

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Controllable Animation of Images and Sketches Using Generative AI

Doctoral thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

Gaurav Rai (PhD20005)

gauravr@iiitd.ac.in

Department of Computer Science and Engineering
Indraprastha Institute of Information Technology Delhi, India

Supervisor:
Prof. Ojaswa Sharma

October 30, 2026

Certificate

This is to certify that the thesis titled **Controllable Animation of Images and Sketches Using Generative AI**, submitted by **Gaurav Rai** (PhD20005), to the Department of Computer Science, Indraprastha Institute of Information Technology Delhi (IIIT-Delhi), for the award of the degree of **Doctor of Philosophy**, is a bonafide record of the research work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.



Prof. Ojaswa Sharma
Associate Professor
Department of Computer Science and Engineering
IIIT-Delhi, India, 110020

Declaration

I acknowledge that I am fully responsible for the entire content of my thesis, including any sections assisted by online tools, including Artificial Intelligence-based tools. I accept full accountability for any violations of ethical standards in publications arising from the use of such tools.



October, 2026
Gaurav Rai
PhD20005
IIIT Delhi, India

Acknowledgement

I would like to express my deepest gratitude to my PhD supervisor, Prof. Ojaswa Sharma, for his invaluable guidance, constant encouragement, and unwavering support throughout my doctoral studies. His intellectual rigor, insightful feedback, and academic vision have been instrumental in shaping this dissertation and in guiding my development as a researcher. His ability to identify meaningful research directions, provide constructive criticism, and encourage independent thinking has been instrumental at every stage of this journey. I am especially grateful for his patience, trust, and confidence in my abilities, which motivated me to persevere through the many challenges that accompany doctoral research. I feel truly fortunate to have had the opportunity to work under his supervision and to learn from his dedication to academic excellence.

I am grateful to the members of my PhD monitoring committee, Prof. A. V. Subramanyam and Dr. Jainendra Shukla, for their valuable time, insightful comments, and constructive suggestions. Their careful evaluation of my work and their encouragement during various stages of my PhD have helped me refine the direction, presentation, and overall quality of this dissertation. Their observations often prompted me to think more critically and to improve my work with greater rigor and clarity. I sincerely appreciate their support and their contribution to my academic development.

I gratefully acknowledge IIIT-Delhi for providing an excellent academic environment, as well as the facilities and infrastructure necessary for carrying out this work. I also sincerely thank the administrative staff of the institute, especially Mr. Raju Biswas and Mr. Imran Khan, for their timely assistance with academic and administrative matters. My appreciation also extends to the IT staff for their prompt and continued technical support whenever required. I am thankful to TiH for awarding the Doctoral Fellowship and to CDNM for supporting the research projects associated with this dissertation.

I would also like to acknowledge my collaborators and co-authors, whose valuable insights, discussions, and contributions have significantly enriched my research. Their intellectual engagement and collegial support have played an important role in shaping several aspects of this dissertation.

I am immensely grateful to my colleagues, lab mates, and seniors for fostering a collaborative, supportive, and enjoyable work environment. The discussions, brainstorming sessions, shared experiences, and mutual encouragement within this community made the PhD journey far more enriching and enjoyable. I am especially grateful to my seniors, Aradhya Neeraj Mathur, Mohit Sharma, and

Gunjan Singh, for their guidance, encouragement, and willingness to share their experience and advice whenever needed. Their mentorship and support were especially valuable during important phases of my academic journey. I would also like to offer a special note of thanks to Dr. Mansi Goel for her academic and intellectual support, particularly during challenging moments of this work. Her thoughtful discussions, encouragement, and perspective were deeply appreciated and helpful in navigating difficult phases of the research process. I am thankful to my colleagues and friends Aarya Patel, Mayuri, Pankaj Kaushik, Himanshu Singh, Megha, and Anshul Goswami. Their companionship, motivation, and goodwill made this long and demanding journey more meaningful and memorable.

This milestone would not have been possible without the contributions and encouragement of all those who have been part of this journey, directly or indirectly. This acknowledgment is a small token of my immense gratitude towards everyone who has supported me along the way.

Gaurav Rai

Publications

1. Rai, G., & Sharma, O. (2025). LiveImage: Motion Condition Guided Diffusion Model for Video Motion Transfer. In Proceedings of the IEEE International Conference on Multimedia & Expo (ICME 2025). Chapter 4

2. Rai, G., Gupta, S., & Sharma, O. (2024). SketchAnim: Real-time sketch animation transfer from videos. In Computer Graphics Forum, e15176. Chapter 5

3. Rai, G., & Sharma, O. (2025). Enhancing Sketch Animation: Text-to-Video Diffusion Models with Temporal Consistency and Rigidity Constraints. In Proceedings of the International Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025). Chapter 6

4. Goswami, P., Rai, G.*, Akula, V. S. A.*, Sharma, O., & Imran, M. (2025). Space-aware indoor furniture recommendation and placement using agentic AI. In Proceedings of the IEEE International Conference on Intelligent Reality (ICIR 2025).

5. Rai, G., Salhotra, A., & Sharma, O. (2026). Generative AI for video-driven image animation: A survey. Computer & Graphics.

6. Mittal, T.*, Rai, G.*, & Sharma, O. (2026). TexSketch: Bringing Texture-Aware Colorization to Sketches. European Conference on Computer Vision (ECCV AI4VA 2026).

Under Review

7. Rai, G., & Sharma, O. (2026). Sketch2Motion: Text-driven 2D sketch to 3D animation via diffusion-guided skeleton optimization. Chapter 7

8. Rai, G., & Sharma, O. (2026). A survey on sketch animation methods. Chapter 2

Abbreviations

AED : Average Euclidean Distance

AI : Artificial Intelligence

AKD : Average Keypoint Distance

ARAP : As-rigid-as-possible

BFS : Breadth First Search

BBW : Bounded Biharmonic Weight

CDT : Constrained Delaunay Triangulation

CDMM : Single Domain Motion Model

ESA : Enhancing Sketch Animation

FID : Frechet Inception distance

GAN : Generative Adversarial Network

LA : Length-area

LBS : Linear Blend Skinning

LLMs : Large Language Models

MLLMs : Multimodal Large Language Models

MCN : Motion Condition Network

MLP : Multi-Layer Perceptron

MSE : Mean Squared Error

MVC : Mean Value Coordinates

SAM : Segment Anything Model

SDMM : Single Domain Motion Model

SDS : Score Distillation Sampling

T2V : Text-to-video

Abstract

Animation is a useful medium for communicating ideas, processes, and emotions, with applications in entertainment, education, and scientific visualization. However, creating it remains largely limited to non-experts, because it requires artistic skill, technical expertise, and substantial manual effort. Static images and hand-drawn sketches, by contrast, are among the most accessible forms of visual expression. This thesis investigates how static images and hand-drawn sketches can be transformed into controllable animations with reduced manual effort.

Images and sketches differ fundamentally in their visual representations and available motion cues. Images are dense raster signals rich in texture and color. Motion can therefore be inferred from appearance-based correspondences such as optical flow, keypoints, and warping fields, and fidelity is measured photometrically against ground-truth video. Sketches are sparse in nature and lack these cues. Their motion must be inferred from geometric representations such as skeletons, meshes, or strokes. Their fidelity means preserving topology, stroke quality, and proportions, and there are no ground-truth animations to measure against. As a result, image animation techniques do not transfer directly to sketches. This thesis bridges two traditions. Generative video models offer rich motion semantics but lack structural guarantees and may hallucinate content, whereas classical animation pipelines provide precise control but require substantial artistic and technical expertise. This thesis fills this gap by combining generative motion synthesis with explicit motion representations to achieve controllable, structure-preserving animation.

The thesis first studies video-driven image animation using learned motion representations and diffusion models, showing that their reliance on dense appearance cues limits their applicability to sparse sketches. It then introduces video-driven sketch animation based on skeletal motion, preserving structural topology without requiring paired training data. Next, it overcomes the need for a reference video by enabling text-driven motion control. Finally, it addresses depth, viewpoint, and self-occlusion by lifting sketches into textured, rigged 3D characters and animating them through diffusion-guided skeletal optimization with physics-inspired constraints. Together, these works progress from appearance-driven to geometry-driven animation, from reference-dependent to semantically controlled motion, and from planar deformation to physically grounded 3D animation, while addressing challenges including large pose variations, temporal consistency, shape fidelity, occlusion, and fine-grained controllability.

Table of Contents

Page

1 Introduction **1**

1.1 Motivation . . . 1

1.1.1 The authoring bottleneck . . . 2

1.1.2 Sketches as the foundation for animation authoring . . . 2

1.1.3 Image animation vs. sketch animation . . . 3

1.2 Research aim, objectives, and challenges . . . 5

1.2.1 Research aim . . . 5

1.2.2 Research challenges . . . 5

1.2.3 Research objectives . . . 6

1.3 Contribution and structure of the thesis . . . 7

2 A brief history of sketch animation **11**

2.1 Sketch representation . . . 12

2.2 History of animation techniques . . . 13

2.3 Applications . . . 15

2.4 Sketch animation techniques . . . 16

2.4.1 Inbetweening approaches for sketch animation . . . 16

2.4.2 Physics-based techniques . . . 21

2.4.3 Data-driven techniques . . . 22

2.4.4 2D sketch to 3D animation . . . 27

2.4.5 Sketch animation interfaces . . . 30

2.4.6 Comparative analysis of animation techniques . . . 33

2.4.7 Datasets and evaluation metrics . . . 37

2.4.8 Recent trends . . . 42

3 Literature review **45**
3.1 Taxonomy of video-driven animation systems 45
3.2 Generative priors for animation 47
3.3 Video-driven image animation 49
3.4 Video-driven sketch animation 50
3.5 Text-driven sketch animation 51
3.6 Text-driven 3D animation 53
3.7 Summary and research gaps 54

4 LiveImage: Motion Condition Guided Diffusion Model for Video Motion Transfer **57**
4.1 Introduction 57
4.2 Preliminaries 58
4.2.1 Keypoint detector 58
4.2.2 Dense motion network 59
4.2.3 Thin-plate spline motion model (TPSMM) 60
4.3 Methodology 61
4.3.1 Keypoint regularization 62
4.3.2 Motion condition network 63
4.4 Experiments 66
4.4.1 Benchmarks 66
4.4.2 Training 66
4.4.3 Results and comparison 68
4.4.4 Ablation Study 70
4.5 Limitations 71
4.6 Conclusion 71

5 SketchAnim: Real-time sketch animation transfer from videos **73**
5.1 Introduction 73
5.2 Methodology 74
5.2.1 Skeleton mapping 75
5.2.2 Motion extraction and motion transfer 77
5.2.3 Skeleton tree node hierarchy 77

5.2.4 Initial pose matching . . . 77
5.2.5 Motion mapping . . . 79
5.2.6 Resolving self-occlusions . . . 79
5.3 Results . . . 80
5.3.1 Evaluation matrices . . . 82
5.4 Comparisons with the state-of-the-art . . . 83
5.4.1 Qualitative evaluation . . . 83
5.4.2 Quantitative evaluation . . . 83
5.5 Limitations and future work . . . 84
5.6 Conclusions . . . 87

6 Enhancing Sketch Animation: Text-to-Video Diffusion Models with Temporal Consistency and Rigidity Constraints 89

6.1 Introduction . . . 89
6.2 Preliminaries . . . 90
6.3 Methodology . . . 92
6.3.1 Length-Area minimization . . . 93
6.3.2 Shape preservation . . . 94
6.4 Experiments and results . . . 96
6.4.1 Implementation details . . . 96
6.4.2 Results and comparison . . . 96
6.4.3 Ablation study . . . 98
6.5 Limitations . . . 99
6.6 Conclusion . . . 100

7 Sketch2Motion: Text-driven 2D Sketch to 3D Animation via Diffusion-guided Skeleton Optimization 101

7.1 Introduction . . . 101
7.2 Methodology . . . 103
7.2.1 Overview . . . 103
7.2.2 Skeleton representation . . . 104
7.2.3 Motion initialization . . . 105
7.2.4 Forward kinematics and skinning . . . 110

7.2.5 Diffusion-based motion-aware score distillation guidance . 111
7.2.6 Physics-inspired motion regularization 112
7.2.7 Secondary motion effects 116
7.3 Experiment and results 117
7.3.1 Implementation details 117
7.3.2 Dataset 119
7.3.3 Evaluation matrices 119
7.3.4 Comparison 122
7.3.5 Ablation study 124
7.3.6 Limitation and future work 126
7.4 Conclusion 127

8 Discussion **131**
8.1 Contribution of the thesis 131
8.2 Limitations 132
8.3 Future direction 133

9 Conclusion **135**
9.1 Summary of the thesis 135
9.2 Objectives revisited 136
9.3 Concluding remarks 137
*

List of Tables

Table		Page
1.1	Comparison between image and sketch representations.	4
1.2	Overview of modalities, representations, motion sources, and motion models used across thesis chapters.	9
2.1	Comparison of prominent sketch-based animation methods.	17
2.2	Comparative analysis of sketch animation techniques with attribute-supported citations	33
2.3	List of sketch datasets.	38
3.1	Mapping of the literature review to their corresponding motion sources, motion models, and source representations.	45
3.2	Research gaps identified in this chapter and the corresponding contributions of this thesis.	55
4.1	Training configuration.	68
4.2	Quantitative comparison of the proposed approach with state-of-the-art methods.	70
4.3	Ablation results—the quantitative evaluation on w/o motion condition, w/o keypoint regularization, and the proposed method.	71
5.1	Quantitative comparison with state-of-the-art methods.	84
6.1	Comparison with state-of-the-art methods.	95
6.2	Ablation results—the quantitative evaluation on w/o LA minimization, w/o shape preserving ARAP, and the proposed method.	99
7.1	Analysis of text-to-3D model animation techniques.	118
7.2	Hyperparameter values used in the experiments.	120
7.3	Quantitative comparison and user-evaluation with state-of-the-art methods.	121

7.4 Quantitative ablation of the proposed approach, evaluating the impact w/o motion initialization, w/o physics-based regularization, and w/o secondary motion, compare to the full method. . . 126

List of Figures

Figure **Page**

1.1 Overview of the four research projects, their corresponding objectives, and the progression between them. The highlighted blocks between successive chapters indicate the key limitation identified by the preceding project that motivates the next. Collectively, the thesis progresses from appearance-driven image animation to geometry-aware sketch animation, from video-driven motion transfer to text-guided semantic control, and finally from 2D deformation to physically grounded 3D animation. 8

