive was answered by 64% as good to very good. Only 1 user reported that it took too long to adapt to the imposed style, the others all gave neutral or positive scores. Similarly, only 1 user mentioned that the sketch style negatively impacts artistic creativity. The most negative point was attributed to the sketching of different views, which 40% found difficult, and 30% somewhat difficult or not difficult.

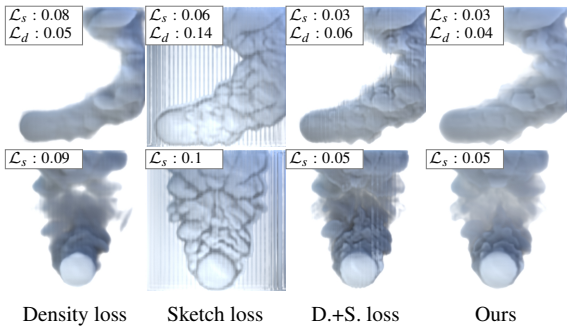


Figure 20: *Evaluation of loss functions. From left to right: density loss, sketch loss, density+sketch losses, density+sketch+depth variation losses (ours). We use the input sketches of Figure 4.*

9.3. Ablation Study

9.3.1. Loss Functions

We evaluated the impact of the different loss functions and illustrate the results in Figure 20. Using only the density loss (Equation

1) like in previous work [DAI*18] we observe large discrepancies between the input sketch and the sketch of the reconstructed density (see also *SI*). When using only the sketch loss (Equation 2) results are very detailed but at the cost of depth ambiguity. If both density and sketch losses are used, the sketch correspondence, as well as the depth reconstruction, are improved, but noise is well visible. Adding the total variation loss in the depth direction (Equation 3) eliminates these artifacts and generates the best results (our model). Figure 20 also embeds a quantitative quality measure (value of density and sketch loss). Lower values stand for higher accuracy.

9.3.2. Recursive Passes

We evaluate our network with various numbers of passes at training and test time. We show the results on front view only in Figure 21, where each row corresponds to 1, 2, and 3 passes during training, respectively, and each column to 1, 3, and 6 successive refinements at test time that we denote by 'f', 'fff' and 'ffffff'. We observed that the difference between training pass 3 and 4 is marginal; we therefore used 3 passes in all our examples. They result in a clearly better reconstruction than the single-pass version, which justifies the use of our rollback loss.

On the other hand, with 3 and more passes during training, the number of successive optimization passes at test time has less influence. The results in 'f' already recovered most of the features of the input sketches and are slightly improved by 'fff', which is hardly distinguishable from 'ffffff'. Next, we alternate front and left view refinement steps (f, fl, flf, flfl) and show the impact on the reconstruction quality in Figure 4. For most examples, a front view optimization followed by a left view refinement (fl) is sufficient; for some examples (*i.e.*, simulated smoke) the quality was further improved by additional passes (flfl).

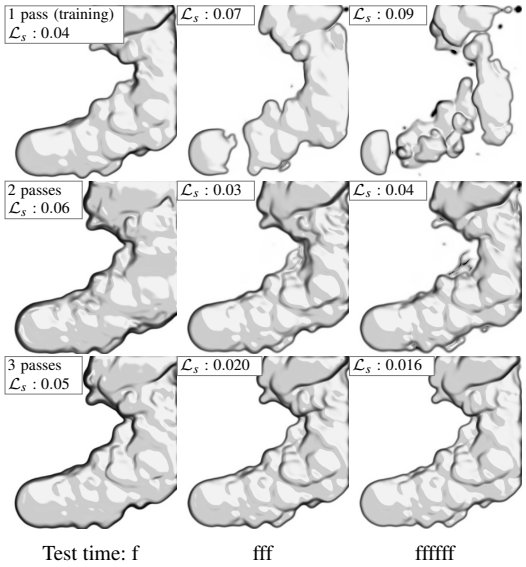


Figure 21: *Evaluation of recursive passes (during training) and inference sequence (at test time). We use 3 passes in practice.*

9.3.3. Sketch Augmentation

We argue that using augmented sketches in the training favors generality. We evaluate this property in Figure 22 and show comparisons of reconstructions from different sketches modified to be far from our initial parameters. On the top row, we input our synthetic sketch of the character scene, where we reduced the brightness and the line width. On the bottom row, we use one of the artist examples that is far from the prescribed style. The middle column shows our results with data augmentation, while the right one shows the results without. Several parts of the sketches were not reconstructed, which shows the need for data augmentation for more generality in the sketch style.

