Recovering Dynamic 3D Sketches from Videos

Jaeah Lee Changwoon Choi Young Min Kim Jaesik Park*
Seoul National University, Republic of Korea
hayanz@snu.ac.kr changwoon.choi00@gmail.com {youngmin.kim, jaesik.park}@snu.ac.kr

Abstract

Understanding 3D motion from videos presents inherent challenges due to the diverse types of movement, ranging from rigid and deformable objects to articulated structures. To overcome this, we propose Liv3Stroke, a novel approach for abstracting objects in motion with deformable 3D strokes. The detailed movements of an object may be represented by unstructured motion vectors or a set of motion primitives using a pre-defined articulation from a template model. Just as a free-hand sketch can intuitively visualize scenes or intentions with a sparse set of lines, we utilize a set of parametric 3D curves to capture a set of spatially smooth motion elements for general objects with unknown structures. We first extract noisy, 3D point cloud motion guidance from video frames using semantic features, and our approach deforms a set of curves to abstract essential motion features as a set of explicit 3D representations. Such abstraction enables an understanding of prominent components of motions while maintaining robustness to environmental factors. Our approach allows direct analysis of 3D object movements from video, tackling the uncertainty that typically occurs when translating real-world motion into recorded footage. The project page is accessible via: `https://jaeah.me/liv3stroke_web`.

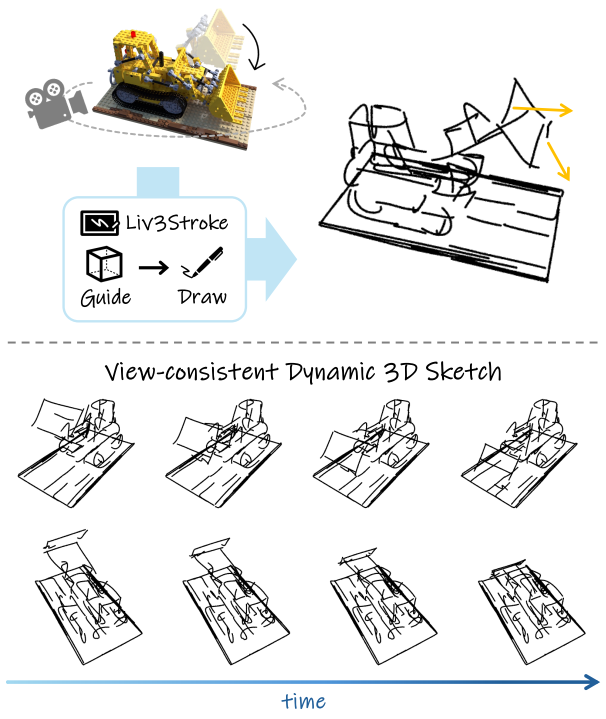


Figure 1. Liv3Stroke is a novel approach that compactly represents object movements using deformable 3D strokes from videos. Our method achieves view-consistent dynamic sketch reconstruction by shifting and deforming the shape of each stroke.

1. Introduction

Tracking 3D movement in video is subject to inherent ambiguity. The 3D motions usually exhibit various forms, including unidentified rigid, deformable, and articulated objects. A video footage observes the 3D motions that are projected onto 2D frames of a moving camera, which further complicates the process of extracting motion components. There have been attempts to understand locally smooth motion patterns, such as motion factorization [31] or segmentation [46]. Recent approaches on novel-view synthesis extend the radiance field formulation into a more general form of dynamic scenes with dense motion fields [25, 34, 44, 47]. However, this formulation solves for complex high-dimensional variables given an under-constrained set-up, and often produces erroneous results that are further deteriorated when the scene is subject to appearance variations.

We find the key insights to express 3D motion through *abstraction* in the form of sketches. Sketches serve as effective tools for compactly visualizing scenes or ideas [10, 11]. While they are subjective expressions, often created by an artist, recent works [41, 42] have shown that we can generate sketches directly from images guided by visual features. They define a perceptual loss in a latent space incorporating deep neural networks and successfully generate abstract sketches. Interestingly, the perceptual loss alleviates enforcing pixel-wise matches in image space, and it

*Corresponding author.

robustly extracts meaningful features even under ambiguous situations without precise alignment. Inspired by this, subsequent works incorporate similar formulations to retrieve view-consistent structural information in 3D [5] or compact representation for dynamic videos [8, 50]. Likewise, we propose that deformable sketch lines can be a flexible yet compact parameterization representing the arbitrary topology of locally smooth 3D motions in videos.

Meanwhile, 3D strokes have the potential to describe the dense field. We can express a 3D concept in a sparse and abstracted form as a wire-like structure [36, 40]. Starting from this property, there have been attempts to apply 3D curves in sketch-based modeling [28] or surface editing [48] to depict intended details. Different from these works, we focus on the "deformability" of each stroke, which enables effectively capturing 3D key features of diverse movements.

To align sketches with dynamic scenes, we define a sketch as a set of deformable 3D strokes and utilize vectorized curves (cubic Bézier curves) for the expression. Using the editing capabilities of vector graphics, we can effectively convey movements by shifting stroke positions and their control points. Before reconstructing moving sketches, we compute a dense guiding motion field since it is challenging to directly align strokes with input video frames. We first reconstruct 3D motion guidance, which is a dynamic point cloud with a deformation network. We propose to optimize motion guidance with rendering loss in perceptual space. This guidance serves as a rough initial 3D motion and location for dynamic 3D strokes. Based on this approach, we fit 3D strokes into movements. We represent motions by individually relocating each curve and adjusting its control points. We minimize the gap between stroke deformation and motion through a coarse-to-fine approach, which enables us to maintain the core shape throughout the movement.

We show its performance in intuitively expressing smooth motion from video frames. Our approach enables to draw core view-consistent object motions in 3D space, and can capture the object's overall shape throughout the entire video sequence while being robust to external factors. We present that our approach can render diverse movements, even if capturing 3D motion from videos is challenging due to the difference between the velocity of camera movements and motions [9].

To summarize, we introduce the following contributions:

- We propose Liv3Stroke, which is the first approach to reconstruct dynamic sketches in 3D space and abstract movements with sparse strokes.
- We define each stroke of the sketch by Bézier curves, and motions are represented by changing each curve's location or shape.
- Our approach can concisely express core structures and motions from natural videos, including monocular video with moving camera poses.
- Our approach draws view-consistent object motions in 3D space and captures their key features while being less affected by environmental factors.

2. Related Work

Dynamic 3D Scene Reconstruction The emergence of neural radiance fields [29] has catalyzed significant advancements in photorealistic 3D reconstruction from multi-view images. Spontaneously, there have been attempts for dynamic scene reconstruction in 3D space. Fridovich *et al.* [7] and Cao and Johnson [4] represent dynamic 3D scenes by Eulerian motion field defined in a space-time 4D grid. Also, there are some works [13, 14, 16, 21, 35, 45] using template model that is specialized in specific objects, such as a parametric human model.

Meanwhile, most existing works [24, 27, 32–34] follow a common pattern; they reconstruct dynamic 3D scenes by optimizing canonical 3D scenes and warp them by learnable deformation fields. Our method follows a similar approach yet focuses on concisely representing motions in 3D space. We aim to recover dynamic sketches by learning the deformation of strokes at each timestep.

Stroke-Based Representation Stroke-based representation expresses scenes with a few strokes. It is one of the simplest sketch representations yet effective in conveying the essential structures and semantics of target objects. There are learning-based methods to synthesize sketches from images [30] by training neural networks on limited image-sketch paired datasets. Optimization-based methods [41, 42] utilize strong prior vision language model [37], and they enable the generation of sketches without training on specific classes. Recently, some works have tackled extending stroke-based representation to the video domain. Gal *et al.* [8] try to animate sketches given a text prompt, and Zheng *et al.* [50] reconstruct abstract dynamic 2D sketches from videos. Furthermore, 3Doodle [5] and EMAP [22] reconstruct 3D strokes from multi-view images, enabling better representation of 3D shapes compared to 2D strokes. However, these works are still limited since they mainly target stationary scenes.

In this paper, we aim to reconstruct motions as sketches in 3D space. We capture dynamic motions from videos by positioning and deforming view-consistent strokes.

3. Method

In this section, we explain how to convey motions with 3D strokes. Our overall method is shown in Fig. 2. We introduce our compact sketch representation for motion abstraction in Sec. 3.1, and Sec. 3.2 details the process of guiding rough motions in 3D space. Based on the motion guidance, we describe dynamic sketches through the process described in Sec. 3.3.