2.1 Raster and vector representations of sketches [142, 143]. 14

2.2 A hybrid deformation framework that supports real-time editing of either the skeleton in the deformed pose or the deformation cage in both rest and deformed configurations [29]. All modifications are automatically propagated, ensuring consistent synchronization across all deformation components. 20

2.3 Rough line-art interpolation [23]. It takes rough vector key drawings (light gray), generates stylistically consistent intermediate strokes (black), and uses an interactive stroke distribution synthesis algorithm that minimizes temporal artifacts. 21

2.4 Two-stage occlusion resolution in stroke-based drawings [86]. (a) Original line drawing composed of multiple strokes with occlusions. (b) Occlusion representation using labeled boundary strokes, where half-arrows indicate relative depth ordering. (c) Occlusion resolution through identification of regions corresponding to occluding surfaces. 21

2.5 Skeleton mapping approach of Animated Drawings method [164]. (a) From the predicted joint keypoints (b) are used to construct a skeletal rig that drives the character animation. (c) The original motion capture pose is taken, with torso and upper-limb joints projected onto a frontal plane and lower-limb joints onto a sagittal plane. (d) The global orientations of the bones are then computed within their respective planes, and the character's joints are rotated accordingly, resulting in the retargeted pose. 24

2.6 Example of a walking cycle generated with Monster Mash [39]. (a) A hand-drawn sketch (b) is first converted into an inflated 3D model with editable control points. (c) The final walking animation is created by defining and recording the motion trajectories of these control points. 28

2.7 Illustration of the 3D skeleton-driven sketch animation pipeline proposed by Zhou et al. [235]. A single input drawing (left) is combined with a target 3D motion sequence (middle) to generate the resulting animated character (right). 29

2.8 A real-time animation system that transforms a single childlike drawing into a 2.5D character viewable from multiple views [163]. It supports motion retargeting using any 3D skeletal motion, making it essential for mixed-reality applications. 30

4.1 Overview of the proposed motion model for image animation. The proposed motion model (LiveImage) has two modules: SDMM and MCN. SDMM detects keypoints, estimates dense optical flow with keypoint regularization, and generates coarse images via inpainting. MCN refines motion using keypoint differences based on weight amplitude, and the diffusion model synthesizes coarse-to-fine frames. 62

4.2 During inference, fixed-size patches from the source image and Gaussian-blurred frames are extracted from motion keypoints. MSE loss aligns appearance features for cross-identity motion transfer. 64

4.3 LiveImage results on VoxCeleb [131], TaiChiHD [156] Fashion[224] and MGif[156] dataset. 67

4.4 Comparison of LiveImage with state-of-the-art methods on VoxCeleb, TaiChiHD, Fashion and MGif datasets. 69

4.5 The qualitative results on different settings and generated results w/o motion condition, w/o keypoint regularization, and with the LiveImage method. 71

4.6 LiveImage struggles with self-occlusion, false correspondence, and cross-domain (sketch) motion transfer during animation. 72

5.1 Sketch animation using motion from exemplar video. 73

5.2 Overview of the SketchAnim framework for sketch-based animation. The method involves taking a sketch and a video as input. The user then draws a skeleton on the initial frame of the video. Mapping of this skeleton to the input sketch is done by shape matching with MVC. Thereafter, the transformation of skeletal bones is estimated by tracking the skeletal joints across the video frames. Finally, the sketch is animated using linear blend skinning with BBW weights. 75

5.3 Video skeleton mapping to sketch via mean-value coordinates. . . 77

5.4 Estimation of the root node as the deepest vertex. The blue color number denotes the smallest sum from the leaf nodes and the corresponding node as the root. 78

5.5 Pose matching of the sketch skeleton to the video skeleton. The skeleton is traversed in a BFS manner starting from the root node (shown in red). At a current node, transformations are applied to edges in its 1-ring to match the orientation of the corresponding source edges. 78

5.6 SketchAnim results on biped and quadruped sketches. 81

5.7 SketchAnim results on sketches of inanimate objects. 82

5.8 Comparison of biped samples with the state-of-the-art methods. 85

5.9 Comparison of quadruped animation with the state-of-the-art methods. 85

5.10 Failure cases such as (a) frontal motion, (b) self-occlusion in rest pose, and (c) error propagation in shape mapping. 86

6.1 Problem with the LiveSketch [53] method. This example illustrates the lack of temporal consistency and the shape distortions that occur during motion. 89

6.2 Network architecture of the proposed framework. The framework takes an initial set of control points, $\mathbf{Z}^{\text{init}}$, as input and predicts the corresponding displacement field, $\Delta\mathbf{Z}$, for the local ($\mathcal{M}_l$) and global ($\mathcal{M}_g$) paths under the guidance of a frozen text-to-video (T2V) diffusion model. The Score Distillation Sampling (SDS) loss, $\mathcal{L}_{\text{SDS}}$, aligns the predicted motion with the input text prompt by backpropagating gradients through the differentiable rasterizer R. For each frame, a Length-Area loss ($\mathcal{L}_{LA}$) is used to maintain temporal consistency and avoid drastic shape changes. Further, ARAP loss ($\mathcal{L}_{arap}$) maintains the sketch stroke's rigidity and prevents shape distortions during motion. 92

6.3 Cubic Bézier curve of each stroke and their corresponding control points. Delaunay triangulation of Bézier control points. 94
6.4 Qualitative results of the proposed method and generated animated sketch sequences from input text prompts. 96
6.5 Comparison with state-of-the-art methods (VideoCrafter1 [22] and LiveSketch [53]). In the figure above, the base of the wine glass is distorted in the previous methods, and in the surfer base, the original shape is missing compared to the proposed method (ESA). The squirrel's tail and body shape preserve the original topology in the proposed method. 97
6.6 Ablation study on different setting such as w/o LA minimization, w/o shape preserving ARAP, and the full method. 98
6.7 Failure cases. 100

7.1 Overview of the proposed method for 2D sketch to 3D animation. 101
7.2 Overview of pipeline architecture of the proposed method for 2D sketch to 3D model animation. 103
7.3 Qualitative results of the proposed method and generated 3D animation sequences from input text prompts. 118
7.4 Qualitative Comparison with state-of-the-art methods AnimateAnyMesh [206] and BiMotion [189]). 119
7.5 Visual results of ablation study on different settings such as w/o motion initialization, w/o physics regularization, w/o secondary motion, and the full method. 125
7.6 Visual comparison of T2V diffusion model results generated by ModelScope [188] and the proposed method. 129

Chapter 1

Introduction

1.1 Motivation

Computer animation has emerged as a transformative technology that bridges art and science to create dynamic visual content across diverse domains, including entertainment, education, healthcare, engineering, cultural heritage, virtual reality, and digital media. As one of the most effective media for communicating temporal, causal, and procedural information, animation enables static concepts to be presented as dynamic processes [183]. Consequently, the demand for animated content has grown rapidly beyond professional studios to include educators, designers, researchers, and everyday content creators. Despite its widespread applications, animation production remains largely confined to trained professionals. Conventional animation pipelines require artistic expertise, technical proficiency, specialized software, and substantial manual effort, making high-quality animation both time-consuming and expensive [87, 103]. Although recent advances in artificial intelligence have significantly reduced manual intervention by enabling faster and more realistic motion generation, most existing methods are designed for photorealistic images with rich textures and well-defined appearance cues [156, 229]. However, they still struggle to handle large pose variations during motion [177]. Extending these approaches to hand-drawn sketches is considerably more challenging. Unlike natural images, sketches are sparse, abstract representations that primarily encode structural contours while omitting color, shading, and fine-grained texture [40]. As a result, appearance-based motion correspondence and identity preservation become unreliable. Table 1.1 summarizes the fundamental differences between image-based and sketch-based representations. Consequently, techniques developed for image animation cannot be directly applied to sketches without dedicated motion representations and explicit structural constraints. Recent advancements in sketch animations lack generalizability [227, 236] and are limited to a specific kind of motion [164]. Addressing this limitation is the primary motivation of this thesis, which aims to develop robust, controllable, and structure-aware animation frameworks that make image and sketch animation more accessible to both experts and novice users.

1.1.1 The authoring bottleneck

Despite significant advances in animation technology, animation authoring remains labor-intensive and faces limited generalization. The authoring bottlenecks are as follows:

Manual content creation. Traditional animation is created frame by frame, typically requiring 12-24 drawings per second of motion, with consistent proportions, style, and timing [19, 87]. Although inbetweening [43] and interpolation techniques [23, 240] reduce manual effort, producing high-quality animation remains time-consuming.

Motion expertise. Creating convincing animation requires not only drawing skills but also a good understanding of animation principles such as timing, anticipation, spacing, squash and stretch, and follow-through [94, 104]. Acquiring these skills takes years of practice, and their absence often results in unnatural or mechanically rigid motion.

Technical complexity. Modern animation pipelines rely on rigging [6], skinning [29, 125], motion retargeting [55], and physics-based methods [21, 85, 200], all of which require specialized software and technical expertise. While procedural animation [91] and motion capture [101, 138] can simplify certain tasks, they introduce additional challenges, including parameter configuration, capture equipment, and motion retargeting.

Limited iteration. Revising an animation often requires redrawing frames or modifying complex rigs, making iterative design slow and costly [92, 196]. As a result, motion is typically refined only after the visual design is finalized, limiting rapid exploration in the early stages of content creation.

Together, these challenges constitute the primary authoring bottleneck addressed in this thesis. The proposed frameworks aim to reduce manual effort, simplify animation workflows, improve motion generation, and enable rapid, controllable animation from visual inputs such as images and sketches.

1.1.2 Sketches as the foundation for animation authoring

Photographs and hand-drawn sketches are simple and widely accessible forms for bringing entities to life. Photographs are universal and capture rich visual appearance, while sketches offer an intuitive means of expressing shape, structure, and visual intent. Sketching is also a fundamental medium for visual thinking

and communication across disciplines, including engineering, design, and education. These characteristics make static images and sketches natural inputs for animation authoring systems that allow easy creation, require minimal technical expertise, and allow users to communicate visual ideas directly. Consequently, enabling these familiar forms of visual expression to be transformed into expressive animations comprises a main motivation of this thesis.

Although sketches are inherently abstract, their abstraction emphasizes essential structural information, such as shape, form, and motion, while suppressing excessive visual details. This makes them particularly suitable for animation authoring, where structural consistency is often more important than photorealistic appearance. Sketches are widely used during early-stage design, including storyboarding, concept art, interface design, and technical illustration, where rapid visualization of motion is particularly beneficial. Unlike photographs, sketches are inherently easy to edit, allowing users to modify structure and proportions with minimal effort, making them well-suited for iterative animation authoring. Furthermore, sketch animation avoids many of the ethical concerns associated with animating photographs of real objects or persons, such as identity misuse and synthetic reenactment.

A common question is, why should sketch animation target novice users when drawing itself is a skilled activity? Creating a single sketch is a common practice across disciplines such as education, engineering, design, and everyday communication, but producing convincing animation requires additional expertise in motion principles, rigging, and animation software. Moreover, the value of a sketch depends on its purpose rather than its artistic quality. A classroom diagram, concept sketch, technical illustration, or a child's drawing can all benefit from animation without requiring professional-level artwork. In many applications, these sketches already exist, and the only remaining challenge is bringing them to life. Accordingly, this thesis is founded on the premise that the principal barrier is animation rather than drawing. By automating the animation process while preserving the original structure and style of the sketch, the proposed frameworks enable a much broader community of users to transform static sketches into expressive animations.

1.1.3 Image animation vs. sketch animation

Although image animation and sketch animation both aim to generate motion from a static visual input, they are fundamentally different problems. They differ in three key aspects: the underlying representation, the cues available for motion inference, and the criteria used to evaluate animation quality. As summarized in Table 1.1, images provide dense appearance information, including texture, color, and shading, whereas sketches contain only sparse structural information. Consequently, state-of-the-art image animation methods [155, 156,

159, 177, 178, 191, 197, 229] rely on dense appearance correspondences and are primarily designed for photorealistic imagery. These characteristics, however, do not extend to line drawings or textureless inputs, where rich appearance information is largely absent. As demonstrated experimentally in Chapter 4, this limitation motivates the transition from appearance-driven image animation to geometry-aware sketch animation.

Recent advances have shown that realistic animation can be generated directly in 2D using learned keypoints [128, 156, 229], deformation fields [159, 177], and correspondence maps [157, 191]. While effective for many applications, these methods cannot explicitly model depth, viewpoint changes, or severe self-occlusions, limiting their ability to generate complex motions. Extending sketch animation from 2D to 3D addresses these limitations by introducing explicit geometric and skeletal representations that enable depth reasoning, viewpoint variation, and physically plausible motion. Building upon prior 2D-to-3D sketch animation methods [39, 163, 236], the final framework proposed in this thesis further incorporates text-driven motion control and physics-based regularization to produce controllable and realistic 3D animations.

The techniques developed in this thesis are intended to benefit a diverse range of users, including educators, designers, artists, content creators, researchers, and amateur users. However, these groups face different challenges and require varying levels of animation fidelity. Most users require fast, plausible, and structurally faithful animations rather than production-quality results. In many applications, preserving the original drawing style is more important than achieving photorealism. These requirements motivate the development of the structure-preserving and controllable animation frameworks proposed in Chapters 5-7.

Table 1.1: Comparison between image and sketch representations.

Image	**Sketch**
Dense, pixel-level representation containing complete appearance information such as texture, shading, illumination, and color.	Abstract, sparse representation that captures structure, contours, and key geometry without color or texture.
Harder to edit; pixel manipulation requires advanced tools (segmentation, inpainting, editing networks).	Relatively easier to edit; users can quickly adjust shapes, exaggerate features, or simplify geometry.
Suitable for photorealistic/texturing tasks: rendering, face/body reenactment, appearance transfer, and realism-critical applications.	Suitable for design, illustration, CAD modeling, UI prototyping, sketch-to-image/video generation, and shape editing.

1.2 Research aim, objectives, and challenges

1.2.1 Research aim

The aim of this thesis is to develop controllable, structure-preserving animation frameworks that generate temporally coherent animations from a single static image or hand-drawn sketch, driven by either an exemplar video or text prompt, without requiring paired motion data, category-specific retraining, or animation expertise from the user. This aim embodies three key principles. **Controllable** implies that users can explicitly specify the desired motion rather than relying solely on unconstrained generative outputs. **Structure-preserving** ensures that the identity, topology, proportions, and stylistic characteristics of the input are maintained throughout the animation process. Finally, **Generalization**, eliminating the need for paired training data, per-category retraining, and expert intervention, makes the proposed frameworks broadly applicable, scalable, and accessible to non-expert users.