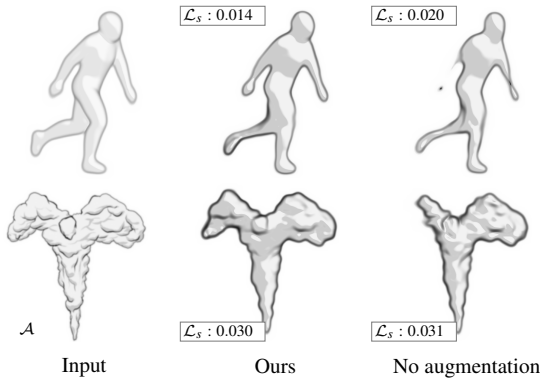


Figure 22: *Using sketch variations in the training improves the robustness of the reconstruction quality. We reduced the brightness and contour width of the sketch of the* character *example (top row) and evaluated the* bird *scene that deviates from the prescribed sketch style (bottom row). From left to right: sketch input, results after training with sketch augmentation, results without augmentation.*

9.3.4. Different Data Set

We justify the need for a new data set by training our model with the state-of-the-art *Scalarflow* data set [EUT19] captured from real smoke. We show in Figure 23 our results with the character scene (top), and the ones obtained after training with Scalarflow (bottom). The results indicate that although Scalarflow exhibits smoke volumes of extremely good quality, it does not embed enough variability to reconstruct arbitrary scenes.

10. Discussion and Conclusion

We presented a method for reconstructing 3D density fields from 2D artist sketches. Our technique fills the currently existing gap between hand-drawn sketches and 3D pre-visualization, allowing an artist for the first time to prototype interactively 3D smoke volumes for testing and communicating ideas. Accordingly, our method targets the early design stage of 3D smoke rather than high-end production. We proposed an updater CNN architecture for optimizing density fields from arbitrary viewpoints, synergistically combined with a differentiable sketch renderer in an end-to-end training. The loss functions are specifically designed for the sketch-to-density

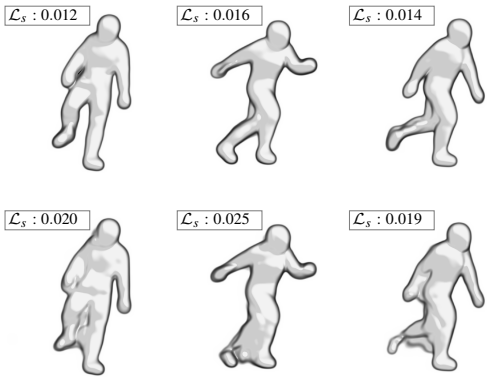


Figure 23: *3D reconstructions computed with a model trained on our data set (top) and Scalarflow (bottom), highlighting the importance of using a well-designed training data set for generic applications.*

problem. The training data set was carefully designed for general applicability, demonstrated by applying the method to diverse density volumes ranging from physics simulations, captured real-world flows, procedural clouds, character animation, and highly non-physical artist sketches.

During the algorithm design, we had in mind that the method must be efficient at test time to enable interactive prototyping, editing, and pre-visualization of smoke animations. This is a prerequisite for future integration into workflows and tools used by artists. Our total inference time per frame is 12.5ms and hence enables real-time previews during sketching. We have implemented a Houdini plugin and demonstrated in interactive authoring sessions that a volumetric smoke shape can be authored by an untrained artist in only a couple of minutes.