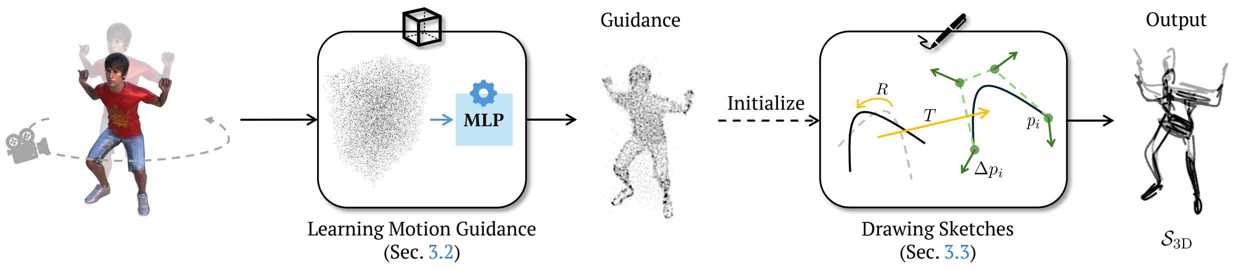


Figure 2. Method overview. We first learn 3D motion guidance from video frames, which are defined as a set of point cloud. Based on this, we can initialize approximated stroke position and motions. We represent movement by transforming each individual stroke through rotation R and translation T, and adjusting its control points $\{p_i\}$ with displacement $\{\Delta p_i\}$, thereby reconstructing a dynamic 3D sketch $\mathcal{S}_{3\text{D}}$.

3.1. Sketch Representation

We define a *sketch* as n black strokes on the white background. Each stroke is represented as a 3D cubic Bézier curve, which is defined with four control points $s_i = \{p_i^j\}_{j=0}^3$, $p_i^j \in \mathbb{R}^3$. To simplify the process, we use fully opaque sketch lines (*i.e.*, fix the opacity as 1), and only adjust the position of control points.

When we project a 3D Bézier curve to the image plane, the represented spline is a 2D rational Bézier curve. However, by assuming that the camera is located sufficiently far from the target object, resulting in negligible perspective distortion, the perspective projection can be approximated as orthographic projection following [5]. Hence, we can consider the projected curve as a general 2D Bézier curve, and use an existing 2D differentiable rasterizer [23] $\mathcal{R}$ to render the projected curves. Each sketch frame $\mathcal{S}$ is mathematically expressed as follows:

$$\mathcal{S} = \mathcal{R}(\Psi(\mathcal{S}_{3\text{D}}, M, K)), \tag{1}$$

where Ψ denotes the projection to a 2D image plane with the extrinsic matrix M and the intrinsic matrix K, and $\mathcal{S}_{3\text{D}} = \{s_i\}_{i=1}^n$ is the 3D sketch defined by a set of curves. Then, we represent the interested object's movements by shifting each curve and its control points.

3.2. Learning 3D Motion Guidance

While existing dynamic scene reconstruction methods target dense photorealistic representation through direct optimization, our goal of compact abstraction demands understanding of the object's essential structure and motion patterns, which cannot be solved by a simple transfer from dense representation. Hence, we first compute approximated 3D motion guidance using low-resolution images. This then helps us determine how the strokes should be located, move, and flow. The detailed framework of this stage is shown in Fig. 3.

We represent the scene using a point cloud $\{P_i\}$ due to their efficiency and ease of manipulation. The motion in the scene is then visualized through the movement of these point

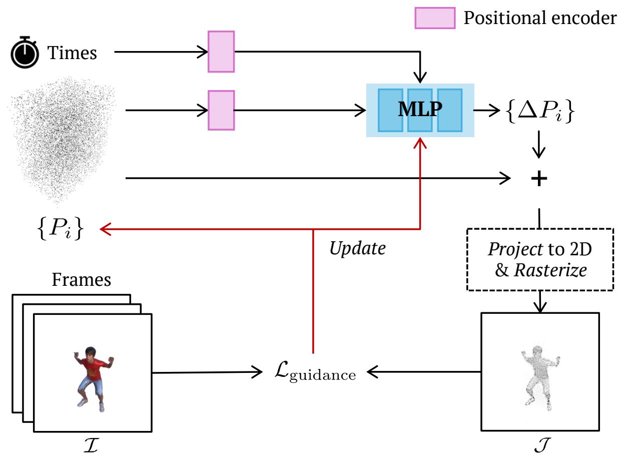


Figure 3. Framework for outlining 3D motion. To get the motion layout, we use a 3D point cloud and compute movements by repositioning its points. An MLP acts as the function that estimates motion $\{\Delta P_i\}$ across the provided video frames.

cloud elements. Like the prior work [34], we implement an MLP-based deformation network that takes time and position encodings as input to estimate spatial deformations at each time step. We use the same positional encoder as in [29] where $\gamma(p) = (\sin(2^l\pi p), \cos(2^l\pi p))_{l=0}^{L-1}$. The projected point is rasterized as a black Gaussian unit. We describe the details of the rasterization process in Appendix.

Meanwhile, we cannot directly compare a projected image $\mathcal{J}$ with the corresponding RGB frame $\mathcal{I}$ since the rendered ouput $\mathcal{J}$ is an image intensity, as shown in Fig. 3, not a natural image as $\mathcal{I}$. Therefore, instead of using pixel-wise loss, we utilize LPIPS loss [49] to get perceptual alignment quality and assess the structural difference:

$$\mathcal{L}^{\text{g}}_{\text{frame}} = \rho(\text{LPIPS}(\mathcal{I}, \mathcal{J}), \alpha, c), \tag{2}$$

where $\mathcal{I}$ and $\mathcal{J}$ are the training image and a rasterized point cloud. $\rho(\mathbf{x}, \alpha, c)$ is a robust function [2] that stabilizes the optimization process by reducing the impact of outliers. The parameters α and c are set to 1 and 0.1, respectively.

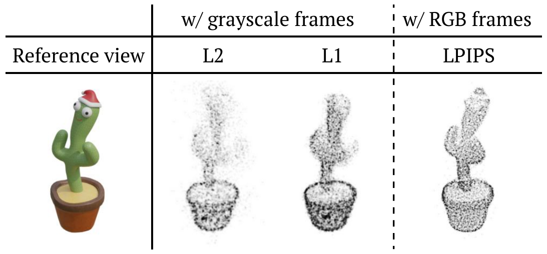


Figure 4. Comparison of loss functions. Compared to pixel-wise losses (L2 and L1 function), LPIPS loss provides stricter structural guidance when computing differences between two images, even with RGB frames.

As shown in Fig. 4, LPIPS loss preserves the structural integrity of the cactus better than L1 or L2 loss. While they produce noisy and scattered representations even with grayscale frames, LPIPS maintains clearer object boundaries and more coherent shape details directly from images.

To achieve smooth motion, we introduce two regularization terms in Sec. 3.2 and Sec. 3.3: a velocity continuity term that ensures stable transitions and a shape stability term that prevents excessive deformation. The former is defined as:

$$\mathcal{L}^{\mathrm{g}}_{\mathrm{temp}} = \| \frac{\Delta P^{t}_{\mathrm{3D}} - \Delta P^{t'}_{\mathrm{3D}}}{t - t'} \|_2, \tag{3}$$

where t' is a neighboring time step of t, randomly sampled within the range $[t - dt, t]$ with dt representing the temporal interval between adjacent frames, and $\{\Delta P^{t}_{\mathrm{3D}}\}$ is changes of point locations at time step t.

For the latter, we need stronger constraints in this step compared to drawing sketches (Sec. 3.3) since we aim to get a "dense" motion field. Hence, we use L1 loss to ensure a more stable shape throughout the entire sequence. We compute the rigid transformation between time step t and t', then compute the following regularization term:

$$\mathcal{L}_{\mathrm{rigid}} = \| {}^{q}R_{t\rightarrow t'} - {}^{q}I \|_1 + \| T_{t\rightarrow t'} \|_1, \tag{4}$$

where $R_{t\rightarrow t'} \in \mathrm{SO}(3)$ and $T_{t\rightarrow t'} \in \mathbb{R}^3$ denote the rigid rotation and translation, respectively. We use the well-known algorithm described in [3] and [12] to compute these terms. The expression ${}^{q}M$ is a corresponding quaternion of the matrix $M \in \mathrm{SO}(3)$ to avoid gimbal lock [38]. The overall objective function is as follows:

$$\mathcal{L}_{\mathrm{guidance}} = \omega_{\mathrm{f}} \mathcal{L}^{\mathrm{g}}_{\mathrm{frame}} + \omega_{\mathrm{t}} \mathcal{L}^{\mathrm{g}}_{\mathrm{temp}} + \omega_{\mathrm{r}} \mathcal{L}_{\mathrm{rigid}}. \tag{5}$$

Following this approach, we can get the hint of 3D movements and initialize canonical stroke locations, *i.e.*, positions before deformation. We further discuss the effect of this obtained guidance in Sec. 4.3.

3.3. Drawing Dynamic Sketches

Based on the extracted motion guidance, we draw motions with 3D strokes by learning the canonical stroke positions (*i.e.*, locations before shifting by the framework) and their deformations. Figure 5 shows the overall pipeline to reconstruct dynamic sketches from input video frames.