1.2.2 Research challenges

To achieve the research aim, this thesis addresses several key challenges. The main challenges are listed below.

C1: Motion inference under large pose variations. Appearance-driven animation methods often degrade under large pose changes, leading to shape distortion, structural artifacts, and temporal jitter. Effectively transferring motion across significant deformations remains a major challenge.

C2: Motion representation for sparse sketches. Unlike natural images, sketches lack texture, color, and other dense appearance cues that traditional correspondence-based methods require. Consequently, motion must be inferred from geometric and topological representations while establishing reliable correspondence with the driving signal.

C3: Structure and style preservation. For sketches, animation quality depends on preserving structural topology, stroke characteristics, geometric proportions, and artistic style rather than photometric realism. Maintaining these properties throughout deformation requires explicit structural constraints.

C4: Temporal coherence. Generating visually plausible individual frames does not guarantee coherent animation. Flickering, geometric drift, and inconsistent motion accumulate over time, making temporal consistency an essential optimization objective.

C5: Occlusion and the limitations of 2D representations. Planar representations cannot explicitly model depth ordering, viewpoint changes, or severe self-occlusions. Addressing these limitations requires either explicit depth reasoning or a transition to 3D geometric representations.

C6: Controllability and physical plausibility. Video-driven animation is constrained by the availability of suitable reference motions, while text-driven motion is inherently ambiguous and may produce physically implausible results. Effective animation, therefore, requires intuitive user control combined with geometric and physics-based constraints.

1.2.3 Research objectives

The research aim is pursued through the following objectives. These objectives are interconnected, with each one addressing limitations identified in the preceding one.

O1: Video-driven image animation. Develop a robust framework for transferring motion from a driving video to a static image while handling large pose variations and preserving structural and appearance consistency. (C1, C4)

O2: Controllable sketch animation. Design structure-preserving frameworks that animate hand-drawn sketches using either an exemplar video or a text prompt, while maintaining sketch topology, artistic style, temporal coherence, and robustness to self-occlusion. (C2, C3, C4, C5, C6)

O3: Sketch-to-3D animation. Generate rigged and physically plausible 3D animations from 2D sketches while addressing the depth, viewpoint, and occlusion limitations of 2D animation. (C5, C6, C3)

The objectives form a clear progression. Video-driven image animation explores the capabilities and limitations of appearance-based methods. It shows that their reliance on dense appearance cues limits their ability to handle sparse, non-photorealistic inputs such as sketches. This motivates a shift toward sketch animation based on geometric and skeletal representations. Video-driven sketch animation enables effective motion transfer but remains dependent on suitable driving videos. This limits user control and the diversity of achievable motions. This limitation is addressed in controllable sketch animation by introducing natural-language prompts as an alternative source of motion. However, 2D representations remain limited in modeling depth ordering, viewpoint changes, and severe self-occlusions. Addressing these limitations requires a transition to 3D representations, motivating 3D sketch animation.

1.3 Contribution and structure of the thesis

This thesis presents comprehensive techniques for both image and sketch animation, addressing their representations, motion sources, and underlying motion models. Figure 1.1 provides an overview of the thesis, illustrating the four research projects, the objective addressed by each, the limitations that motivate the subsequent project, and their principal contributions. The proposed methods introduce novel frameworks for motion transfer and synthesis across different modalities while addressing key challenges in sketch animation, including self-occlusion, generalization, and automation. Table 1.2 summarizes the key characteristics of the proposed and reviewed methods. It describes the input modality (image or sketch), representation (raster, vector, or mesh), motion source (video, text, or skeletal motion), and motion model, including keypoint-based dense flow, video-based skeletal motion, and text-to-video diffusion priors. This provides a unified view of the different animation paradigms considered in the thesis and highlights how motion is transferred and synthesized across image and sketch animation.

Creating realistic character animations manually is labor-intensive and technically challenging, requiring expertise to handle complex motions while preserving structural integrity. Video-to-image motion retargeting addresses this challenge by transferring motion from a driving video to a static image to generate animations. However, existing methods often struggle with large pose variations, resulting in artifacts and temporal inconsistencies. To address these limitations, the approach proposed in Chapter 4 combines keypoint regularization within a single-domain motion model to capture coarse motion, with diffusion-based refinement to correct distortions, enabling the generation of high-quality and coherent animations.

This thesis introduces several approaches to make sketch animation easier and more reliable. First, Chapter 5 introduces a method that transfers motion from a real video to a sketch by extracting skeletal motion and mapping it onto the sketch, enabling natural deformation. The approach uses shape-matching-based skeleton mapping for motion transfer and incorporates depth ordering to handle occlusions.

Chapter 6 proposes a skeleton-free approach that allows users to animate sketches using text descriptions, with additional constraints to preserve shape and ensure temporal consistency. This method uses text-to-video diffusion networks and aligns the sketch's motion using the Score Distillation Sampling (SDS) loss [139]. To facilitate the temporal consistency and shape preservation, length-area (LA) minimization and As-Rigid-As-Possible (ARAP) [82] loss have been proposed, respectively.

Animating 2D hand-drawn sketches in 3D space addresses the limitations of purely 2D methods, particularly in handling complex motions and depth cues.

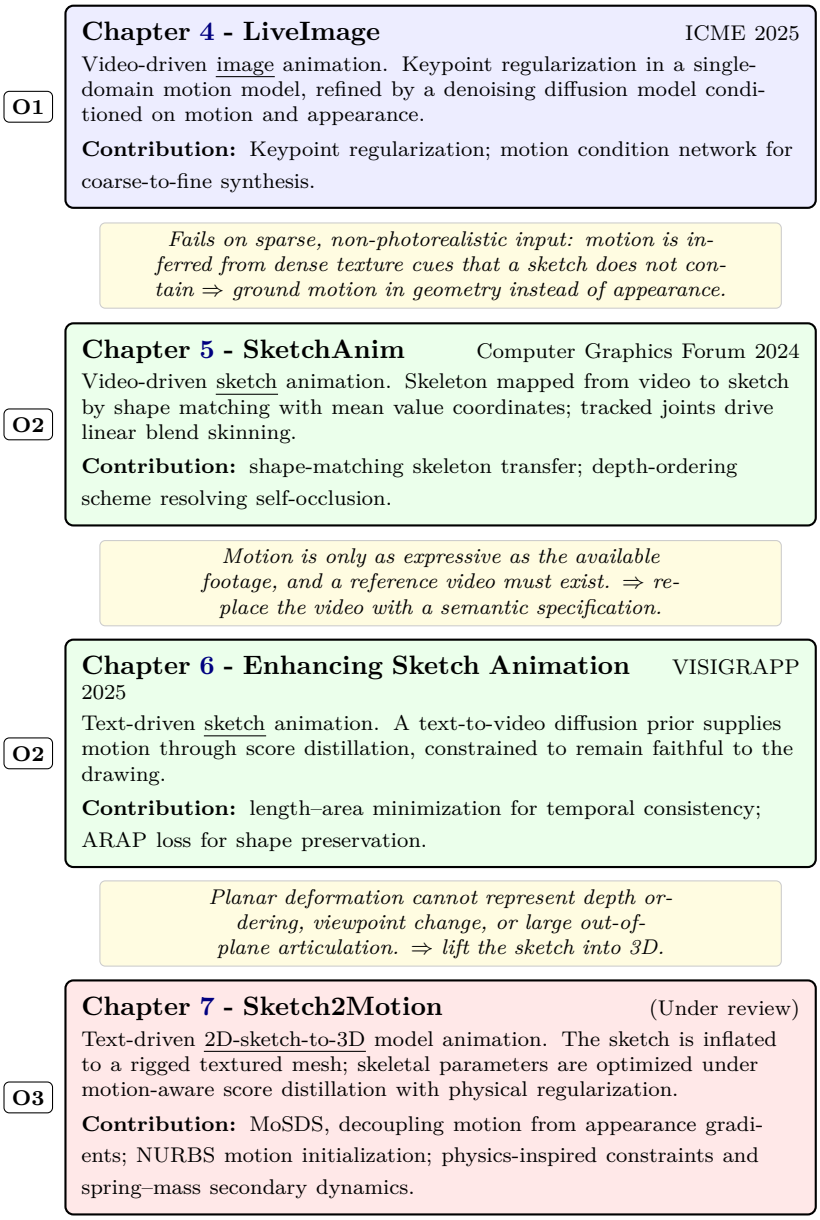


Figure 1.1: Overview of the four research projects, their corresponding objectives, and the progression between them. The highlighted blocks between successive chapters indicate the key limitation identified by the preceding project that motivates the next. Collectively, the thesis progresses from appearance-driven image animation to geometry-aware sketch animation, from video-driven motion transfer to text-guided semantic control, and finally from 2D deformation to physically grounded 3D animation.

Table 1.2: Overview of modalities, representations, motion sources, and motion models used across thesis chapters.

Chapter	**Modality**	**Representation**		**Motion source**	**Motion model**
		Input	**Intermediate**		
Chapter 4	Image	Raster	Raster	Video	Keypoint-based
Chapter 5	Sketch	Raster	2D Mesh	Video	Video skeletal motion
Chapter 6	Sketch	Vector	Vector	Text	T2V diffusion model
Chapter 7	Sketch	Raster	3D Mesh	Text	T2V diffusion model

In this direction, Chapter 7 introduces a 3D model animation method by inflating 2D sketches into 3D models, allowing animators to apply natural motion dynamics while preserving structural relationships. Using a text-to-video diffusion model as motion guidance with physics-based regularization. In this approach, motion is represented using explicit skeletal transformations, which are propagated to 3D meshes via forward kinematics and linear blend skinning. A text-to-video diffusion model provides semantic guidance via SDS, while physics-inspired smoothness, topology, and contact constraints ensure stable, realistic motion. This approach addresses challenges such as occlusion handling, realistic motion transfer, and shape preservation, enabling expressive, temporally coherent 3D sketch animations that bridge traditional 2D art with modern animation technologies.

Thesis statement

The central premise of this thesis is that controllable and structure-preserving generative animation does not necessarily require a larger generative model, but a more explicit motion representation that constrains what the model can change.

Chapter 2

A brief history of sketch animation

A sketch is a quick, freehand drawing used to visually express ideas, objects, or scenes. A sketch may contain flat regional colors, but it lacks the dense photometric information found in natural images, such as continuous shading, illumination gradients, cast shadows, and texture. Sketch animation plays a vital role in the creative process and technological innovation by enabling quick, expressive visualization of ideas and by offering a way to create dynamic visual storytelling frameworks. It is widely used in movie production, education, and interactive design, allowing artists, designers, and animators to explore motion, timing, and scene composition without the complexity of detailed rendering. This makes it an essential tool for storyboarding, prototyping, and early concept development across domains such as entertainment video, game design, educational illustration, and user experience. Over the years, different sketch animation techniques have emerged, each having its advantages and challenges.

Traditional hand-drawn sketch illustrations are animated frame-by-frame [87]. As computational techniques and interactive technologies advance, these methods evolve to support more diverse and expressive forms. Animation frameworks with procedural animation techniques [91, 92, 199] have seen significant advances over time. Many such approaches animate sketches using skinning methods with various control handles such as points, skeletons, and cages. To further automate the process of animation, motion capture (MoCap) technologies [97, 101] have been explored to generate sketch animations by mapping captured motion data from body sensors to sketches. Physics-based simulation methods [2, 47] produce realistic and natural motion effects in sketch animation. Furthermore, sketch interpolation methods [8, 86, 132, 154, 162, 219, 240] reduce manual effort and overcome data dependency.

In recent years, advancements in deep learning and generative AI have significantly transformed the field of sketch animation, making animation generation more accessible to novice users. Furthermore, these techniques [171, 233] animate sketches using generative models such as diffusion models [12, 73, 167],

guided by either video-based motion or text-based motion descriptions. These approaches allow users to customize motion according to their requirements, thereby expanding creative possibilities in sketch animation while enhancing the realism, fluidity, and responsiveness of the resulting animated sequences. Despite these advancements, there are significant challenges in the field of sketch animation, such as computation complexity, trade-offs between quality, user control, accessibility, multi-user systems, and real-time interactive applications. Additionally, user-friendly sketch animation interfaces overcome the complexity for both novices and experts, enabling access to sketch animation technologies. Various sketch animation techniques, such as keyframe-based animation, autocomplete/interpolation, data-driven methods like motion capture, video motion retargeting, and text-driven animation, have continuously evolved with technological advancements to create more immersive and natural sketch animations. This chapter comprehensively maps the evolving landscape of sketch animation by categorizing and critically evaluating key methodologies, highlighting technological innovations, and discussing the inherent challenges and future opportunities.

2.1 Sketch representation

Sketches are simplified and abstract representations of objects, characters, or scenes, depicted by line drawing, contours, raster sketch, and strokes that effectively convey essential structure and story [116, 151]. Sketches are inherently challenging to process due to their abstract and incomplete visual characteristics, such as the lack of texture, ambiguous or overlapping strokes, and discontinuous or broken contour lines, which make it difficult to reliably infer object boundaries and structural details. Unlike detailed illustrations, sketches focus on capturing proportions, movement, and the intent of the drawing over exact realism. It allows the flexibility to the artists or users to experiment with ideas, modify poses or perspectives, and convey actions or a story efficiently. Their flexible nature makes them highly suitable for creative pipelines that prioritize speed and adaptability. Sketch representation is critical for interpreting and generating visual content or animation with minimal inference. Different sketch representations capture different aspects of visual information, offering various advantages and posing distinct challenges. The key representations of the sketches are vector and raster, as described below.

Raster sketches A raster sketch uses a pixel-based representation with the level of detail determined by the image resolution. It can contain texture information bounded by the silhouette, reducing complex shapes or layouts into a simplified grid-based structure as shown in Figure 2.1(a). Their pixel-aligned format makes them suitable for convolutional neural networks [106, 164] and image processing tasks, leading to significant performance improvements across

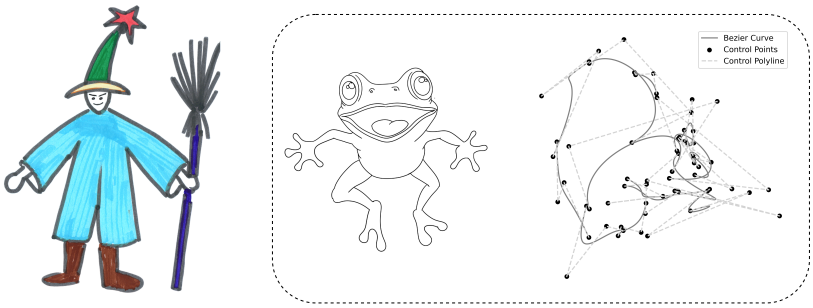
several tasks, including sketch classification, sketch-based image retrieval, and sketch animation. The disadvantage of this representation is that a coarse resolution results in inaccurate or overly simplistic object shapes and placements. Mapping continuous shapes onto a discrete grid often introduces distortions or unwanted artifacts.