We decided to use a high-level sketch style inspired by effects drawing principles, which is easy to use and that allows rapid prototyping of ideas. A general problem of sketch-based modeling is that the simplified representation used in the sketches introduces ambiguities since several possible density fields correspond to a given input sketch. This limitation is especially pronounced when comparing the reconstructed densities with their ground truth counterparts, showing that the reconstructions are smoother than the original volumetric field. The neural post-processing step alleviates this to some extent; it synthesizes details that are visually important but, on the other hand, are physically incorrect (and hence deviate from the ground truth density fields).

Another limitation of our work is that arbitrary drawing styles of artists are not supported. We rely on the fact that artists follow our design principles for the sketching style, which is, for example, using clean contours and two-color toon shading. Due to our sketch augmentation strategy during training, we get a robust reconstruction even if an artist's style slightly deviates from the prescribed one, such as using different contour widths, brightness, or light direction. Since imposing a certain sketch style may hinder creative freedom, future work may address possibilities to relax this constraint, e.g., using a pre-processing network that converts any sketch to our style.

We have evaluated the artists' experience with our method and their perception of its usefulness in production pipelines. The artists were very enthusiastic about the overall idea, the result they achieved, and the potential impact on industrial workflows. They especially liked the idea to use it as a tool to communicate early ideas with the client or art director. As the main drawback the prescribed style and sketching with multiple viewpoints was mentioned. While the sketch style is a limitation of our method, all artists responded positively to the simple use of the system and the robustness of the reconstructions, which can be accounted to the prescribed style.

We have further demonstrated a potential application beyond density keyframe authoring. We used our reconstructed densities as target shapes to guide a physics simulation with the most recent fluid control technique. The resulting animation sequences are smooth and generate realistic flow patterns due to the underlying dynamics.

For production quality authoring higher resolution density fields are needed. Our method does not readily extend to higher resolutions because of memory requirements during training, especially because of our recursive rollback loss. Enabling scalability of the method to higher-resolution settings, for example by using patch-based approaches, could be an interesting avenue for future research.

Acknowledgement

The authors would like to thank Maurizio Nitti (bird sketches and puppy animation), Isabelle Hock (dissolving character), Fraser Rothnie (rhino) for their artistic contributions and Daniel Dorda, Emilie Yu, Capucine Nghiem for testing our tool. The work was supported by the Swiss National Science Foundation under Grant No.: 200021_168997.

References

[BHN07] BRIDSON R., HOURIHAM J., NORDENSTAM M.: Curl-noise for procedural fluid flow. *ACM ToG 26*, 3 (July 2007), 46–es. 6

[CL] Cloud data. URL: www.technology.disneyanimation.com/clouds. 8

[CSGC16] CORDIER F., SINGH K., GINGOLD Y., CANI M.-P.: Sketch-based modeling. In *SIGGRAPH ASIA 2016 Courses* (2016), SA '16. 2

[DAI*18] DELANOY J., AUBRY M., ISOLA P., EFROS A. A., BOUSSEAU A.: 3d sketching using multi-view deep volumetric prediction. *Proc. ACM CGIT 1*, 1 (July 2018). 2, 4, 5, 9, 10, 11, 12

[DFRS03] DECARLO D., FINKELSTEIN A., RUSINKIEWICZ S., SANTELLA A.: Suggestive contours for conveying shape. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22*, 3 (July 2003), 848–855. 6

[EBC*15] ENTEM E., BARTHE L., CANI M.-P., CORDIER F., VAN DE PANNE M.: Modeling 3d animals from a side-view sketch. *Computers & Graphics 46* (2015), 221–230. Shape Modeling International 2014. 1

[EHT18] ECKERT M.-L., HEIDRICH W., THUEREY N.: Coupled Fluid Density and Motion from Single Views. *CGF 37*, 8 (2018), 47–58. 3

[EUT19] ECKERT M.-L., UM K., THUEREY N.: Scalarflow: A large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM ToG 38*, 6 (2019). 3, 12

[FST21] FRANZ E., SOLENTHALER B., THUEREY N.: Global transport for fluid reconstruction with learned self-supervision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2021). 3