Before drawing a moving sketch, we load the guided motion field learned in Sec. 3.2. To initialize curves, we sample points via FPS from the outlier-filtered point cloud at $t = 0$ as the first control points. These points serve as reference markers (*i.e.*, indices) of each curve, and we use their positional encodings as the input of the transformation networks $\mathcal{M}_{\mathrm{R}}$ and $\mathcal{M}_{\mathrm{T}}$ for stroke-wise deformation, along with temporal encodings. The remaining control points are generated progressively by adding a base radius (*e.g.* $r = 0.02$ for *squat* scene) plus random offsets, maintaining a minimum distance ($\delta = 1.0 \times 10^{-3}$) between consecutive points. We initialize $\mathcal{M}_{\mathrm{T}}$ using the weights from our previously trained MLP (described in Sec. 3.2), since both networks handle per-stroke translation.

We model stroke movements as the composition of two components: (1) per-stroke rigid transformations controlling position and orientation and (2) control point adjustments that manage shape changes. Each component is computed through its corresponding MLP. The rotation $R^t_i \in \mathrm{SO}(3)$ and translation $T^t_i \in \mathbb{R}^3$ of the i-th stroke at the time step t is computed as follows:

$$R^t_i = \zeta(\mathcal{M}_{\mathrm{R}}(\gamma(P_i), \gamma(t))), T^t_i = \mathcal{M}_{\mathrm{T}}(\gamma(P_i), \gamma(t)) \tag{6}$$

where $\gamma(\cdot)$ indicates the positional encoder as in Sec. 3.2, and $\zeta(\cdot)$ represents the conversion of angles from quaternion to rotation matrix. Then, each control point of the stroke is transformed as $q^{tj}_{i} = R^t_i p^j_i + T^t_i$ where $s_i = \{p^j_i\}$. Using these relocated strokes, we compute the changes in control point positions $\{\Delta p^{tj}_{i}\}$ as the following equation to deform each curve's control points:

$$\Delta p^{tj}_{i} = \mathcal{M}_{\mathrm{L}}(\gamma(q^{tj}_{i}), \gamma(t)), \tag{7}$$

where $\mathcal{M}_{\mathrm{L}}$ is another network to compute additional changes of each control point. This optimization process employs a coarse-to-fine approach to efficiently capture global and local features. Initially, the method operates at low resolution to identify the overall structure and key patterns by learning $\mathcal{M}_{\mathrm{R}}$ and $\mathcal{M}_{\mathrm{T}}$ in the coarse stage. As the resolution increases, we train $\mathcal{M}_{\mathrm{L}}$ to capture finer details in the fine stage progressively. The canonical location of control points $\{p^j_i\}$ are optimized throughout all stages.

The final 3D sketch at the time step t is:

$$\mathcal{S}^t_{\mathrm{3D}} = \mathcal{S}^0_{\mathrm{3D}} + \xi(\Delta \mathcal{S}^t_{\mathrm{3D}}), \tag{8}$$

where $\mathcal{S}^0_{\mathrm{3D}}$ is the canonical state and $\Delta \mathcal{S}^t_{\mathrm{3D}}$ represents the accumulated changes throughout the framework. The function $\xi(\mathbf{x}) = \frac{\mathbf{x}}{1+\exp(-a(|\mathbf{x}|-b))}$, with constants a and b, suppresses small movements during static periods. We demonstrate the effect of this correction in Appendix.

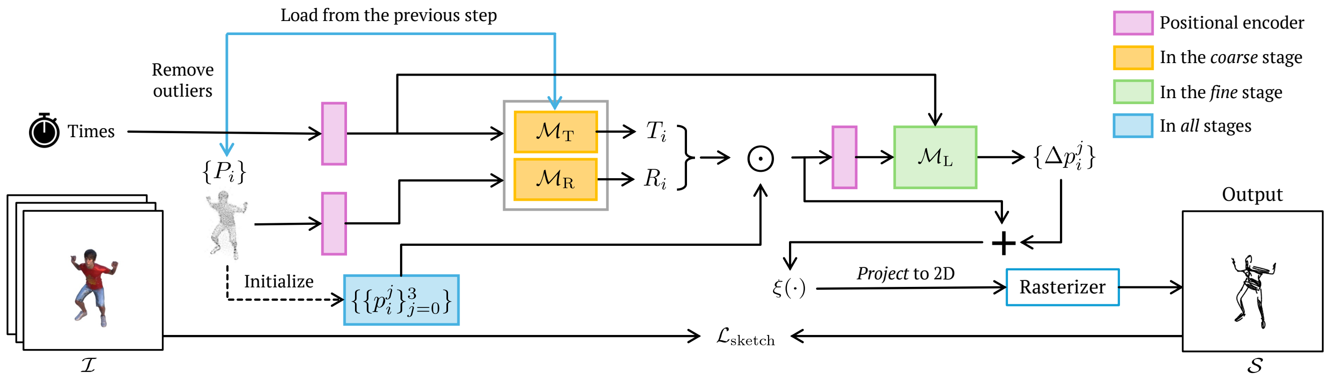


Figure 5. Pipeline for rendering sketches. We extract the motion as sketches by learning the deformation of each stroke. To achieve this, we separate stroke motions as (1) per-stroke transformations, which are composed of rotation R_i and translation T_i and (2) shifting each control point of the stroke $\{\Delta p_i^j\}$. We use MLPs, $\mathcal{M}_\mathrm{R}$, $\mathcal{M}_\mathrm{T}$, and $\mathcal{M}_\mathrm{L}$, as the function of deformation at the given time step.

We use the following perceptual loss in the whole process since we need semantic understanding to convey sketches [41], which is the same as 3Doodle [5]:

$$\mathcal{L}^\mathrm{s}_\mathrm{frame} = \lambda_\mathrm{s}\rho(\mathrm{LPIPS}(\mathcal{I},\mathcal{S}),\alpha,c) + \mathrm{dist}(\mathrm{CLIP}(\mathcal{I}),\mathrm{CLIP}(\mathcal{S})), \tag{9}$$

where $\mathrm{CLIP}(\cdot)$ symbolizes extracted from CLIP [37] image encoder and $\mathrm{dist}(x,y) = 1 - \frac{x\cdot y}{\|x\|\cdot\|y\|}$ indicates the cosine distance. We adopt the same parameter values ($\alpha = 1$ and $c = 0.1$) of the robust function ρ as in Sec. 3.2.

As mentioned in Sec. 3.2, we additionally use regularization terms to enhance quality. For velocity continuity, we define the following objective function with the overall changes $\Delta\mathcal{S}^t_\mathrm{3D}$:

$$\mathcal{L}^\mathrm{s}_\mathrm{temp} = \lambda_\mathrm{t}\|\frac{\Delta\mathcal{S}^t_\mathrm{3D} - \Delta\mathcal{S}^{t'}_\mathrm{3D}}{t-t'}\|_2, \tag{10}$$

where t' is a a neighboring time step of t as same in Eq. 3. The other term for stabilized structure is as follows:

$$\mathcal{L}_\mathrm{reg} = \begin{cases} \lambda_\mathrm{r}(\|{}^qR_i - {}^qI\|_2 + \|T_i\|_2), & \text{if } \textit{coarse} \text{ stage} \\ \lambda_\mathrm{l}\|\Delta p_i^j\|_2, & \text{otherwise} \end{cases} \tag{11}$$

where R_i and T_i are the per-stroke rotation and translation, and Δp_i^j is the local movement of the control point computed by $\mathcal{M}_\mathrm{L}$. In addition, ${}^qM$ denotes the quaternion representation of the matrix M, as mentioned in Sec. 3.2.

Finally, we draw the motion as sketches using the following objective function:

$$\mathcal{L}_\mathrm{sketch} = \mathcal{L}^\mathrm{s}_\mathrm{frame} + \mathcal{L}^\mathrm{s}_\mathrm{temp} + \mathcal{L}_\mathrm{reg}. \tag{12}$$

We further discuss the effect of each term in Sec. 4.3.

4. Experiment

Implementation Details We optimize all networks and sketch parameters using Adam optimizer [20]. Moreover, in Sec. 3.2, we learn motion guidance starting from a point cloud with $10k$ points. We assign values for Eq. 5 as $\omega_\mathrm{f} = 0.1$, $\omega_\mathrm{t} = 0.05$, and $\omega_\mathrm{r} = 1.0\times10^{-4}$. Similarly, for the equations in Sec. 3.3, we set the following parameters to each term of the objective function: $\lambda_\mathrm{s} = 0.01$ for Eq. 9, $\lambda_\mathrm{t} = 0.01$ for Eq. 10, and $\lambda_\mathrm{r} = \lambda_\mathrm{l} = 1.0\times10^{-3}$ for Eq. 11. Moreover, we set $a = 100$ and $b = 0.05$ for the correction function $\xi(\cdot)$ in Eq. 8. Our perceptual distance in Eqs. (2) and (9) employ VGG16 model for LPIPS loss [49]. We also use features from a pretrained RN101 model of CLIP encoder [37] for Eq. 9 in sketch synthesis. Users can control the level of detail by setting the number of strokes. More implementation details can be found in the Appendix.