Vector sketches Vector sketches represent drawings as parameterized stroke sequences, such as polylines [107], Bézier curves [52] (see Figure 2.1(b)), or splines [5, 89]. Unlike raster sketches, vector sketches store drawing primitives in a resolution-independent format. The advantage of resolution independence is that it allows easy scaling and transformation without losing sketch quality. It is more suitable for modeling and editing fine-grained stroke properties and for neural network-based generation. The major challenge with vector sketches is that they are complex and cannot be generated accurately, especially from raster sketches. Handling artistic or highly abstract sketches where stroke order and structure vary widely is difficult. Vector sketches represent objects and scenes through continuous or discrete strokes. It is a simple, intuitive, and compact representation, suitable for many applications such as, retrieval, recognition, and generation. It is effective at capturing essential structural and shape information. Vector drawings can be ambiguous due to a lack of texture and fine details, making it challenging to distinguish between visually similar classes based on their outlines. In similar lines, temporal sketches make an add-on in vector sketches by incorporating the temporal sequence of strokes, which is how a drawing evolves. Each stroke has associated timestamps or sequential order, containing the artist's process. The advantage of temporal sketches is that they capture rich, dynamic information about the drawing process. It enables models to learn what was drawn and how it was drawn. The drawback is that temporal information can be noisy and requires large-scale, temporally annotated datasets.

In summary, while vector sketches offer a simple and direct abstraction to enhance flexibility and editing potential, temporal sketches enrich the representation with dynamic context and pixel-level abstractions or representations of visual concepts. However, each representation brings advantages and challenges and requires specialized processing techniques for robust sketch understanding and generation.

2.2 History of animation techniques

The history of animation outlines its roots from early optical experiments to the sophisticated digital techniques of the present. Traditional animation was created using the illusion of movement through hand-painted or sequential images with optical tools such as zoetropes or magic lanterns before the invention of



(a) Raster sketch **(b) Vector sketch**

Figure 2.1: Raster and vector representations of sketches [142, 143].

modern cameras. These devices relied on persistence of vision, a perceptual phenomenon that allows the human eye to blend discrete images into a single moving scene. Initially, artists and inventors experimented with hand-painted sequences on glass slides or rotating drums, laying the foundation for later developments in animated storytelling. Animation evolved into a structured production process with the advent of film and photography. Subsequently, cel animation introduced the practice of layering transparent sheets (cels) containing drawings over static backgrounds. The workflow evolved to include consistent registration across frames, along with tracing and inking on transparent cels placed over pencil roughs or painted backgrounds. Further, the development of keyframe animation techniques marked a significant milestone: animators sketch the primary poses through keyframes, then fill in the inbetweens by checking the motion before the final painting. This hierarchical process introduces the principles of timing and spacing, allowing animators to control the consistency and expressiveness of movement. Later, the technique enabled more systematic quality control, as animators could preview motion arcs using line tests before committing to final paint and photography. These techniques present fundamental principles in animation, including anticipation, squash and stretch, and follow-through, which remain essential in both traditional and digital animation.

Later, xerography reduced labor intensity by transferring line art directly onto cels, while the rotoscope projector enabled frame-by-frame tracing of live-action footage. Rotoscoping bridged the gap between live-action cinematography and drawn animation, controlling realistic and stylized works. As computing and digital interfaces evolved, animation moved into a new phase characterized by real-time interaction and intuitive control. Moscovich and Hughes [129] introduce motion-by-example systems, where animators can manipulate objects or draw gestures directly on a digital interface, and the system records these motions for playback. This offers intuitive sketch animation for non-experts, though it currently lacks precision control compared to keyframe methods. Recent technological advancements have made more efficient and automatic solutions for

sketch and extended sketch animation by preserving stroke topology, ensuring temporal consistency, and automating in-between frame generation while retaining the expressive qualities of hand-drawn strokes.

Sketch animation has evolved from traditional keyframe and interpolation-based methods to advanced data-driven and generative approaches. Recent methods span motion capture, physics-based simulation, and deep learning, enabling more intuitive, flexible, and intelligent animation pipelines. In particular, the integration of generative AI leveraging diffusion models and multimodal prompts (e.g., video or text) has empowered novice users to produce smooth, expressive animations with minimal input. This chapter also examines recent advancements, highlighting how modern sketch animation methods enable controllable, realistic, and semantically rich animations that bridge the gap between user intent and dynamic motion synthesis.

2.3 Applications

Sketch animation is an adaptable technology with various applications across diverse domains. Its ability to synthesize dynamic visual narratives from abstract and minimal inputs makes it valuable in creative industries and assistive technologies.

Creative tools for artists Sketch animation technologies empower artists by offering intuitive tools for generating animation content from static illustrations. Users can create dynamic transitions, motion sequences, and visual styles with minimal effort. Creative tools leveraging sketch animation allow for rapid visualization of ideas, iterative storytelling, and stylistic innovation, allowing animation production for novice users and animators.

Animation in Education and Storytelling Sketch animation delivers an intuitive and attractive medium for education and storytelling. In educational settings, sketch animation can facilitate the explanation of complex illustrations by visualizing them dynamically. Animators or users can use sketch animation techniques to create interactive enhancements with minimalistic and artistic aesthetics. Moreover, sketch-based storytelling is particularly effective for children's educational content, e-books, and scientific illustration animation.

Medical and scientific visualization Sketch animation helps to visualize biological processes, surgical procedures, or physical phenomena in a simplified and dynamic way. Immersive technologies such as AR/VR will be beneficial for visualizing biological processes. It can serve as an educational aid for students, a training resource, or accessible user information.

Game and character design Sketch animation is essential for rapidly prototyping characters, actions, and environments in the game industry. Through animated sketches, artists can quickly iterate on character motion, pose transitions, motion mapping, character stylization, or scene layouts before committing to fully rendered assets. It also enables the dynamic generation of character actions from raw inputs and supports creative exploration and stylistic diversity in game development.

Virtual avatars and social communication Sketch animation helps to create expressive virtual avatars for social media platforms, virtual meetings, and gaming environments. Sketch-based avatars, driven by user inputs or facial expressions, offer a stylized choice to photorealistic avatars, reducing computational costs while maintaining user privacy and personalization. Also, it represents emotions and gestures in a visually appealing, smooth, customizable form.

Data visualization and infographics Animated sketches can be used to bring data visualizations to real-life use cases. Sketch-based animations can dynamically draw out data trends, making information more engaging and easier to understand, rather than creating static charts or graphs. Designers can illustrate building transformations, mechanical operations, or assembly processes in industrial design workflows through animated sketches.

In summary, sketch animation provides a lightweight, expressive, and highly flexible visual language framework that expands its role in education, design, entertainment, storytelling, healthcare, and the arts, highlighting its transformative potential for the future of visual media.

2.4 Sketch animation techniques

2.4.1 Inbetweening approaches for sketch animation

2D sketch animation focuses on transforming static illustrations into dynamic sequences, allowing for expressive freedom and the creation of controllable animations. Recent methods tackled the problem using geometry-based deformation techniques. The evolution of keyframe-based and inbetweening methods for animation aims to reduce the manual workload of generating intermediate frames while preserving artistic control, style coherence, and structural consistency. Earlier, Bregler et al. [14] uses motion capture data to interpolate between user-defined keyframes, significantly reducing the time-intensive task of hand-drawing every frame. On similar lines, approaches like interactive contour tracking [1] allow users to define object outlines that could be automatically interpolated across video frames. At the same time, bridging the 2D-3D gap,

Sketch animation methods	Additional inputs			Input types		Dimension		Interactivity		Speed		Motion model
	Skeleton	Video	Text	Vector	Raster	2D	3D	Manual	Automatic	Real-time	Offline	
Inbetweening												
Davis et al. [33]	✓			✓		✓		✓		✓		Pose / skeleton editing
BetweenIT [196]				✓		✓		✓			✓	Keyframe interpolation
Autocomplete [212]				✓		✓		✓		✓		Stroke-based prediction
Corda et al. [29]	✓			✓		✓		✓		✓		Skeleton + cage deformation
Jiang et al. [86]				✓		✓			✓		✓	Data-driven inbetweening
Siyao et al. [162]		✓			✓	✓			✓		✓	Line inbetweening (raster)
Chen et al. [23]				✓		✓			✓		✓	Frame interpolation
Brodt and Bessmeltsev [16]	✓				✓	✓			✓		✓	Bitmap + skeleton-aware
Mo et al. [127]				✓		✓			✓		✓	Joint stroke-based
Zhu et al. [240]				✓		✓			✓		✓	Thin-plate / inbetweening
Physics-based methods												
Zhu et al. [238]				✓		✓			✓	✓		Fluid simulation
Physink [152]				✓		✓			✓	✓		Rigid-body physics
Guay et al. [60]	✓			✓			✓		✓	✓		Elastic line dynamics
Lingens et al. [115]					✓	✓			✓		✓	Physics-based optimization
MoCap based techniques												
MotionMaster [108]	✓				✓	✓		✓			✓	MoCap-based transfer
TraceMove [138]	✓	✓		✓		✓		✓			✓	MoCap-based transfer
Pose2Pose [202]	✓	✓			✓	✓			✓		✓	MoCap-based transfer
Animated Drawings [164]		✓			✓	✓			✓		✓	Data-driven mapping
Video-based motion transfer												
Live Sketch [170]		✓		✓		✓		✓			✓	Data-driven mapping
Wakey-wakey [208]		✓	✓		✓	✓			✓		✓	Keypoint / conditional
Xie et al. [210]		✓			✓	✓			✓		✓	Video-based transfer
SketchAnim [142]	✓	✓			✓	✓			✓	✓		Video-based transfer
SketchAnimator [36]		✓		✓		✓			✓		✓	Text-to-Video diffusion
Text-driven animation												
Gal et al. [52]			✓	✓		✓			✓		✓	Text-to-Video diffusion
DynamicTypography [119]			✓	✓		✓			✓		✓	Text-to-Video diffusion
Rai and Sharma [143]			✓	✓		✓			✓		✓	Text-to-Video diffusion
Flipsketch [4]			✓	✓		✓			✓		✓	Text-to-Video diffusion
AnimateSketches [36]			✓	✓		✓			✓		✓	Text-to-Video diffusion
FlexClip [96]			✓	✓		✓			✓		✓	Text-to-Video diffusion
MoSketch [117]			✓	✓		✓			✓		✓	Text-to-Video diffusion
GroupSketch [114]			✓	✓		✓			✓		✓	Text-to-Video diffusion
2D sketch to 3D animation												
MagicToon [45]	✓				✓		✓	✓			✓	Skeleton-driven rigging
Photo-Wakeup [195]		✓			✓		✓		✓		✓	Pose-driven rigging
MonsterMash [39]				✓			✓		✓	✓		Point-based deformation
DrawingSpinUp [236]	✓				✓		✓		✓		✓	MoCap-based transfer
Yoon et al. [221]	✓				✓		✓		✓		✓	MoCap-based transfer
Zhou et al. [235]	✓				✓		✓		✓		✓	MoCap-based transfer
Smith et al. [163]	✓				✓		✓		✓		✓	MoCap-based transfer
Sketch Animation tools / Interfaces												
Kitty [91]				✓		✓		✓			✓	Interactive / rule-based
Draco [92]				✓		✓		✓			✓	Interactive / heuristics
EnergyBrushes [211]					✓	✓		✓		✓		Interactive / physics-guided
MotionAmplifier [94]				✓		✓		✓		✓		Interactive / rule-based
Willett et al. [199]					✓	✓		✓		✓		Mixed (user+auto)
RealityCanvas [207]				✓		✓		✓		✓		Mixed (user + AR blending)
DrawTalking [147]				✓		✓			✓		✓	Text-to-Video / speech-guided
Augmented Physics [62]					✓	✓			✓		✓	Physics-augmented synthesis

Table 2.1: Comparison of prominent sketch-based animation methods.

Davis et al. [33] introduce a system that reconstructs 3D articulated figure poses from 2D sketches, allowing artists to predict keyframes. The system handles the ambiguity of 2D to 3D translation by providing the most likely pose and allowing for user refinement. It is further improved by Dalstein et al. [32] that interpolates vector-based workflows by introducing topology-aware operations that could handle complex changes in shape and connectivity, supporting more robust and flexible animation of vector drawings. Further advancement comes with optimization and learning-based techniques to generate accurate, automated, and consistent sketch frames.

Optimization-based methods

To handle complex sketches, techniques emerge to tackle contour correspondence through patch alignment and clustering strategies to maintain consistent relationships between parts of complex objects [222]. Other directions focused on animating static line drawings [48, 220], such as reconstructing plausible stroke orders for line illustrations, enabling temporally consistent animation, and applying structure-preserving deformation for articulated multi-part objects. BetweenIT [196] advances through semi-automated stroke interpolation techniques that modeled trajectories with logarithmic spirals and curvature warps, focusing on generating tightly aligned inbetweens. Additionally, underlining classical animation principles, Guay et al. [59] incorporate the line of action as a structural and expressive guide for pose creation, aligning computational techniques with foundational animation principles. This trajectory highlights a progression from data-assisted interpolation to semantically aware, style-preserving stroke animation, reflecting an effort to blend automation with artistic intent in modern animation pipelines. Further, Quasi-3D rotation [50] enables interpolation of facial sketches across turning views by modeling approximate 3D motion from 2D inputs, allowing for convincing face rotations without requiring full 3D reconstruction. On similar lines, Autocomplete [212] presents a user-guided predictive system in which the animator provides a keyframe stroke, and the system uses a graph-based pattern-estimation module to predict the next stroke in the sequence. This interactive approach allows users to accept, reject, or edit each predicted frame, balancing automation with creative oversight. These developments reflect a growing focus on structure-aware, predictive, and editable animation systems that accelerate the inbetweening process while respecting the user's stylistic and compositional intent. Recently, inbetweening techniques have progressed toward context-aware and structure-preserving approaches that enhance automation and artistic control. Initial improvements present contextual coherence via context-aware inbetweening [219], which considers the spatial neighborhood of strokes to produce smoother and more visually consistent transitions. At the same time, DiLight [18] incorporates guideline-assisted curve correspondence, improving control over curved stroke behavior during interpolation. At

the structural level, adaptations of optical flow for line art introduce pixel-free motion estimation via distance transforms, making flow-based techniques applicable to sketch inputs [132].