[Gil12] GILLAND J.: *Elemental Magic: The Art of Special Effects Animation*. CRC Press, 2012. 2, 3, 9, 11

[HGY17] HAN X., GAO C., YU Y.: Deepsketch2face: A deep learning based sketching system for 3d face and caricature modeling. *ACM Trans. Graph. 36*, 4 (July 2017). 2

[HKYM16] HUANG H., KALOGERAKIS E., YUMER E., MECH R.: Shape synthesis from sketches via procedural models and convolutional networks. *IEEE TVCG 23* (01 2016), 1–1. 2

[HXF*19] HU Z., XIE H., FUKUSATO T., SATO T., IGARASHI T.: Sketch2vf: Sketch-based flow design with conditional generative adversarial network. *CAVW 30* (05 2019), e1889. 3

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *IEEE CVPR* (2016). 5

[JHR*15] JUNG A., HAHMANN S., ROHMER D., BEGAULT A., BOISSIEUX L., CANI M.-P.: Sketching folds: Developable surfaces from non-planar silhouettes. *ACM Trans. Graph. 34*, 5 (2015). 1

[KAGS19] KIM B., AZEVEDO V. C., GROSS M., SOLENTHALER B.: Transport-based neural style transfer for smoke simulations. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 38*, 6 (nov 2019), 1–11. 3, 8

[KAGS20] KIM B., AZEVEDO V. C., GROSS M., SOLENTHALER B.: Lagrangian Neural Style Transfer for Fluids. *ACM Transactions on Graphics (Proc. SIGGRAPH)* (2020). 3

[KAT*19] KIM B., AZEVEDO V. C., THUEREY N., KIM T., GROSS M., SOLENTHALER B.: Deep Fluids: A Generative Network for Parameterized Fluid Simulations. *Computer Graphics Forum 38*, 2 (2019). 5

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *ICLR* (2015). 7

[KYZ14] KAZMI I. K., YOU L., ZHANG J. J.: A survey of sketch based modeling systems. In *CGIV* (2014), pp. 27–36. 2

[LGK*17] LUN Z., GADELHA M., KALOGERAKIS E., MAJI S., WANG R.: 3d shape reconstruction from sketches via multi-view convolutional networks. In *3DV* (2017). 2

[LPL*17] LI C., PAN H., LIU Y., TONG X., SHEFFER A., WANG W.: Bendsketch: Modeling freeform surfaces through 2d sketching. *ACM Trans. Graph. 36*, 4 (July 2017). doi:10.1145/3072959.3073632. 1

[LPL*18] LI C., PAN H., LIU Y., SHEFFER A., WANG W.: Robust flow-guided neural prediction for sketch-based freeform surface modeling. *ACM Trans. Graph. (SIGGRAPH ASIA) 37*, 6 (2018). 2

[LSGV18] LI M., SHEFFER A., GRINSPUN E., VINING N.: Foldsketch: Enriching garments with physically reproducible folds. *ACM Transaction on Graphics 37*, 4 (2018). 1

[LSS*19] LOMBARDI S., SIMON T., SARAGIH J., SCHWARTZ G., LEHRMANN A., SHEIKH Y.: Neural Volumes: Learning Dynamic Renderable Volumes from Images. *ACM TOG 38* (2019), 1–14. 2

[LTJ18] LIU H.-T. D., TAO M., JACOBSON A.: Paparazzi: Surface Editing by way of Multi-View Image Processing. *ACM Transactions on Graphics* (2018). 2

[MF06] MCGUIRE M., FEIN A.: Real-time rendering of cartoon smoke and clouds. NPAR '06, p. 21–26. 6

[MG] Mixamo gallery. URL: www.mixamo.com. 8

[MHZ*15] MIAO Y.-W., HU F., ZHANG X., CHEN J., PAJAROLA R.: SymmSketch: Creating symmetric 3D free-form shapes from 2D sketches. *Computational Visual Media 1*, 1 (March 2015), 3–16. 1