Datasets The inputs are video frames capturing moving objects from a non-stationary camera, along with viewpoints and timesteps. Since the D-NeRF [34] dataset lacks camera motion, we rendered a new synthetic dataset with a consistent camera trajectory. Each scene includes 100 frames with ground-truth camera parameters and images, and all viewpoints are on an upper hemisphere enclosing the target objects. We also evaluate our model on the real-world scenes from [17], [26], and [32], videos where the camera captures the object while in motion.

Baselines To the best of our knowledge, we are the first to abstract objects in motion as sketches in 3D space. Hence, we mainly assess our results by comparing with three baseline methods with varying input/output stroke representations. Two of them aim to generate 2D sketches. CLIPasso [41] create an abstract sketch with 2D strokes from a single image, and Zheng *et al.* [50] introduce a framework that generates dynamic sketches from videos. On the other hand, Suggestive Contours [6] derives capture geometric contour features from 3D mesh. Since we have only a 2D video sequence, we first extract meshes using the state-of-the-art dynamic mesh reconstruction [26] for this baseline.

Method	Strcutural alignment (↑) Novel views	Strcutural alignment (↑) Fixed views	Motion prompt similarity (↑) Novel views	Motion prompt similarity (↑) Fixed views
CLIPasso	0.760 ± 0.107	0.740 ± 0.127	0.659 ± 0.007	0.664 ± 0.011
Sketch Video Syn.	0.663 ± 0.115	0.657 ± 0.135	0.654 ± 0.011	0.658 ± 0.011
Sugg. Contours	0.784 ± 0.102	0.750 ± 0.119	0.661 ± 0.013	0.656 ± 0.016
Liv3Stroke (Ours)	0.693 ± 0.096	0.683 ± 0.108	0.656 ± 0.006	0.656 ± 0.008

(a) Quantitative results of dynamic 3D sketches.

Method	Per-frame Chamfer (↓)	Motion velocity distance ($\times 10^{-3}$) (↓)
4DGS	0.205 ± 0.046	4.60 ± 3.00
Deformable 3DGS	0.302 ± 0.170	5.05 ± 4.06
SC-GS	0.294 ± 0.055	4.21 ± 2.75
DG-Mesh	0.277 ± 0.059	4.10 ± 2.70
Liv3Stroke (Ours)	0.252 ± 0.049	4.16 ± 2.34

(b) Quantitative results of 3D guidance accuracy.

Table 1. Overall quantitative results. **(a)** Quantitative metrics on sketch video frames. We evaluate structural alignment score with edge images of referenced frames and CLIP feature-based motion prompt similarity. **(b)** Quantitative results of 3D guidance accuracy. Our approach can capture meaningful 3D motion information as the point cloud sequence, even though we do not pursue realistic scenes.

Additionally, we compare GS-based [19] dynamic scene reconstruction works [15, 26, 44, 47] to check how well the output describes the motion. Note that DG-Mesh [26] aims to extract mesh starting from Gaussian splatting techniques, while the others mainly focus on realistic dynamic scene reconstruction in 3D space.

4.1. Quantitative Results

We provide quantitative results of sketch videos in Tab. 1 (a). In this table, we evaluate how well the sketch preserves the overall structure and the desired movement. We evaluate both aspects from novel camera trajectories and fixed viewpoints. For structure preservation, we employ MS-SSIM [43] metrics between generated sketch frames and their corresponding reference edge views, which are extracted from PiDiNet [39]. For motion expression quality, we compute CLIP [37] feature similarities between each frame and the text prompt structured as "A sketch of $\{movement\}$", and normalize values to $[0, 1]$. Note that we use ViT-B/32 [1] models in this evaluation, which remain independent from our optimization pipeline to ensure unbiased evaluation.

We observe that our approach has the smallest deviations in both scenarios, showing that Liv3Stroke can consistently capture the structure throughout the whole input sequence. Sketch Video Synthesis achieves the lowest scores in this validation since it mainly focuses on depicting detailed features rather than capturing the whole movement. While CLIPasso performs well in this validation, its higher deviations show that it lacks consistency in structure throughout the sequence. Similarly, Suggestive Contours achieve the highest scores, as it tend to generate outline contours as illustrated in Fig. 6. We further discuss about this in Sec. 4.2.

In motion prompt similarity scores, Liv3Stroke achieves consistent scores with the smallest deviation in both novel camera trajectory and fixed viewpoint scenarios compared to other approaches. This demonstrates that our approach maintains stable performance in motion expression while being robust to varying camera viewpoints. Furthermore, Sketch Video Synthesis achieves the lowest score in the camera-moving scenario compared to the second-highest score when the camera is stationary. This is because they are dependent on layered neural atlas [18], which cannot represent big movements.

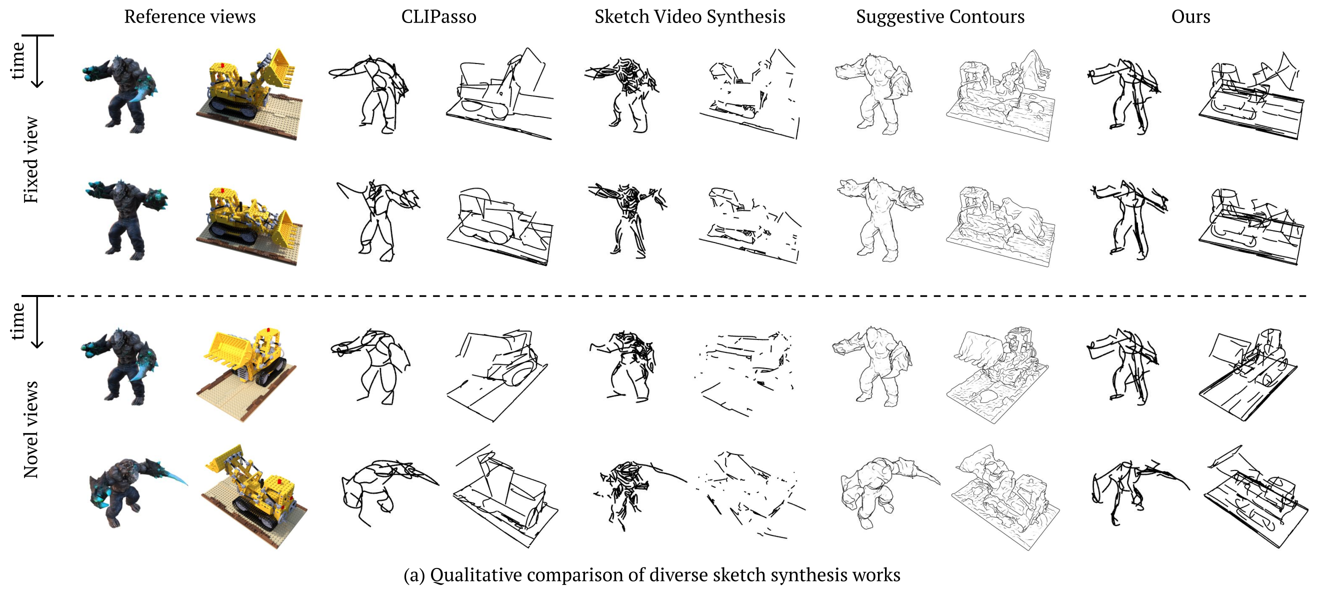
We also evaluate motion guidance performance described in Sec. 3.2. Table 1 (b) presents motion accuracy comparisons between GS-based dynamic approaches [15, 26, 44, 47] and our method, measured on our synthetic dataset. Using point clouds extracted from mesh at each time step as ground truth, we assess the performance through two metrics: (1) per-frame structural accuracy measured by Chamfer distance between point clouds and (2) motion velocity accuracy calculated by L2 loss between ground truth and predicted point position changes at each time step. Although 4DGS achieves better per-frame structural accuracy scores than ours, their overall structure deteriorates as shown in Fig. 7. We discuss this phenomenon further in Sec. 4.2.

Meanwhile, Liv3Stroke performs similarly to DG-Mesh but with fundamentally different objectives. While DG-Mesh focuses on precise mesh reconstruction, Liv3Stroke aims for efficient motion capture by simply repositioning point clouds to obtain approximate 3D motion information. This distinction in goals highlights the efficiency of our approach since we achieve similar quantitative results despite using a simpler approach focused on motion rather than detailed geometry.

4.2. Qualitative Results

We provide the qualitative results of our sketches in Figs. 1 and 6. In Fig. 6 (a), we compare our method against existing baselines under both fixed viewpoints and novel camera trajectories.