Several methods leverage artists' realistic drawing skills to create intuitive, artist-friendly controls that simplify complex tasks. The objective is to move from manual, low-level manipulation to high-level, artistic control that maintains visual fidelity and control. Hahn et al. [68] introduce a sketch-based approach that enables intuitive static character posing. Extending from static to dynamic control, SketchiMo [26] introduces a sketch space that lets users edit motion trajectories and abstract animation constraints directly through drawing, with the system translating these sketches into optimization constraints for motion refinement. Simultaneously, hybrid deformation systems, Corda et al. [29], enable unified manipulation of the posed skeleton, the rest cage, and the deformed cage. In this formulation, the articulated skeleton pose drives structural motion, while the rest cage defines the geometric embedding of the mesh. The resulting deformed cage, updated through coupling constraints, refines and stabilizes the surface deformation. By tightly synchronizing with the framework, the framework combines the structural advantages of skeleton-driven articulation with the expressive flexibility of cage-based mesh deformation, as shown in Figure 2.2. These techniques established a robust theoretical and computational foundation for high-quality, real-time, and structure-preserving deformations, operating as core tools in modern sketch and character animation systems.

More advanced interpolation techniques have expanded the expressiveness and control in animation by incorporating structural awareness, partial automation, and user-in-the-loop prediction. It is further extended by Shen et al. [154], introducing multi-level sketch-aware interpolation with region, stroke, and pixel-level guidance, which preserves high-level structure and fine-grained detail. Later, to handle occlusions and depth in bitmap sketches, Brodt and Bessmeltsev [16] combine skeleton motion, bitmap deformation, and 2.5D lifting, enabling plausible interpolation even in complex scenes. Pushing toward rough-sketch workflows, the non-linear embedding-based method [42] enables transient stroke representations that support expressive and non-deterministic interpolation for rough, exploratory sketches (see Figure 2.3). Pushing further into rough sketch domains, Even et al. [43] introduce occlusion-aware inbetweening for rough drawings with dynamic layouts and stroke visibility control to handle appearance in layered, evolving compositions. Recently, Cartoonimator [144] introduced a paper-based tangible interface for keyframe animation, allowing users to create animations through physical manipulation of drawings rather than conventional digital timelines. The approach emphasizes embodied interaction to reduce the learning curve for novice animators and support creative exploration. The work highlights the potential of tangible and sketch-based systems to improve accessibility and engagement in animation authoring tools.

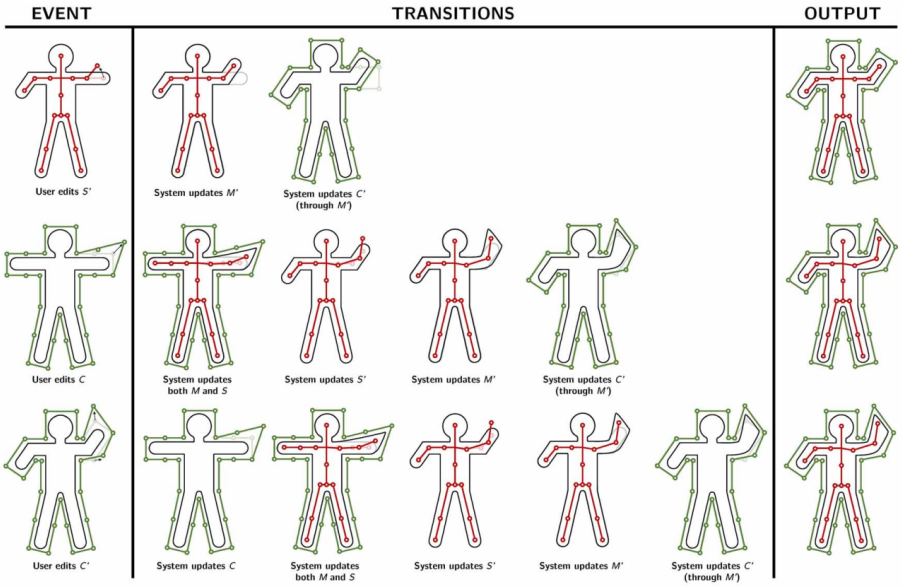


Figure 2.2: A hybrid deformation framework that supports real-time editing of either the skeleton in the deformed pose or the deformation cage in both rest and deformed configurations [29]. All modifications are automatically propagated, ensuring consistent synchronization across all deformation components.

Learning-based interpolation techniques

The advent of learning-based inbetweening brought structure awareness and generative capabilities to the field. Chen et al. [23] introduce a method for efficient stroke selection in vector-based interpolation to enable frame prediction, while self-occlusion is tackled through boundary-aware stroke propagation [86] as given in Figure 2.4. These methods signal a shift toward flexible, semantically grounded, and structurally robust inbetweening frameworks capable of supporting a wide range of animation styles and input fidelity levels, from clean vector drawings to rough, gesture-based input. Building on the evolution of inbetweening, recent methods introduce multi-level stroke correspondence, hybrid representations, and occlusion-aware deformation techniques to push the limits of automation and expressiveness in sketch-based animation. Siyao et al. [162] propose AnimeInbet, reframing line inbetweening as graph fusion of vertices. Mo et al. [127] propose a method that enables accurate vector-stroke correspondence across frames, thereby improving temporal consistency in vector animations. Complementing this, Zhu et al. [240] introduce thin-plate spline-based interpolation with a keypoint-matching module, estimating per-frame keypoint flows and enabling smooth, large-motion interpolation between line drawings. These advancements reflect a trajectory toward semantically controlled, structure-aware, and occlusion-resilient inbetweening, making the animation of vector and exploratory rough drawings more accessible, flexible, and robust.

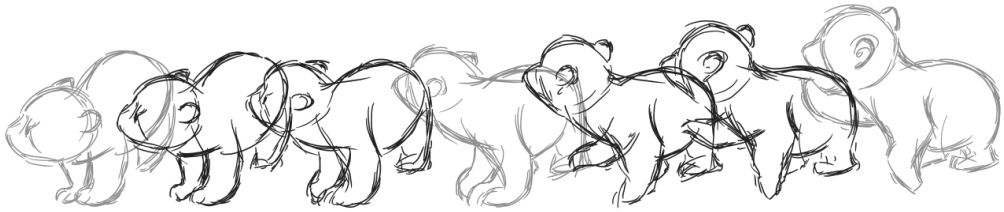


Figure 2.3: Rough line-art interpolation [23]. It takes rough vector key drawings (light gray), generates stylistically consistent intermediate strokes (black), and uses an interactive stroke distribution synthesis algorithm that minimizes temporal artifacts.

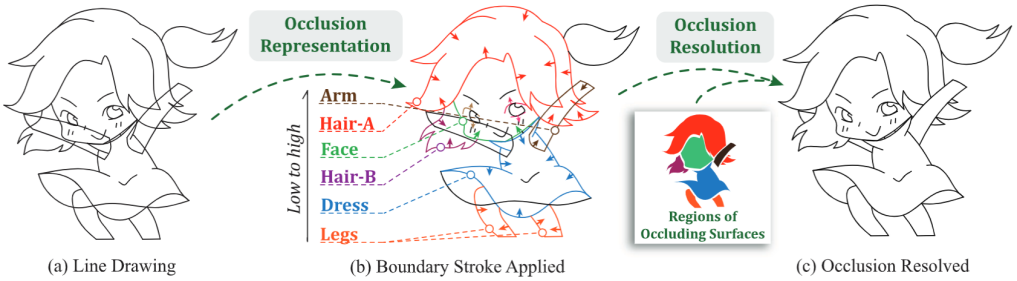


Figure 2.4: Two-stage occlusion resolution in stroke-based drawings [86]. (a) Original line drawing composed of multiple strokes with occlusions. (b) Occlusion representation using labeled boundary strokes, where half-arrows indicate relative depth ordering. (c) Occlusion resolution through identification of regions corresponding to occluding surfaces.

The trajectory reflects a progression: from geometry and contour-based methods to structure-preserving interpolation, then to learning-based, stroke- and skeleton-centric systems. Each stage addressed weaknesses of the previous methods, moving from simple contour tracking to handling complex multi-contour objects, then to occlusions, topology changes, and finally to deep models that capture stroke-level and semantic priors. Modern systems integrate multi-level correspondence (region, stroke, pixel), skeleton guidance, and deep generative priors to balance automation with artistic control. Combined, these approaches mark the transformation of inbetweening from a labor-intensive manual process into a semi-automated, structure- and style-preserving workflow, significantly reducing production costs while enlarging creative freedom.

2.4.2 Physics-based techniques

In recent years, physics-based animation techniques have evolved from simple static imagery and keyframe-based representations to immersive, real-world interactive storytelling, with a focus on fluid simulation, physical constraints, and realistic motion. Traditional methods, such as Chuang et al. [27], use spectral noise synthesis to generate stochastic motion textures for layered still images,

producing looping video textures of passive elements. It extends realistic fluid motion by decomposing example videos into average images, flow fields, and residuals, allowing users to paint and refine custom flow patterns for target still images [136]. These approaches gradually evolved into more interactive systems, such as sketch-based hydraulic simulations and direct manipulation tools that enable real-time fluid illustration through 2.5D sketch-based hydraulic graphs [238]. This framework illustrates many fluid systems, such as physiological and engineering fluid systems, while Physink [152] brought direct manipulation of physics-enabled sketches for causal, editable animations. This evolution was advanced by tools that simplified novice workflows using cutout and layered interfaces [165], and by systems that introduce physically-based elastic motion into character rigs, enhancing expressive dynamics [60]. Later, Lingen et al. [115] automate character animation through neural networks and evolutionary search in a physics-based environment. Integrating neural networks and generative models enables more refined control over fluid motion, including the generation of velocity fields from sketches [79]. Immersive platforms expand this paradigm into AR, VR, and mixed reality, allowing animated sketches to respond to real-world objects for responsive motion and user gestures, and to integrate motion and audio brushes for mixed-reality animated storyboards [41, 105, 174].

However, despite advances in artistic control and responsiveness, these systems remain limited in their ability to fully reproduce realistic, physics-based fluid behavior in sketches, highlighting a gap in integrating sketch-specific stylization with physically accurate fluid simulation.

2.4.3 Data-driven techniques

This section presents an overview of data-driven techniques for sketch animation. In recent years, researchers have explored diverse motion sources, including motion capture (MoCap) data, videos, and text, to extract motion information and map it to sketch-based representations.

MoCap-based sketch animation methods

MoCap offers a wide range of motion-driven, skeleton-based, and data-driven animation techniques to enhance motion consistency, stylization, and accessibility for experienced animators and beginners alike. Previously, efforts in this field focused on enhancing raw MoCap data with stylized 2D deformations, injecting artistic expressiveness into rigid 3D motion sequences [111]. Subsequent methods align MoCap trajectories with human body models [33], producing accurate, realistic animations, although such approaches are limited in handling non-human or arbitrary forms. To introduce creative flexibility and secondary motion, systems emerge [97, 108] to retrieve and refine 3D motions from user-drawn pose sketches,

enabling intuitive animation prototyping without complete MoCap setups and uses secondary motion that allows deformation of surrounding objects based on the dynamics of a driving shape. Other techniques support user-performed motion transfer in cutout-style animations and introduce skeleton-based deformation pipelines that simplify 2D animation production [7, 137]. Additionally, to bridge the physical correctness, refinements such as spacetime vertex constraints refined the MoCap sequences to meet physical plausibility and environmental constraints. It illustrates structured motion data, user input, and stylized control, paving the way for accessible yet physically grounded animation systems.

Recent methods have moved toward stroke-centric learning-based models. These methods are a progressive blending of motion capture, data-driven, sketch-based input, and learning-based representations to support both precision-driven animation (MoCap, skeleton embeddings, physical models) and creative, stylized, or accessible workflows (sketch-to-motion, style transfer, pose mapping). Further expanding the data-driven animation pipeline, techniques have prioritized accessibility, physical accuracy, and real-time interactivity, especially for users with minimal artistic or technical expertise. 3A2A [31] allows stick figure inputs to be transformed into expressive hand-drawn animations. Stroke-centric and sketch-guided techniques, such as those enabling motion transfer from video or example-based style propagation, now offer powerful tools for direct, intuitive control while embedding data-driven intelligence for automated motion inference. In sketch-based workflow, Patel et al. [138] introduce Tracemove, a data-assisted system that offers frame-by-frame animation support by incorporating predefined motion patterns. While such tools offer greater control and stylistic guidance, they often rely on manual input across multiple stages, underlining the need for future systems to blend automation with intuitive authoring interfaces. Parallelly, data-driven systems focus on stylized rendering in line drawing animations from videos [10].

Subsequent advancements introduce structure-aware deformation, skeleton-agnostic representations, and style-aware motion synthesis, expanding applicability to various motion domains and animation. In this direction, Pose2Pose [202] maps performer-specific gestures to 2D characters. Earlier methods used MoCap for motion retargeting but were limited to human-like motions; subsequent techniques added structure-aware deformation, skeleton-agnostic embeddings, and style-aware synthesis, enabling adaptation to diverse characters, styles, and motion domains. Further, ToonSynth [38] explores the technique of transferring style and motion by learning from hand-colored animations, enabling consistent stylistic and motion transfer to new skeletal sequences. Meanwhile, high-fidelity motion capture systems [101] demonstrate their value in open-domain animation tasks, offering precise control and dense tracking, though with increased complexity and domain-specific constraints. These systems collectively reflect a broader push toward generalizable, stylized, and artifact-free animation pipelines that support creative flexibility and structural adaptability.