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV* (2020). 2

[NGDA*16] NISHIDA G., GARCIA-DORADO I., ALIAGA D. G., BENES B., BOUSSEAU A.: Interactive sketching of urban procedural models. *ACM Trans. Graph. 35*, 4 (July 2016). 2

[Nit] NITTI M.: Puppy animation. 8

[ODO16] ODENA A., DUMOULIN V., OLAH C.: Deconvolution and checkerboard artifacts. *Distill 1*, 10 (2016), e3. 7

[PHT*13] PAN Z., HUANG J., TONG Y., ZHENG C., BAO H.: Interactive localized liquid motion editing. *ACM TOG 32*, 6 (nov 2013). 3

[QLWQ21] QIU S., LI C., WANG C., QIN H.: A rapid, end-to-end, generative model for gaseous phenomena from limited views. *Computer Graphics Forum* (2021). 3

[SBRS10] STIVER M., BAKER A., RUNIONS A., SAMAVATI F.: Sketch based volumetric clouds. In *Proc. ICSG* (2010), SG'10. 1, 3, 10, 11

[SBS21] SMIRNOV D., BESSMELTSEV M., SOLOMON J.: Learning manifold patch-based representations of man-made shapes. In *International Conference on Learning Representations (ICLR)* (2021). 2

[SMC04] SELLE A., MOHR A., CHENNEY S.: Cartoon rendering of smoke animations. NPAR '04, p. 57–60. 6

[SSISI16] SIMO-SERRA E., IIZUKA S., SASAKI K., ISHIKAWA H.: Learning to simplify: fully convolutional networks for rough sketch cleanup. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 1–11. 7

[SZF*20] SHEN Y., ZHANG C., FU H., ZHOU K., ZHENG Y.: Deepsketchhair: Deep sketch-based 3d hair modeling. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1. 2

[TACS21] TANG J., AZEVEDO V. C., CORDONNIER G., SOLENTHALER B.: Honey, I Shrunk the Domain: Frequency-aware Force Field Reduction for Efficient Fluids Optimization. *Computer Graphics Forum* (2021), to appear. 9, 11

[WB19] WAILLY B., BOUSSEAU A.: Line rendering of 3d meshes for data-driven sketch-based modeling. *Journées Francaises d'Informatique Graphique et de Réalité virtuelle* (2019). 3

[WBC08] WITHER J., BOUTHORS A., CANI M.-P.: Rapid sketch modeling of clouds. SBM'08. 2, 3

[WCPM18] WANG T. Y., CEYLAN D., POPOVIUNDEFINED J., MITRA N. J.: Learning a shared shape space for multimodal garment design. *ACM Trans. Graph. 37*, 6 (Dec. 2018). 2, 3

[WQWF18] WANG L., QIAN C., WANG J., FANG Y.: Unsupervised learning of 3d model reconstruction from hand-drawn sketches. MM '18, p. 1820–1828. 2

[WXCT19] WERHAHN M., XIE Y., CHU M., THUEREY N.: A multi-pass gan for fluid flow super-resolution. *Proc. CGIT* (2019). 2, 4, 5

[XCS*14] XU B., CHANG W., SHEFFER A., BOUSSEAU A., MCCRAE J., SINGH K.: True2form: 3d curve networks from 2d sketches via selective regularization. *ACM TOG (Proc. SIGGRAPH) 33*, 4 (2014). 3

[XFCT18] XIE Y., FRANZ E., CHU M., THUEREY N.: Tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Trans. Graph. 37*, 4 (July 2018). 5

[ZGZS20] ZHONG Y., GRYADITSKAYA Y., ZHANG H., SONG Y.: Deep sketch-based modeling: Tips and tricks. In *3DV* (2020), pp. 543–552. 3

[ZIH*11] ZHU B., IWATA M., HARAGUCHI R., ASHIHARA T., UMETANI N., IGARASHI T., NAKAZAWA K.: Sketch-based dynamic illustration of fluid systems. *ACM Trans. Graph. 30*, 6 (2011), 1–8. 3