From these scenarios, our method exhibits several key advantages. CLIPasso demonstrates inconsistent structural details and unstable stroke placements across all scenarios. While Sketch Video Synthesis achieves better temporal coherence from fixed viewpoints, it struggles to maintain structural integrity during camera motion. These limitations stem from their primary focus on 2D synthesis, resulting in an insufficient understanding of 3D motions. Furthermore, it struggles with the *lego* scene in Fig. 6 (a), as it



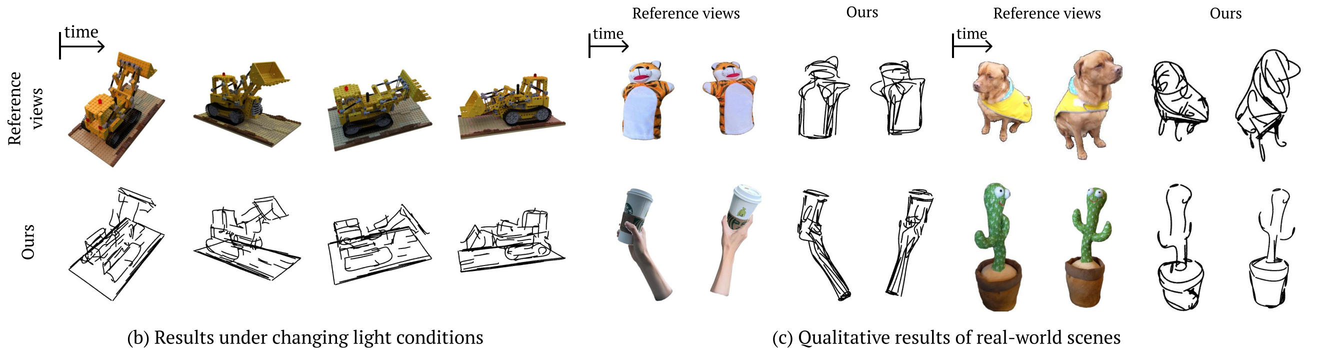

time
Reference views
Ours
Reference views
Ours
time
Reference views
Ours
time
(b) Results under changing light conditions
(c) Qualitative results of real-world scenes

Figure 6. Qualitative results of sketches. **(a)** Comparison with baselines. Our approach can generate sketches with view-consistent motions from RGB frames simply by locating and deforming each stroke. Note that Suggestive Contours [6] requires the input mesh, hence cannot capture motions directly from videos. **(b)** Liv3Stroke also can represent 3D movements even when RGB values change due to the surrounding environment. **(c)** Results of real-world scenes. Our approach successfully captures movements in real-world scenarios.

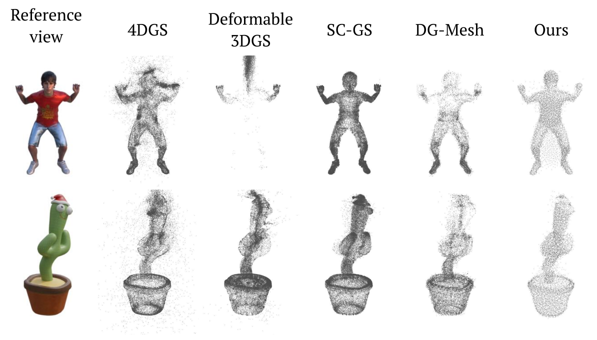


Figure 7. Qualitative results on guidance. Our approach captures clearer structure compared to photorealistic reconstruction methods despite not primarily aiming for accurate scene reproduction.

tends to produce overly detailed representations instead of focusing on essential features. Although Suggestive Contours generates more structured results, it requires meshes for drawing, making it heavily rely on the quality of input meshes. Additionally, the results tend toward detailed depiction rather than abstraction. In contrast, our approach successfully represents movements in both fixed and moving camera scenarios, while maintaining consistent structural integrity throughout the motion sequence. The advantage of our method becomes particularly evident in novel camera trajectories, where it robustly preserves 3D structural features across different viewpoints.

Figure 6 (b) shows our sketch representation's robustness to lighting conditions, a significant external factor that affects frame RGB values. Liv3Stroke maintains consistent sketch video generation while preserving core structure and motion, even with varying lighting colors throughout the sequence.

In addition, our approach can also effectively represent real-world scenarios as in Fig. 6 (c). It can capture various objects and their movements. Despite the complexity of real-world scenes, it preserves key structural characteristics of each object. We highly recommend finding the supplement

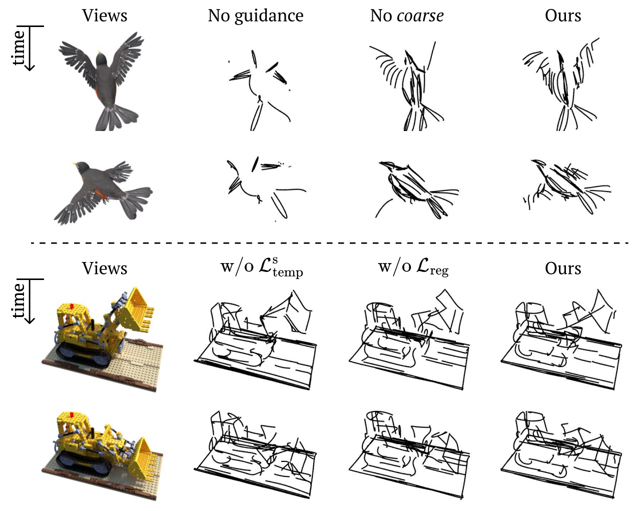

Views
No guidance
No *coarse*
Ours
time
Views
w/o $\mathcal{L}_{\text{temp}}^{\text{s}}$
w/o $\mathcal{L}_{\text{reg}}$
Ours
time

Figure 8. Ablation study on design choices in sketch reconstruction.

	Per-frame Chamfer (↓)	Motion velocity distance ($\times 10^{-3}$) (↓)
w/o $\mathcal{L}_{\text{temp}}^{\text{g}}$	0.258 ± 0.043	6.81 ± 5.41
w/o $\mathcal{L}_{\text{rigid}}$	0.253 ± 0.048	5.45 ± 3.74
L2 in $\mathcal{L}_{\text{rigid}}$	0.253 ± 0.045	4.61 ± 2.94
Ours (mask)	0.257 ± 0.049	4.12 ± 2.84
Ours (grayscale)	0.249 ± 0.039	4.41 ± 2.46
Liv3Stroke (Ours)	0.252 ± 0.049	4.16 ± 2.34

Table 2. Ablation study on generating motion guidance.

to see more results.

Meanwhile, Fig. 7 compares our point cloud reconstruction with other methods. We observe that 4DGS exhibits noisy point distributions and Deformable 3DGS struggles with overall structure preservation, particularly visible in the elongated artifacts. While DG-Mesh demonstrates higher performance in 3D shape reconstruction due to its mesh-focused approach, it shows concentrated point distribution in certain areas. Compared to these methods, our method successfully maintains structural integrity during object motion with uniform point distribution despite optimizing for image intensity rather than realistic reconstruction objectives.

4.3. Ablation study

We further validate the details of our framework through an ablation study. Figure 8 shows the qualitative results of design choices in generating sketches.

As observed, directly optimizing strokes without guidance leads to disconnected line segments that barely capture the motion. Without coarse guidance, while the overall shape is preserved, the output lacks coherent structure and key features, such as wings of the bird. Our full framework generates the most balanced results, maintaining global postures and fine details. Similarly, it is difficult to capture the key motion when learning without $\mathcal{L}_{\text{temp}}^{\text{s}}$. As shown in Fig. 8, we observe unstable stroke variations in the generated sketches even when there are no changes between input frames. Without $\mathcal{L}_{\text{reg}}$, although motion dynamics are preserved, the model struggles to capture the essential structural characteristics of the object. Compared to these results, our complete model reconstructs sketches with the best structural representation and expresses stable movements, demonstrating how each component contributes to the final sketch quality.

We also conduct an ablation study to analyze the effectiveness of each component in our motion guidance generation framework, as described in Sec. 3.2. Table 2 shows the quantitative results measuring per-frame structure and motion velocity accuracy. Our full framework achieves the best performance among the conditions related to terms in the objective function. Without $\mathcal{L}_{\text{temp}}^{\text{g}}$, the model shows degraded performance, especially in motion velocity distance per time step, indicating the importance of temporal coherence in extracting motion guidance. When removing $\mathcal{L}_{\text{rigid}}$, we observe similar performance in structural similarity but decreased motion velocity accuracy. We can interpret this that $\mathcal{L}_{\text{rigid}}$ helps capture consistent motion patterns. Using L2 loss instead of our proposed $\mathcal{L}_{\text{rigid}}$ function shows a slightly high motion velocity distance, demonstrating the effectiveness of our rigid loss formulation. Furthermore, our approach ensures the accuracy of both 3D structure and motion when using RGB images instead of masks or grayscale frames. We provide more comparisons in Appendix.