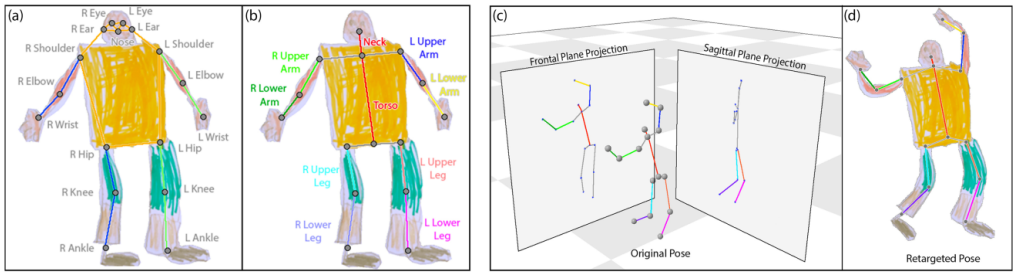


Figure 2.5: Skeleton mapping approach of Animated Drawings method [164]. (a) From the predicted joint keypoints (b) are used to construct a skeletal rig that drives the character animation. (c) The original motion capture pose is taken, with torso and upper-limb joints projected onto a frontal plane and lower-limb joints onto a sagittal plane. (d) The global orientations of the bones are then computed within their respective planes, and the character's joints are rotated accordingly, resulting in the retargeted pose.

Meanwhile, Animated Drawings [164] integrates pose estimation from video, such as alphapose, with skeletal rigging to animate hand-drawn characters. An ARAP deformation loss preserves structural consistency during joint rotations, maintaining the visual fidelity of the original sketch given in Figure 2.5. This system constructs a character rig by projecting motion-capture anchors to skeletal joints and computing bone orientations within their respective planes. However, this approach is primarily limited to bipedal motion, reducing its generalizability to non-human objects or characters. These stroke-level and pose-to-sketch systems represent a promising step toward precise, temporally-aware, and artist-preserving animation frameworks, balancing data-driven automation and user-controllable semantics in sketch-based animation.

Video-driven sketch animation

Traditional sketch animation tools primarily rely on keyframe-based animation or motion capture (MoCap) data. Video motion retargeting aims to map the video motion from source to target, which enables identity-preserving motion and controllable motion transfer across different identities. Cross-domain video motion retargeting addresses the challenge of motion transfer between different domains, such as real video to sketch, avatars, and cartoon characters. Traditional motion retargeting tools operate within a single domain and do not tackle the domain gap in appearance, articulation, and temporal coherence. Single-domain motion models (SDMM) [156, 159, 177] are not suitable for sketches due to the distinctive nature of sketches. Cross-domain motion models [75, 186, 215, 230] overcome the challenges of SDMM by establishing the pose correspondence and alignment. However, sketch animation remains a challenge, and these methods cannot preserve the appearance and motion information. Further, a cross-domain motion

model JOKR [128] proposes a framework that learns joint keypoints of input videos and domain confusion loss for shared representation. Further, the shared feature information is passed to the generator module to generate animated sequences. The advantage of this method is that it does not require a large dataset for training. This method represents significant strides in bridging the gap of cross-the-motion retargeting and tackling identity preservation, stylization, and temporal consistency challenges. Further, cross-domain video motion retargeting for sketch animation, moving from early pose-alignment and keypoint-based mapping to modern diffusion- and stroke-level approaches that enable automatic, flexible, and controllable motion transfer while preserving sketch appearance and structure. Diffusion-based methods [76, 210, 216] require extensive training data and provide slow inference. Also, it does not give attention to motion consistency in the animated sketches.

With the rise of video-assisted sketch animation, researchers have focused on extracting motion cues from real-world videos and transferring them to freehand sketches, enabling dynamic animation from static drawings. Live Sketch [170] introduces a video-driven sketch animation framework by presenting a control point-based tracking technique that estimates motion trajectories from video and maps them onto corresponding sketch control points. This two-stage pipeline motion extraction and motion transfer allows video-based motion to drive the deformation and animation of sketches. To preserve the visual quality of drawn elements, the system uses a multi-layered approach that handles self-occlusion and a stroke-preserving ARAP function for rigidity constraints, ensuring the strokes maintain their original shape during deformation.

Recent advances in sketch animation have increased focus on stroke-level representations, enabling better control over shape and motion while improving temporal consistency and interpretability. In this direction, Wakey-Wakey [208] proposes a stroke-level representation capable of generating sketch animation by disentangling shape and motion, allowing independent control of object geometry and motion, and producing temporally consistent animations from static input. Further, sketch animation methods have increasingly focused on video-driven motion customization and cross-domain compatibility, enabling more expressive and adaptable workflows. In this domain, SketchAnimator [218] introduces a one-shot motion customization pipeline through a structured three-stage process: appearance learning, motion learning, and video prior distillation using SDS loss. This framework allows users to transfer motion from a reference video to a sketch while preserving visual style and temporal consistency. Expanding on this, Xie et al. [210] propose a cyclic reconstruction mechanism to improve motion consistency and domain adaptability. However, its reliance on domain-specific training data and limitation to specific motion types restricts generalization. Recently, Zhu et al. [241] introduced a differentiable motion trajectory–based framework that models vector sketch stroke motion as continuous polynomial functions, enabling globally optimized, temporally coherent sketch animations

with reduced flickering and improved consistency.

These works advanced integration of motion extraction, cross-domain adaptation, and appearance preservation techniques. Previous methods provided coarse motion transfer but lacked domain-specific fidelity; mid-stage approaches introduce stroke-level and region-guided constraints to preserve sketch appearance during deformation; recent methods integrate diffusion priors, motion decomposition, and skeleton-based mapping to achieve higher fidelity, temporal coherence, and flexibility across motion types. It marks a shift from domain-agnostic motion mapping toward sketch-aware, structure-preserving, and semantically controllable animation pipelines, significantly lowering the manual workload in 2D sketch animation production while expanding creative control. It highlights the growing focus on motion fidelity, adaptability, and semantic mapping in sketch animation pipelines, while highlighting open challenges such as 3D generalization, data efficiency, and occlusion handling in freeform animation systems. Further, combining large language models and natural language understanding with sketch-based animation has enabled a new generation of systems in which users can drive motion, expression, and interaction in sketches with simple text prompts.

Text-driven sketch animation

Recent research has marked a significant shift in sketch animation toward semantic, language-driven, and diffusion-based workflows, enabling users to animate complex sketches, clipart, and narrative scenes.

Text-driven animation moves beyond traditional keyframing by leveraging text-to-video diffusion models and geometry-aware losses to animate complex scenes in a controllable yet automated way. Systems such as Gal et al. [52] pioneered this direction by introducing a text-to-sketch animation by using pretrained text-to-video diffusion models with score distillation sampling (SDS) [139] loss to animate Bézier-curve sketches using local deformation and global transformation modules. Further advancement adapts diffusion priors for flipbook-style raster animations [4], introducing fine-tuning for sketch-style frames, reference frame noise refinement, and dual-attention composition for visual consistency. These approaches combine static art and temporal storytelling with minimal artist effort. Extending beyond sketches, Dynamic Typography [119] advances text-prompt driven animation to deform and animate letters while preserving legibility, and AniClipart [205] applies Bézier-curve motion regularization and ARAP deformation to clipart, followed by FlexiClip [96], which adds temporal Jacobians, continuous-time modeling, and flow matching loss for smooth, coherent clipart animations. These methods redefine the sketch animation pipeline by tightly integrating semantic control, diffusion-based generation, and geometric regularization, setting a foundation for accessible yet expressive animation.

Recent approaches have made significant advances toward multi-object, instance-aware, and story-driven animation. These techniques enable the animation of complex vector scenes while maintaining semantic coherence, motion synchronization, and storytelling. For instance, AnimateSketches [36] introduces prompt-guided instance-aware masks (PGIM) and mask-based SDS to animate multi-object vector sketches without affecting the rest of the scene elements, while MoSketch [117] combined LLM-based scene parsing and motion decomposition with compositional SDS loss to achieve synchronized multi-object animation. Further, Liang et al. [114] propose a structured two-stage pipeline for animating complex vector sketches with multiple objects, consisting of motion initialization and motion refinement, where users can group sketch parts and assign keyframes. A group-based displacement network (GDN), enhanced with context-conditioned features and a text-to-video diffusion backbone, refined these motions to produce temporally consistent, smooth animations, making the workflow intuitive. Stepping into narrative generation, FairyGen [232] automates story-driven cartoon creation from a single child's drawing by using MLLM-based storyboarding, style-propagation adapters, cinematic shot design, 3D proxy motion reconstruction, and an image-to-video diffusion model with a two-stage motion customization adapter for identity preservation and temporal modeling. These methods mark a substantial hop toward semantic, multi-object, and story-centric animation systems, blending LLM reasoning, visual grounding, and diffusion-powered generation to achieve expressive, scalable, and accessible animated content creation.

These works have a clear trajectory from stroke-level motion control to text-driven scene animation using large pretrained diffusion and language models. Previous methods and their enhancements tackled single-object, motion-preserving animation, which evolved into style-preserving and temporally consistent pipelines. Recent advances address multi-object awareness and complex motion representation, culminating in full-scene, story-driven systems that integrate LLM-based planning, style transfer, and physically plausible motion. It marks a paradigm shift toward intuitive, text-and-sketch-driven authoring pipelines in which artistic style, temporal coherence, and narrative structure are jointly optimized, drastically lowering the barrier to creating high-quality, expressive animations.

2.4.4 2D sketch to 3D animation

2D-to-3D character animation has emerged as a promising research area that aims to transform static 2D sketches into animated 3D models. In this direction, Reinert et al. [145] focus on generating 3D textured animation sequences from the 2D video using skeletal sketching, followed by segmentation, tracking, and texturing. Building on this foundation, Bessmeltsev et al. [11] introduce gesture-based sketching on input character models, projecting 2D skeletons to estimate new poses. It allows for more expressive user input but still relies heav-

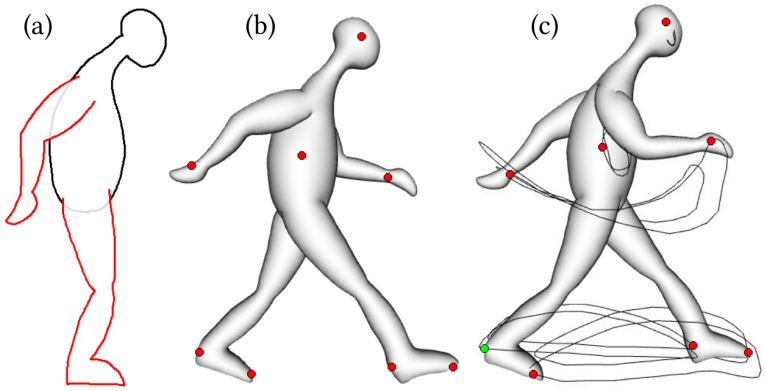


Figure 2.6: Example of a walking cycle generated with Monster Mash [39]. (a) A hand-drawn sketch (b) is first converted into an inflated 3D model with editable control points. (c) The final walking animation is created by defining and recording the motion trajectories of these control points.

ily on existing 3D templates, limiting the generation of entirely novel characters. Subsequent systems, such as MagicToon [45], incorporate an augmented reality interface that allows users to draw a 2D sketch, generate a corresponding 3D model, and deform it via skeletal manipulation. This improves accessibility for novice users, enabling them to create, manipulate, and rig 3D models from simple 2D sketches.

Building upon previous sketch-to-3D animation frameworks, Weng et al. [195] propose Photo Wake-up, which takes an image input, performs segmentation and 2D pose estimation, and uses an SMPL [121] template model to project the 2D pose on the image with a skinning map. It uses a rigged 3D mesh with a combined depth and skinning map to animate it with Mocap, and textures it with an inpainting background. Progressing further, sketch-based approach [15] enables direct 3D pose retrieval from bitmap inputs without requiring detailed structural information. The pose uses custom rigging and skinning using standard retargeting tools. It enabled freehand creative input but focused on pose reconstruction rather than full 3D textured animation. Although this improved motion consistency and flexibility, it requires a detailed sketch structure, and authenticity was not completely ensured. Simultaneously, a more playful and intuitive system called MonsterMash [39] introduces an interactive solution where users draw the strokes, and the method inflates the sketch drawing and animation in real-time into a 3D deformable model as shown in Figure 2.6. It proposes a user-friendly interface for 3D character creation from 2D drawing and manipulation by integrating sketch-based modeling with a physics-inspired animation system. This trajectory reflects a shift toward systems highlighting real-time interactivity, minimal user effort, and more prominent creative tools, bridging the gap between simple sketching and expressive 3D animation.

In recent years, hand-drawn 2D sketch to 3D model generation techniques [109, 168, 180, 226, 231] provide an edge to generate the textured animation-ready 3D meshes directly from 2D sketches. These methods provide high-fidelity reconstructions with rich detail and enable animation-ready meshes. However, they often depend on user-provided constraints or structural hints, limiting their utility for unstructured freehand art. Removing this dependency for the complex structure of 2D textured sketches, DrawingSpinUp [236] presents a framework to animate characters from single drawings by inferring 3D geometry and motion cues using Mixamo deformation priors and shape-consistency networks. It synthesizes plausible 3D motion without requiring complex user input, thus bridging the gap between casual 2D sketching and dynamic 3D character animation and opening new possibilities for accessible, sketch-driven animation pipelines.

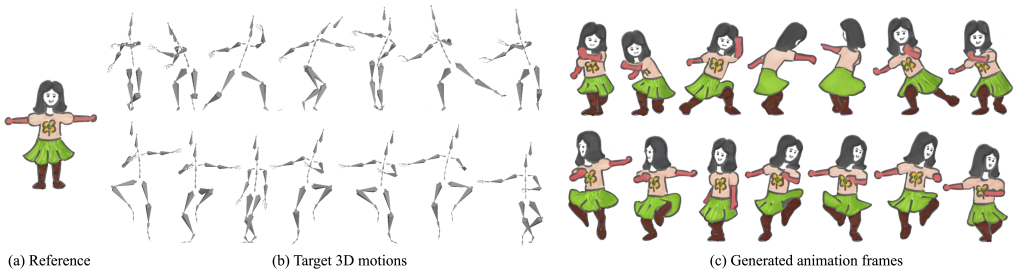


Figure 2.7: Illustration of the 3D skeleton-driven sketch animation pipeline proposed by Zhou et al. [235]. A single input drawing (left) is combined with a target 3D motion sequence (middle) to generate the resulting animated character (right).