5. Conclusion

In this work, we introduce Liv3Stroke, a novel approach that bridges the gap between video analysis and motion abstraction by representing 3D object movements through dynamic stroke manipulation. Our method demonstrates that diverse movements can be effectively represented with sparse strokes by relocating and deforming them.

Looking forward, our work opens new possibilities for understanding scene flow dynamics and estimating large-scale motion fields, potentially advancing computer vision and motion analysis domains. In addition, building on our approach's representation of motion through 3D strokes, stroke-based physical control could be achieved by discretizing the dense field and designing functions that map physical properties between strokes and the field.

Limitations Since we only consider view-independent strokes, our approach cannot render view-dependent representations such as a contour of the rounded object. We expect to overcome these limitations by adopting view-dependent strokes with superquadrics as proposed in 3Doodle [5].

Acknowledgments

This work was supported by IITP grant (RS-2021-II211343: AI Graduate School Program at Seoul National University) (5%) and No.2021-0-02068: Artificial Intelligence Innovation Hub (45%)) and NRF grant (No. 2023R1A1C200781211 (55%)) funded by the Korea government (MSIT).

References

[1] Dosovitskiy Alexey, Beyer Lucas, Kolesnikov Alexander, Weissenborn Dirk, Zhai Xiaohua, Unterthiner Thomas, Dehghani Mostafa, Minderer Matthias, Heigold Georg, Gelly Sylvain, Uszkoreit Jakob, and Houlsby Neil. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 6

[2] Jonathan T Barron. A general and adaptive robust loss function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4331–4339, 2019. 3

[3] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, pages 586–606. Spie, 1992. 4

[4] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2023. 2

[5] Changwoon Choi, Jaeah Lee, Jaesik Park, and Young Min Kim. 3doodle: Compact abstraction of objects with 3d strokes. *ACM Transactions on Graphics*, 43(4):1–13, 2024. 2, 3, 5, 8, 12

[6] Doug DeCarlo, Adam Finkelstein, Szymon Rusinkiewicz, and Anthony Santella. Suggestive contours for conveying shape. *ACM Transactions on Graphics*, 22(3):848–855, 2003. 5, 7, 13

[7] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 2

[8] Rinon Gal, Yael Vinker, Yuval Alaluf, Amit Bermano, Daniel Cohen-Or, Ariel Shamir, and Gal Chechik. Breathing life into sketches using text-to-video priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4325–4336, 2024. 2

[9] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *Advances in Neural Information Processing Systems*, 35:33768–33780, 2022. 2

[10] Yulia Gryaditskaya, Mark Sypesteyn, Jan Willem Hoftijzer, Sylvia C Pont, Frédo Durand, and Adrien Bousseau. Opensketch: a richly-annotated dataset of product design sketches. *ACM Transactions of Graphics*, 38(6):232–1, 2019. 1

[11] Aaron Hertzmann. Why do line drawings work? a realism hypothesis. *Perception*, 49(4):439–451, 2020. 1

[12] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the optical society of America A*, 4(4):629–642, 1987. 4

[13] Liangxiao Hu, Hongwen Zhang, Yuxiang Zhang, Boyao Zhou, Boning Liu, Shengping Zhang, and Liqiang Nie. Gaussianavatar: Towards realistic human avatar modeling from a single video via animatable 3d gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 634–644, 2024. 2

[14] Shoukang Hu, Tao Hu, and Ziwei Liu. Gauhuman: Articulated gaussian splatting from monocular human videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20418–20431, 2024. 2

[15] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4220–4230, 2024. 6, 12

[16] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. In *European Conference on Computer Vision*, pages 402–418. Springer, 2022. 2

[17] Erik Johnson, Marc Habermann, Soshi Shimada, Vladislav Golyanik, and Christian Theobalt. Unbiased 4d: Monocular 4d reconstruction with a neural deformation model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6598–6607, 2023. 5

[18] Yoni Kasten, Dolev Ofri, Oliver Wang, and Tali Dekel. Layered neural atlases for consistent video editing. *ACM Transactions on Graphics*, 40(6):1–12, 2021. 6

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions of Graphics*, 42(4):139–1, 2023. 6

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[21] Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. Hugs: Human gaussian splats. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 505–515, 2024. 2

[22] Lei Li, Songyou Peng, Zehao Yu, Shaohui Liu, Rémi Pautrat, Xiaochuan Yin, and Marc Pollefeys. 3d neural edge reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21219–21229, 2024. 2

[23] Tzu-Mao Li, Michal Lukáč, Gharbi Michaël, and Jonathan Ragan-Kelley. Differentiable vector graphics rasterization for editing and learning. *ACM Transactions of Graphics*, 39(6): 193:1–193:15, 2020. 3

[24] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2

[25] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 1

[26] Isabella Liu, Hao Su, and Xiaolong Wang. Dynamic gaussians mesh: Consistent mesh reconstruction from dynamic scenes. In *ICLR*, 2025. 5, 6

[27] Yu-Lun Liu, Chen Gao, Andréas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13–23, 2023. 2

[28] Zhongjin Luo, Jie Zhou, Heming Zhu, Dong Du, Xiaoguang Han, and Hongbo Fu. Simpmodeling: Sketching implicit field to guide mesh modeling for 3d animalmorphic head design. In *The 34th annual ACM symposium on user interface software and technology*, pages 854–863, 2021. 2

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2, 3

[30] Umar Riaz Muhammad, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy M Hospedales. Learning deep sketch abstraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8014–8023, 2018. 2

[31] Atsuhiro Noguchi, Umar Iqbal, Jonathan Tremblay, Tatsuya Harada, and Orazio Gallo. Watch it move: Unsupervised discovery of 3d joints for re-posing of articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3677–3687, 2022. 1

[32] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2, 5

[33] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: a higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions of Graphics*, 40(6), 2021.

[34] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1, 2, 3, 5, 11

[35] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20299–20309, 2024. 2

[36] Zhiyu Qu, Lan Yang, Honggang Zhang, Tao Xiang, Kaiyue Pang, and Yi-Zhe Song. Wired perspectives: Multi-view wire art embraces generative ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021. 2, 5, 6

[38] Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985. 4

[39] Zhuo Su, Wenzhe Liu, Zitong Yu, Dewen Hu, Qing Liao, Qi Tian, Matti Pietikäinen, and Li Liu. Pixel difference networks for efficient edge detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5117–5127, 2021. 6

[40] Kenji Tojo, Ariel Shamir, Bernd Bickel, and Nobuyuki Umetani. Fabricable 3d wire art. In *ACM SIGGRAPH 2024 Conference Proceedings*, 2024. 2

[41] Yael Vinker, Ehsan Pajouheshgar, Jessica Y Bo, Roman Christian Bachmann, Amit Haim Bermano, Daniel Cohen-Or, Amir Zamir, and Ariel Shamir. Clipasso: Semantically-aware object sketching. *ACM Transactions on Graphics*, 41(4):1–11, 2022. 1, 2, 5, 12, 13

[42] Yael Vinker, Yuval Alaluf, Daniel Cohen-Or, and Ariel Shamir. Clipascene: Scene sketching with different types and levels of abstraction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4146–4156, 2023. 1, 2

[43] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, pages 1398–1402. Ieee, 2003. 6

[44] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. 1, 6, 12

[45] Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. Gaussian head avatar: Ultra high-fidelity head avatar via dynamic gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1931–1941, 2024. 2

[46] Kaizhi Yang, Xiaoshuai Zhang, Zhiao Huang, Xuejin Chen, Zexiang Xu, and Hao Su. Movingparts: Motion-based 3d part discovery in dynamic radiance field. In *ICLR*, 2024. 1

[47] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20331–20341, 2024. 1, 6, 12

[48] Emilie Yu, Rahul Arora, J Andreas Baerentzen, Karan Singh, and Adrien Bousseau. Piecewise-smooth surface fitting onto unstructured 3d sketches. *ACM Transactions on Graphics*, 41 (4):1–16, 2022. 2

[49] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 3, 5

[50] Yudian Zheng, Xiaodong Cun, Menghan Xia, and Chi-Man Pun. Sketch video synthesis. In *Computer Graphics Forum*, page e15044. Wiley Online Library, 2024. 2, 5, 13

Recovering Dynamic 3D Sketches from Videos

Supplementary Material

A. Methodological Details

A.1. Point Cloud Rasterization

To rasterize the point cloud to an image plane in Sec. 3.2, we first project the 3D points $P_{3D} = \{P_i\}$ onto the 2D space $P_{2D} = \{\tilde{P}_i\}$. For each pair of i-th normalized grid point and j-th point of the normalized point cloud $\hat{P}_{2D}$ to image dimensions, we use the Gaussian function to compute a rendered intensity J_{ij}:

$$J_{ij} = \exp\left(-\frac{D_{ij}^2}{2\sigma_j^2}\right), \tag{13}$$

where D_{ij} is the Euclidean distance between two points and σ_j indicates the point size factor that controls the contribution area of each point.

We dynamically adjust σ_j based on the depth of the point $\{d_j\}$ to account for perspective projection effects. Points farther from the camera are rendered with smaller sizes, following standard 3D rendering principles. Using the normalized depth $\{\hat{d}_j\} = \{\frac{d_j - d_{min}}{d_{min} - d_{max}}\}$, each point size σ_j is computed as:

$$\sigma_j = \frac{\mu}{0.5\mu \min(W, H)} \times \hat{d}_j \times \beta, \tag{14}$$

where μ and β denote the scaling and deblurring factor, and W, H are the image width and height. We set $\mu = 10$ and $\beta = 0.5$.

We aggregate the Gaussian contributions from all point cloud points to each grid point to generate the final rendered image. The intensity value for each pixel is computed by summing these contributions. We then normalize the intensities by dividing by the maximum value, ensuring the final image $\mathcal{J} \in \mathbb{R}^{H\times W}$ values fall within an appropriate range for visualization or processing. This process can be expressed as the following equation:

$$\mathcal{J}_i = \frac{\sum_{j=1}^{M} R_{ij}}{\max(\sum_{j=1}^{M} R_{ij})}. \tag{15}$$

The results can be shown in Figs. C and F. Note that each guidance view in Fig. C is rasterized into a 100×100 resolution image, which represents the actual resolution used for generating motion guidance in synthetic scenes.

A.2. Effect of the Suppression Function $\xi(\cdot)$

As described in Sec. 3.3, we adjust the suppression function $\xi(\cdot)$ to prevent unintended stroke movements in sketch synthesis. Figure A shows the effect of this function. We observe that the model struggles to suppress undesired stroke movements even when no motion occurs. This demonstrates that our full approach achieves higher performance in extracting core motions.

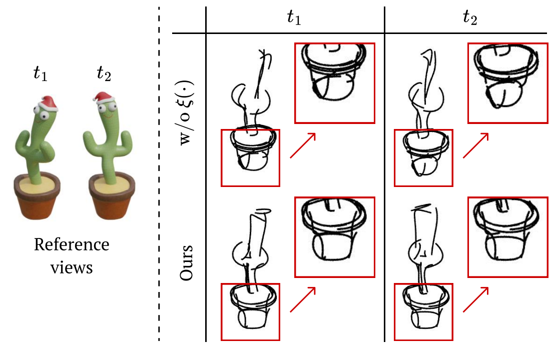
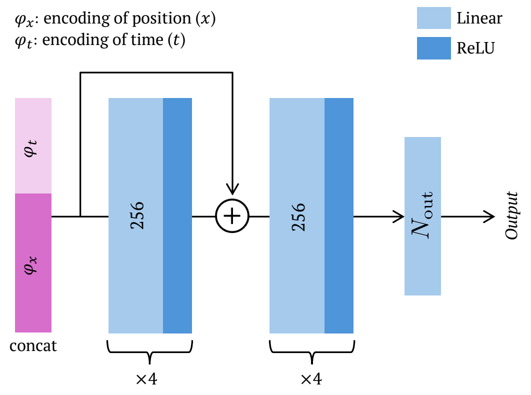


Figure A. Effectiveness of the function for suppression $\xi(\cdot)$. Without motion suppression, we observe noisy stroke movement at different time steps (t_1 and t_2) even if there are no motions.



φ_x: encoding of position (x)
φ_t: encoding of time (t)
Linear
ReLU
φ_t
φ_x
concat
256
×4
256
×4
N_{out}
Output

Figure B. Network architecture. All networks in the framework share the same architecture. N_{out} indicates the dimension of the output, which is 4 in $\mathcal{M}_{\text{R}}$, and 3 in the others.

B. Implementation Details

B.1. Network Architecture

Our network architecture, illustrated in Fig. B follows a consistent MLP structure across all components in Sec. 3.2 and 3.3, adopting a similar design to that proposed by [34]. Input is the concatenation of positional encoding of time φ_t and positions φ_x, and each linear layer, except for the final layer, outputs a 256-dimensional feature vector. The network $\mathcal{M}_{\text{R}}$ yields outputs in $\mathbb{R}^{N\times 4}$, while other networks produce output vectors in $\mathbb{R}^{N\times 3}$. $\mathcal{M}_{\text{R}}$ outputs quaternions for each stroke's rotation, which are subsequently converted to rotation matrices for stroke deformation.

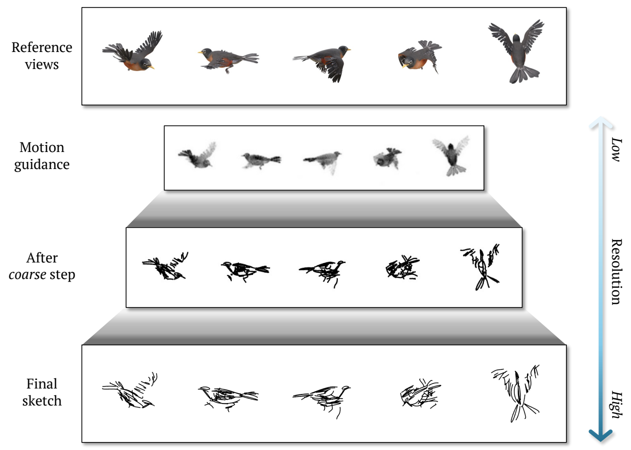


Figure C. Different resolution through processing stages. Our method gradually increases resolution according to stages to compute the location and deformation of strokes.

B.2. Optimization Details

The learning parameters differ between reconstructing synthetic datasets and real-world scenes. For synthetic scenes, we set the frequency value $L = 10$ for both temporal and spatial positional encoding. For real scenes presented in Fig. 9, we use $L = 8$ for temporal and $L = 10$ for spatial encoding. Learning rate values are also slightly different in synthetic and real scenes. For the former, in drawing process as described in Sec. 3.3, we apply a learning rate of 5.0×10^{-4} to $\mathcal{M}_\text{T}$ and $\mathcal{M}_\text{R}$, and 1.0×10^{-3} to all other parameters. For the latter, during sketch reconstruction, we apply a learning rate of 5.0×10^{-4} to the canonical stroke positions, 1.0×10^{-4} to $\mathcal{M}_\text{T}$ and $\mathcal{M}_\text{R}$, and 2.5×10^{-4} to $\mathcal{M}_\text{L}$. We set the learning rates $lr_\text{pcd} = 1.0 \times 10^{-3}$ and $lr_\text{mlp} = 5.0 \times 10^{-4}$ to optimize the canonical point cloud (*i.e.*, the point cloud before network-based shifting) and the motion guidance function detailed in Sec. 3.2 for all scenes. In addition, during learning motion guidance, to embed core motion information into the network, we initialize a canonical point cloud at $t = 0$ and reset the network parameters in the middle of the process.

Meanwhile, our framework is structured to gradually increase resolution as the optimization process progresses. As shown in Fig. C, We initially obtain motion guidance at quarter resolution of the target image size. Then, in the coarse stage of sketch synthesis, we render strokes into a 50% of the full resolution. We finally get moving sketches by optimizing the full resolution of the target frame size. For instance, for synthetic scenes, we first learn guidance at 100×100 resolution and then optimize the per-stroke transformation using 200×200 frames. The final output produces sketch frames at 400×400 resolution.

	Per-frame Chamfer (↓)	Motion velocity distance ($\times 10^{-3}$) (↓)
4DGS[†]	0.286 ± 0.057	4.24 ± 2.71
Deformable 3DGS[†]	0.269 ± 0.071	4.01 ± 2.67
SC-GS[†]	0.289 ± 0.053	3.99 ± 2.65
Liv3Stroke (Ours)	0.252 ± 0.049	4.16 ± 2.34

Table A. Quantitative results of 3D motion guidance accuracy of [†]GS-based works with filtering based on the opacity value.

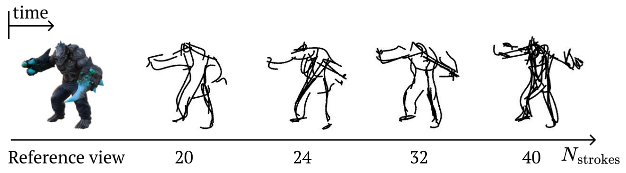


Figure D. The effects of using different numbers of strokes. When reconstructing a sketch video, we allow users to set N_strokes. More strokes produce detailed sketches, while fewer strokes yield abstract ones.

C. Additional Results

We provide results of all rendered synthetic scenes in Fig. F. Compared to other existing works, our framework can represent diverse movements and key features of the view-consistent structure directly from RGB video frames. We visualize guidance views at the full target image resolution for better clarity. We highly recommend finding videos in the supplementary material to see the whole movement of each scene.