Recent advancements in 2D-to-3D character animation have increasingly prioritized freeform sketch input, shape consistency, and robustness to occlusion, pushing the boundaries of accessibility and visual fidelity. It allows freeform sketch drawing to 3D deformation with shape consistency using stylization, without providing additional structural information. These models achieve consistent shape-preserving stylization across motion sequences, but face challenges such as self-occlusion and visual artifacts. Addressing this, Yoon et al. [221] introduce an occlusion-robust stylization technique that integrates flow-depth edge detection (FDED) to infer reliable edge maps even under unseen or occluded poses. This framework ensures smooth, stable, and non-jittery animated sketches even in the presence of self-occlusion. On a similar note, Zhou et al. [235] introduce a hybrid animation approach that combines 3D skeletal guidance with diffusion-based video generation to produce natural, temporally coherent animations of hand-drawn 2D characters. By refining rigged motion with a domain-adapted diffusion model that adds realistic secondary dynamics, it produces expressive, high-quality animations while preserving the original artistic style as given in Figure 2.7. Complementing this, Smith et al. [163] introduce an animation system that takes a single childlike figure drawing and converts it into a 2.5D character model, animated using 3D skeletal motion and viewed from different viewpoints.

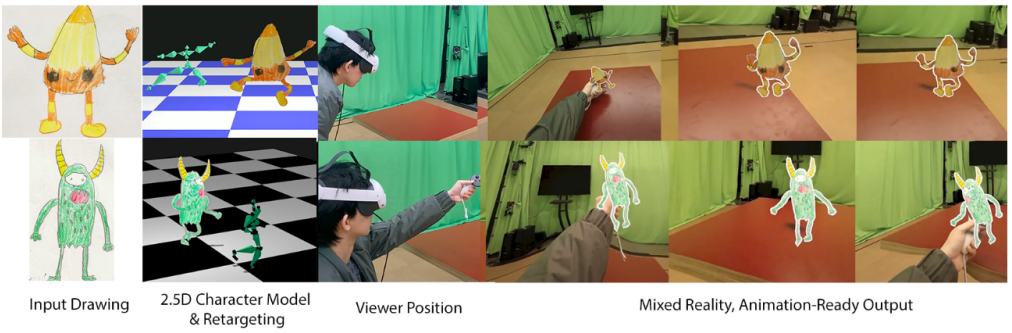


Figure 2.8: A real-time animation system that transforms a single childlike drawing into a 2.5D character viewable from multiple views [163]. It supports motion retargeting using any 3D skeletal motion, making it essential for mixed-reality applications.

It offers a Mixed-reality application suitable for real-time modeling and animation and generates view-dependent motion retargeting, as shown in Figure 2.8. Further, sketch-based character animation has reached a new milestone with the introduction of end-to-end systems capable of generating fully textured, rigged, and animated 3D characters directly from freehand sketches and narrative instructions [233]. This approach represents a significant leap in expressiveness and automation, unifying storyboarding, rigging, and motion synthesis into a single pipeline. Integrating sketch-based priors with semantic action cues represents the most advanced step yet, enabling textured, rigged, and animated 3D characters directly from freehand sketches and narrative instructions. However, the computational cost of large-scale learning models and the challenge of balancing automation with fine-grained user control remain open research problems.

2.4.5 Sketch animation interfaces

Sketch-based animation has undergone a remarkable evolution over the past few years. Starting with keyframe and data-driven methods to advance automatic and stylized techniques using generative models, researchers have constantly pushed the boundaries of how novice animators can create animation smoothly with less time and effort, without compromising the animation quality. Early systems in sketch-based animation focused on enhancing traditional workflows by integrating computational capabilities directly into the artistic process. One major direction involves augmenting hand-drawn keyframe animation [85] with 3D physical effects such as cloth, fluids, and particles into 2D drawings, enabling secondary motion while preserving the animator's control. Efforts have emerged to digitize tactile art forms through multitouch platforms that recreate sand animation, simplify creation, mix with video, and enable gesture-based replay, bridging the gap between tactile artistry and digital flexibility [93]. These developments emphasized preserving the expressiveness of handcrafted input while

offering digital flexibility. Stylization tools [95] extend this principle to pen-and-ink illustrations, enabling partial texture drawing with automatic synthesis and editable filling, preserving the artist's style while reducing repetitive work. Sketch-based interfaces soon expanded into educational contexts [21], automatically animating sketched physics diagrams by recognizing shapes, annotations, and equations, and using a domain-specific physics engine for validation, illustrating how sketch-based interfaces can integrate reasoning and visualization. Later, systems tackled the challenge of bringing static drawings to life by allowing artists to define example poses, interpolation rules, and procedural physics-based transitions, thus merging handcrafted pose design with simulation-driven motion for broader reusability [88].

Further, researchers made a significant effort to introduce dynamic and expressive possibilities for animated illustrations. A wave of sketch-based animation tools [91, 92] that move beyond aiding static-to-dynamic transformation toward rich behavioral control, procedural effects, and real-time interaction, while maintaining an accessible interface for both novices and experts. In the continued effort to facilitate and enrich sketch-based animation, subsequent systems introduce more dynamic and interactive capabilities while preserving the expressive integrity of hand-drawn illustrations. One key advancement enables artists to embed continuous, coordinated motions into static sketches using motion handles and kinetic textures, enabling dynamic environmental effects like swaying leaves or rippling water without disrupting the timeless quality of still art [92]. This approach maintained the timeless visual style while layering in dynamic elements.

Building on interactive storytelling, Kitty [91] extended sketch-based animation to include functional relationships between entities via a graph-based model and narrative interactivity, making it suitable for applications in infographics, educational content, and interactive storytelling, such as children's books, where visual elements can behave in contextually appropriate and responsive ways.

Sketch-based animation continued with systems that expanded the scope of motion control, expressiveness, and interactivity while minimizing dependency on complex inputs like motion capture or dense datasets. More advancements enable artists to define complex 3D motions from single strokes through space-time curves and dynamic lines of action [61], offering complete geometric control without requiring motion capture or large datasets. Building on direct manipulation, adding user-controllable motion effects using brushes introduces user-controlled stylized effects called EnergyBrushes [211], which shifted toward stylized natural phenomena, allowing coarse energy strokes to drive particle-based simulations of water, fire, or smoke, combining physical plausibility with artistic expressiveness. Further simplification came through a modular exaggeration tool [94] that allows users to deform and enhance motion effects using plug-and-play amplifiers, enabling rapid iterations even by novices. Real-time performance contexts

were also addressed, with interfaces that triggered animations via multitouch input and predictive suggestions [201], and systems that automate the addition of physically plausible secondary motion [200] to live-performed animations using parameterized rigs. On similar lines, Ciccone et al. [28] targets cyclic animation creation, offering a capture-based interface to extract and edit loops easily, while a mixed-initiative system [198] emerges that progressively layered control and automation, allowing artists to animate still images using scribbles for segmentation, texture extraction, and kinetic texture synthesis. by bridging static composition with dynamic expressiveness while maintaining a high degree of user agency and creative flexibility.

In recent years, sketch-based and immersive animation tools have highlighted cross-modal interaction, immersive environments, and generative AI integration to expand creative possibilities beyond traditional 2D interfaces. For instance, the augmented reality platform [207] advances AR sketching by enabling responsive, improvisational animation in real-world contexts, offering six animation primitives such as object binding, particle effects, and flip-book animation derived from real-world scribble animation practices, making it suitable for storytelling, education, and prototyping. Parallelly, cross-modal systems push natural interaction by combining sketching and speech [147] to create and control interactive worlds, giving users programming-like control without coding and enabling fluid, improvisational world-building for narrative contexts. In a production setting, Guajardo et al. [58] demonstrate how generative models can integrate into professional workflows to facilitate the creative pipeline, enhancing efficiency while preserving artistic intent. Additionally, immersive editing tools [237] tackle VR motion editing by merging spatial and temporal control into a unified interface through keyposes and trajectories, allowing intuitive, direct manipulation of complex 3D motion with reduced editing time. These innovations connect static educational content and interactive simulation by leveraging computer vision (segment anything [100]) and multimodal LLMs to extract textbook diagrams and transform them into embedded, interactive physics simulations, enhancing learning engagement [62].

Sketch animation tools progressively reduce manual workload while maintaining creative control, incrementally automating the animation process, from physical effects and gesture replication to texture generation, semantic interpretation, and procedural motion, moving from assisting traditional animation to creating dynamic, adaptable animations with minimal artist input. Further advancements were made with progressive layering of control, automation, and interactivity. They evolve from introducing procedural motion into static sketches toward stroke-based specification of spatial-temporal movement and particle-driven dynamic effects, to stylization and exaggeration tools, and real-time performance integration. Later systems expand into specialized workflows like motion loops and novice-friendly static image animation. Collectively, they trace a shift from isolated effect generation to integrated frameworks that combine

Table 2.2: Comparative analysis of sketch animation techniques with attribute-supported citations

Attributes / Techniques	**Inbetweening approach**	**Physics-based**	**Data-driven**			**2D sketch-to-3D model animation**
			MoCap	**Video-driven**	**Text-driven**	
User Control	High [16, 196, 212]	Moderate [60, 152, 238]	Low [164]	Low [210]	Low [4, 36, 119, 205]	Moderate [39, 45]
Automation	Low [132, 196]	High [105, 152, 238]	High [31, 164]	High [208, 210]	High [36, 52]	High [221, 236]
Generalization	Low [212]	Moderate [115]	Moderate [138]	Moderate [218]	High [114, 119]	Moderate [45, 236]
Temporal Coherence	High [132, 154]	High [41, 152]	Moderate [38]	Moderate [241]	Moderate [4, 52, 96]	Moderate [195, 221]
Realism	High [132, 240]	High [60, 174]	High [164]	High [210]	Moderate [114, 205]	Moderate [235]
User effort	Moderate [127, 212]	High [105, 115]	Moderate [31, 164]	Moderate [210]	Low [52, 117, 119]	Moderate [39, 163]
Flexibility	Moderate [16]	Low [238]	Moderate [138]	Moderate [218]	High [36, 114]	Low [236]
Speed	Moderate [196]	Low [152]	Moderate [164]	Moderate [170]	Low [36, 119]	Low [221]
Expressiveness	High [16, 154]	High [105]	Moderate [38]	Moderate [170]	Low [52, 205]	Moderate [45, 235]
Abstraction Level	Low [132]	High [41, 238]	Low [164]	Moderate [241]	High [4, 52, 119]	Low [39]

user-driven artistic input with procedural, physical, and interactive systems, pushing sketch-based animation toward being both expressive and accessible in live, interactive, and production contexts. Further recent methods lie in their shift toward multimodal, immersive, and AI-assisted authoring, expanding from earlier sketch-based animation systems into real-world integration, multimodal storytelling, production-ready AI augmentation, immersive motion control, and domain-specific intelligent content transformation.

2.4.6 Comparative analysis of animation techniques

Each technique is discussed in the context of sketch animation, followed by a comparative study of its suitability for different animation types and use cases, highlighting the conditions under which each approach is preferable and the associated trade-offs. This analysis identifies which methods are effective under challenging conditions and which are sensitive to stroke ambiguity, occlusion, topological changes, or interactions among multiple objects.

User control User control refers to the extent of manual intervention and precision that an animator possesses during the animation process. Keyframe-based interpolation [16, 196, 212] provides strong control, enabling users to define essential poses and adjust timing. In contrast, physics-based techniques [60, 152, 238] provide moderate control, as motion generation depends on predefined physical constraints and simulation parameters rather than user manipulation. At the same time, data-driven methods, including MoCap [164], video-driven [210], and text-driven systems [36, 119, 210], significantly reduce user control, with the animation system making most motion decisions based on input data or natural language descriptions. Similarly, 2D sketch–to–3D model animation techniques [39, 45] provide moderate control, as user input defines sketch drawing and pose while motion generation is partially automated.

Automation This feature captures how much of the animation process is automated by the system. Keyframe-based interpolation methods [132, 196] offer low automation as the system primarily assists with generating inbetween frames based on manually defined key poses. In contrast, physics-based animation techniques [60, 105, 152] achieve high automation by simulating motion dynamics controlled by physical laws, enabling realistic effects such as collisions, elasticity, and inertia with minimal user intervention. Data-driven methods, such as motion-capture-based systems [31, 164] and video-driven methods [208, 210], are fully automated, as motion patterns are learned from real-world performance data or directly captured from video, respectively. Similarly, text-driven methods [36, 52] are highly automated, relying on generative models to synthesize motion from natural-language descriptions. 2D sketch to 3D models [221, 236] where predefined motions often derived from motion-capture or video data are automatically mapped onto 3D models based on user-provided sketches.

Generalization Generalization refers to an animation system's ability to adapt to diverse characters, motion styles, and drawing variations without retraining or supervision from specific animation sources or datasets. Keyframe-based interpolation systems [212] exhibit limited generalization, as motion retargeting is typically required to transfer animations across characters with different proportions or skeletal structures. Physics-based methods [115] offer moderate generalization, since their adaptability depends on consistent physical parameters and accurate geometric representations, which may not generalize well across stylized or abstract characters. Data-driven approaches such as MoCap [138] and video-driven methods [218] depend on how closely the target motion resembles the source during motion transfer, offering moderate generalization, while text-driven methods [114, 119] show strong generalization through generative models that can synthesize motion from abstract text descriptions. However, their performance is bound by the variety and quality of training data. Finally, 2D sketch–to–3D animation techniques [39, 45, 236] offer moderate generaliza-

tion, as sketch-based inputs can accommodate diverse drawing styles, while the underlying 3D representations still impose structural and rigging constraints.

Temporal coherence Temporal coherence measures the smoothness and consistency of motion over time. Keyframe-based interpolation methods [132, 154] achieve high temporal coherence by explicitly enforcing smooth transitions between user-defined key poses. Physics-based method [41, 152] offers high temporal coherency by simulating motion under continuous physical properties, creating naturally smooth trajectories over time. Data-driven methods such as MoCap [38] and video-based methods [241] maintain temporal consistency by following subsequent video pose changes, providing moderate coherency. Text-driven animation systems [4, 52, 96] provide moderate coherence, depending on the underlying model's capacity (text-to-video diffusion) to maintain temporal structure across frames. Finally, 2D sketch–to–3D model animation techniques [195, 221] offer moderate temporal coherence, as the 3D model animation relies on data-driven motion sources.

Realism Realism evaluates the perceptual quality of animation, reflecting how closely the generated motion resembles real-world behavior. Keyframe-based animation [154, 240] achieves high realism with well-placed keyframes and motion trajectories. Physics-based animation techniques [60, 174] typically produce highly realistic motion by explicitly simulating physical interactions, forces, and material properties, ensuring that movement adheres to real-world dynamics. Data-driven approaches, including MoCap [164] and video-driven methods [210], achieve high realism by often using motion patterns derived from captured or annotated real-world data. Text-driven animation methods [114, 205] generally offer moderate realism, which depends on the specificity and quality of textual prompts as well as the expressive capacity of the underlying generative models. Finally, 2D sketch–to–3D model animation techniques [235] provide moderate realism, as the mapping from abstract sketches to 3D geometry often relies on simplified rigs and predefined motion templates, limiting the fidelity of fine-grained motion details and physical interactions.