C.1. Quantitative Results of the Motion Guidance

We present additional quantitative results of the motion guidance that we obtained from Sec. 3.2 and filtered results of GS-based dynamic reconstruction works [15, 44, 47] according to the opacity value with a threshold of $\alpha = 0.5$. Table A and Tab. 1 (b) of the main paper shows our method's capability to capture meaningful 3D motion information, although it does not pursue realistic reconstruction.

C.2. Results of Different Number of Strokes

We provide results to show the effect of the number of strokes as in Fig. D. Like [5] and [41], we can control abstraction levels of sketches by adjusting the number of curves. With a higher number of strokes, we can capture more detailed features, while fewer strokes result in more abstract representations.

C.3. Limited Multi-View Information

We provide results under limited multi-view information in varied conditions. From a frontal view, we captured frames along a circular trajectory around the object, collecting 100 frames over an angle $\theta(^\circ)$ while maintaining a constant dis-

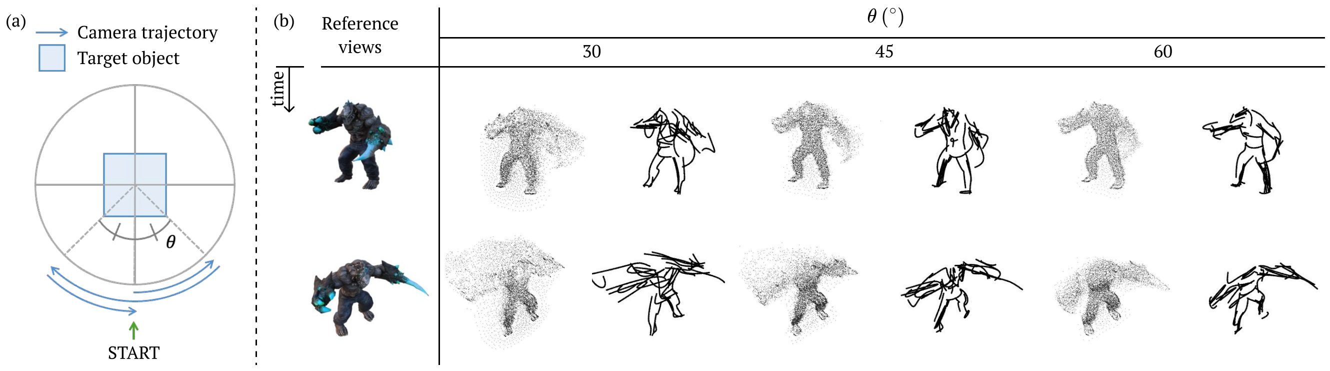


Figure E. Results of limited multi-view information. **(a)** Experimental setup for data collection. We followed a circular path around the object to capture frames, beginning from the frontal view (marked with a green arrow). **(b)** Results of the motion guidance and sketch under different angles. Our approach can achieve 3D motion sketch representation with limited yet sufficient viewpoint information ($\theta \geq 45$), even when the motion guidance exhibits noise, such as at $\theta = 45$.

Method	Novel views		Fixed views	
	Motion	Structure	Motion	Structure
CLIPasso	3.32	3.08	2.87	2.70
Sketch Video Syn.	2.66	2.64	3.93	3.77
Sugg. Contours	4.26	4.29	3.82	3.90
Liv3Stroke (Ours)	3.32	3.08	3.21	2.94

Table B. User study results. Note that "Motion" denotes the evaluation of how well the result describe the desired movement, and "Structure" is the score of how well it contains key features of the 3D structure.

tance from the center. The detailed experimental setup is visualized in Fig. E (a).

Our approach renders 3D sketches of the object in motion with sparse yet adequate viewpoint information, as illustrated in Fig. E (b). Additionally, we find that our method can roughly capture the 3D key structure of the object even when the motion guidance exhibits noise, such as at $\theta = 45$.

C.4. User study

We provide a questionnaire to evaluate the perceptual implication of generated sketches. Participants rated the sketches on a five-point scale (1-5), evaluating them from both novel camera viewpoints and the fixed perspective. The rating criteria were: (1) how effectively the sketch captures the motion and (2) how well it conveys the 3D structure of the target object.

Table B summarizes the answers of 44 participants. Overall, Suggestive Contours [6] achieves the highest scores across all metrics, which can be attributed to its direct contour extraction from 3D meshes, as illustrated in Fig. 6. Unlike other methods that rely on image-based processing, this approach results in higher evaluation scores. For novel views, LiveStroke performs comparably with CLIPasso [41]. While Sketch Video Synthesis [50] has a higher score in the fixed views, it struggles to effectively capture 3D geometric features and motion characteristics when evaluated from moving camera perspectives. LiveStroke exhibits only minimal performance decrease when transitioning from the novel perspectives to the fixed view, demonstrating consistent performance regardless of the viewing perspective. This stability distinguishes our approach from others, which shows significant performance variations between different viewpoints.

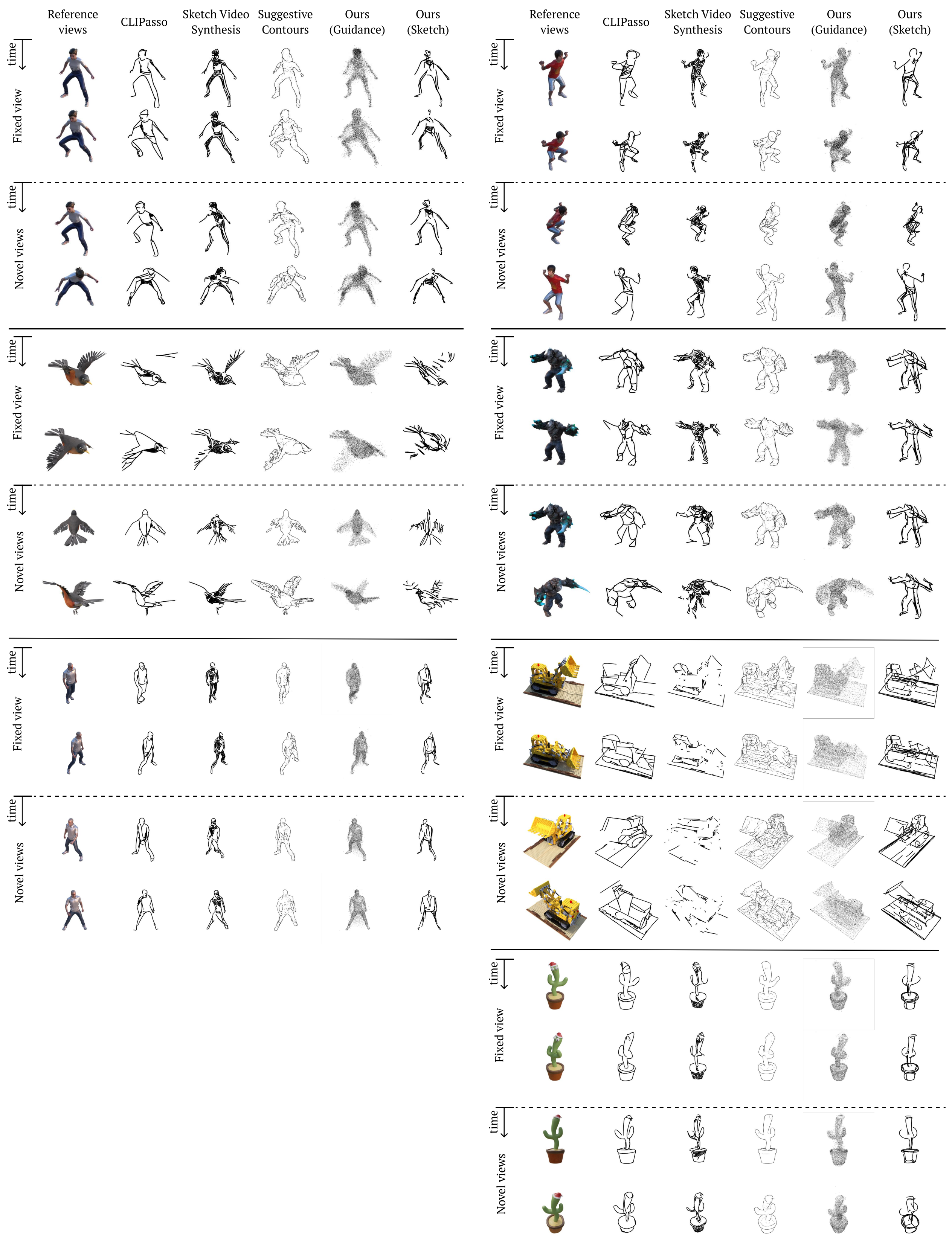


Figure F. Results of synthetic scenes. Our approach can represent diverse motions by using view-consistent deformable 3D strokes.