User effort This attribute reflects the technical effort and cognitive load required to create animations. Keyframe-based [127, 212] methods require key poses as input, resulting in moderate overall effort despite the technical complexity of pose design and timing control. Physics-based methods [105, 115] are more complex to maintain the physical behavior of the sketch motion. Data-driven methods, including MoCap [31, 164] and video-driven techniques [210] require moderate user effort due to extensive data preprocessing, feature extraction, and motion-mapping pipelines. Text-driven systems [52, 117, 119], conceptually simple to use, depend on generative models, and generate text-

aligned animation without any complexity. Finally, 2D sketch–to–3D model animation techniques [39, 163] require moderate user efforts, as they balance intuitive sketch-based interaction with the additional effort of 3D reconstruction, rigging, and motion transfer.

Flexibility Flexibility assesses how well an animation method adapts to different use cases, motion styles, or character designs. Keyframe-based interpolation technique [16] offers moderate flexibility, allowing artists to customize timing and poses within a constrained structure. In contrast, Physics-based approaches [238] are less flexible, as they are bound by rule-based motion generation and parameter constraints. In comparison, MoCap [138] and video-driven methods [218] demonstrate moderate flexibility, performing well within the scope of captured or learned motions but showing limited adaptability to unseen styles or novel character configurations. Text-driven systems [36, 114] are highly flexible through prompt-based variety, allowing a wide range of animations to be generated from text descriptions. Finally, 2D sketch–to–3D model animation techniques [236] generally offer low flexibility, as they rely on predefined rigging structures and motion mappings that restrict adaptation to significantly different characters or animation styles.

Speed Speed measures the time required by an animator or system to generate a usable animation output. Keyframe-based animation methods [196] achieve moderate speed, as interpolation between manually defined poses accelerates motion generation but remains constrained by the amount of user input and editing time. In contrast, physics-based methods [152] are computationally intensive, as they rely on continuous physical simulations and rule-based solvers, leading to slower performance. Data-driven approaches (MoCap and video-driven) [164, 170] are moderate, as motion data is extracted and transferred to the sketch. At the same time, text-driven animation [36, 119] uses a text-to-video diffusion model, which requires significant training time to estimate motion and align it to animate the sketch, resulting in low speed. Finally, 2D sketch–to–3D model animation techniques [221] also exhibit low speed, due to the additional cost of 3D reconstruction, occlusion handling, rigging, and motion mapping.

Expressiveness Expressiveness captures the dynamic, stylistic richness, or emotional variation that an animation method can convey. Keyframe-based interpolation systems [16, 154] enable high expressiveness, as animators can manually compose exaggerated poses, subtle timing, and stylized motion arcs. Physics-based methods [105, 238] achieve high expressiveness by simulating natural secondary dynamics, such as squash, stretch, and fluid-like deformations, enabling visually rich motion effects. Data-driven methods, such as MoCap [38],

have moderate expressiveness, as they mainly replicate physical motion without much abstraction, whereas video-driven methods [142, 170] offer low expressiveness, as they primarily depend on video motion. In contrast, text-driven approaches [52, 205] often exhibit limited expressiveness, as generative models prioritize semantic accuracy and motion plausibility at the expense of stylistic diversity, thereby complicating fine-grained artistic control. Finally, 2D sketch–to–3D model animation techniques [45, 235] offer moderate expressiveness, as secondary motions and stylized deformations can be incorporated, but are constrained by predefined rigs, motion templates, and the fidelity of sketch-to-3D reconstruction.

Abstraction level Abstraction refers to the symbolic or stylized characteristic of an animation, reflecting how far the motion departs from real-world behavior. Keyframe-based interpolation methods [48, 132] exhibit low abstraction, as motion is typically represented through explicit pose-to-pose transitions with limited stylistic deviation. Physics-based methods [41, 105] demonstrate higher abstraction, as physical simulations often produce non-literal or exaggerated effects driven by material properties and fluid-like dynamics. Data-driven techniques such as MoCap systems [31] exhibit the least abstraction, replicating real-world motion with minimal interpretive variation, while video-driven methods [241] provide moderate abstraction through motion stylization and representation preferences during transfer. In contrast, text-driven animation [4, 52, 114, 119] supports high abstraction, capable of generating motion for the different levels of input sketch abstraction based on the input prompt. Finally, 2D sketch–to–3D model animation techniques [39] typically exhibit low abstraction, as they rely on predefined 3D structures and motion mappings that constrain stylization and deviation from representation.

2.4.7 Datasets and evaluation metrics

Sketch datasets

Over the years, several datasets have been introduced to advance research on sketch understanding, each focusing on different aspects of sketch representation, as shown in Table 2.3. The TU-Berlin dataset [40] is widely used and contains 20,000 samples across 250 categories. It captures arbitrary objects via static 2D representations, making it useful for sketch classification and retrieval tasks, but it lacks stroke-sequence information. Another dataset, QuickDraw [66], provides over 50 million vector sketches with temporal stroke data, offering significant opportunities for large-scale recognition, sequence modeling, and generative tasks. However, the quick-drawing constraint results in noisy and often oversimplified sketches. To address fine-grained understanding, SketchSeg-10K [187] introduces fine-grained and open-source pixel-wise segmented sketches for part-level recogni-

tion, although its category diversity remains limited. It contains 10,000 samples across 10 classes. SketchFix-160 [150] pairs noisy sketches with their clean counterparts for sketch correction tasks, but suffers from a small sample size. It contains 3904 sketch samples across 160 categories. The Sketchy dataset [148] bridges sketches and photographs, enabling research in cross-modal retrieval like sketch-based image retrieval. Although imperfect alignment between sketches and photos limits its effectiveness, it has 125 categories and 75,471 sketches of 12,500 objects. The SketchyScene dataset [243] addresses the lack of richly annotated scene-level sketches, particularly for training and evaluating models that understand or generate structured sketches. It contains around 29,056 scene-level sketches, 7000+ pairs of scene templates and photos, and 11000+ object sketches. Animated Drawings [164] contains pixel-based hand-drawn sketch characters useful for animation tasks moving toward dynamic sketch representations. It contains over 178,000 amateur drawings of human figures, since it is limited to bipedal, and most sketches are noisy.

Despite these advances, most datasets operate primarily in 2D, and 3D-aware sketch datasets remain an open research frontier. These datasets outline a clear transition from static to temporal, dynamic sketches, broadening the landscape of sketch-based modeling and generation.

Table 2.3: List of sketch datasets.

Dataset	**Representation**	**Sketch samples**	**Category**	**2D/3D**	**Input type**	
					Pose	**Arbitrary**
TU-Berlin [40]	Stroke	20K	250	2D Object		✓
Quickdraw [66]	Stroke	50M+	345	2D Object		✓
SketchSeg-150K [140]	Stroke	150K	20	2D Object	✓	
SketchFix-160 [150]	Stroke	3904	160	2D Object	✓	
Sketchy [148]	Stroke	75K	125	2D Object		✓
SketchyScene [243]	Pixel	29K	–	2D Scene	✓	
Amateur Drawings [164]	Pixel	177K+	–	2D Object	✓	

Evaluation metrics

The evaluation of the sketch animation quality captures not only the fidelity of individual frames but also the temporal consistency, smoothness, and perceptual features of the animation sequence. Sketch animation models require assessing the visual quality of generated sequences and the temporal coherence of motion across frames. Overall, the practical evaluation of sketch animation requires a balanced combination of evaluation metrics that offer reproducible quantitative analysis and subjective human evaluations, which capture the perceptual qualities essential for the practical use case. The evaluation matrices for sketch animation are categorized based on image/frame quality, stroke/contour accuracy, temporal consistency, and animation-specific characteristics.

Image/Frame quality metrics

These metrics evaluate the quality of individual frames within an animation, often by comparing a generated frame $\hat{I} \in \mathbb{R}^{h \times w}$ to a ground-truth reference image $I \in \mathbb{R}^{h \times w}$.

Fréchet inception distance (FID) FID metric [71] measures the distribution similarity between real and generated sketch sequences. A lower FID score indicates that the generated sketches are more similar to real sketches in terms of visual quality. It evaluates the overall realism of the generated frames but does not directly assess temporal smoothness across them.

Peak signal-to-noise ratio (PSNR) PSNR [56] measures the ratio between the maximum possible strength of a signal and the strength of corrupting noise. In image evaluation, it quantifies the difference between two images at the pixel level. A higher PSNR value indicates a more accurate reconstruction, indicating the generated frame is closer to the ground-truth image.

Structural similarity index (SSIM) Unlike PSNR, which only considers pixel-level differences, SSIM [193] evaluates the structural similarity between two images. It considers illumination, contrast, and structure, making it a better predictor of perceived image quality. A score closer to 1.0 indicates higher similarity.

Learned perceptual image patch similarity (LPIPS) LPIPS [228] measures the distance between feature representations of two images extracted from a pre-trained neural network (like VGG [161] or AlexNet [102]). By comparing images in perceptual space, LPIPS better captures human similarity estimation, as it is less sensitive to minor pixel shifts and more focused on overall visual content.

Stroke/Contour accuracy

These metrics are specific to the unique characteristics of sketches, focusing on the accuracy of the lines and contours. Let the ground-truth stroke set be denoted by $S = (s_i)_{i=1}^{N}, s \in \mathbb{R}^2$ and the generated stroke set by $\hat{S} = (\hat{s}_j)_{j=1}^{M}, \hat{s} \in \mathbb{R}^2$.

Chamfer distance (CD) Chamfer Distance [9] estimates the average closest distance between two sets of points or curves. In sketch animation, it is used to

evaluate how closely a set of generated strokes matches the ground-truth stroke set. A lower CD value indicates a better match.

$$\mathrm{CD}(S, \hat{S}) = \frac{1}{|S|} \sum_{s \in S} \min_{\hat{s} \in \hat{S}} \|s - \hat{s}\|_2 + \frac{1}{|\hat{S}|} \sum_{\hat{s} \in \hat{S}} \min_{s \in S} \|\hat{s} - s\|_2. \tag{2.1}$$

Hausdorff distance (HD) Hausdorff Distance [81] estimates the maximum distance of a point in one set to the nearest point in the other set. Its high sensitivity to outliers and large deviations makes it a conservative evaluation metric; a low Hausdorff distance guarantees high overall accuracy with no significant structural errors.

$$\mathrm{HD}(S, \hat{S}) = \max \left\{ \max_{s \in S} \min_{\hat{s} \in \hat{S}} \|s - \hat{s}\|_2, \ \max_{\hat{s} \in \hat{S}} \min_{s \in S} \|\hat{s} - s\|_2 \right\}. \tag{2.2}$$

Temporal consistency

Temporal coherence evaluates the smoothness and consistency of the animation sequence. It assures the drastic pose change between consecutive frames that helps to avoid jittery or abrupt transitions. Temporal consistency metrics are essential for evaluating smooth, non-jitter, and realistic animation flows from one frame to the next.

Temporal warping error Temporal warping measures how well a frame $\hat{I}_{t+1}$ aligns with the previous frame $\hat{I}_t$ when warped using the estimated optical flow [120]. A lower warping error implies smoother transitions and reduced temporal jitter.

$$\mathcal{L}_{\text{warp}} = \frac{1}{T-1} \sum_{t=1}^{T-1} \frac{1}{h \cdot w} \left\| \hat{I}_t - W(\hat{I}_{t+1}, \hat{\mathbf{F}}_t) \right\|_2^2, \tag{2.3}$$

where $\mathcal{L}_{\text{warp}}$ is the warping error, $\hat{F}_t$ denotes optical flow from $\hat{I}_t$ to $\hat{I}_{t+1}$, T denotes the total number of frames in the video sequence and $W(\hat{I}_{t+1}, \hat{\mathbf{F}}_t)$ represents backward warping of frame $\hat{I}_{t+1}$ using the estimated flow.

Fréchet video distance (FVD) Similar to FID, FVD [184] is a metric for evaluating the realism of video sequences. It measures the distance between the feature distributions of real and generated videos. A low FVD score indicates that the generated video sequence is visually and temporally realistic, capturing the actual dynamics of real motion.

Animation-specific quality metrics

Sketch-to-video consistency Sketch-to-video consistency [52] is primarily used for text-driven sketch animation. It estimates the consistency of the generated sketch animation with the input sketch. It uses the CLIP [141] model and estimates the cosine similarity between the input and generated animated sketches. Higher values indicate stronger visual consistency between the input sketch and the generated animation.

$$\mathcal{S}_{\text{SV}} = \frac{1}{T}\sum_{t=1}^{T}\frac{\langle \phi_{\text{img}}(I_{\text{ref}}), \phi_{\text{img}}(\hat{I}_t)\rangle}{\|\phi_{\text{img}}(I_{\text{ref}})\|_2 \|\phi_{\text{img}}(\hat{I}_t)\|_2}, \tag{2.4}$$

where $\mathcal{S}_{\text{SV}}$ is sketch-to-video consistency score and $\langle \cdot \rangle$ denotes the inner product (dot product). I_{ref} denotes the input reference sketch, $\hat{I}_t$ denotes the generated sketch animation at time t, T denotes the total number of frames in the generated sketch animation, and $\phi_{img}(\cdot)$ denotes the CLIP image encoder.

Text-to-video alignment Text-to-video alignment [52] estimates the alignment of the generated sketch animation to the input text prompt $\mathcal{T}_{text}$. It uses the X-CLIP [133] model, trained on video recognition, to estimate the alignment score. Higher alignment scores indicate better semantic correspondence between the text prompt and the generated sketch animation.

$$\mathcal{S}_{\text{TV}} = \frac{\left\langle \phi_{\text{text}}(\mathcal{T}_{\text{text}}), \phi_{\text{vid}}(\{\hat{I}_t\}_{t=1}^{T})\right\rangle}{\|\phi_{\text{text}}(\mathcal{T}_{\text{text}})\|_2 \|\phi_{\text{vid}}(\{\hat{I}_t\}_{t=1}^{T})\|_2}, \tag{2.5}$$

where $\mathcal{S}_{\text{TV}}$ is text-to-video alignment score, ϕ_{text} denotes the input text prompt, $\phi_{vid}(\cdot)$ and and $\phi_{text}(\cdot)$ denotes the X-CLIP video and text encoder, respectively.

User/subjective evaluation

User evaluations are essential for assessing sketch animations because their quality is determined by human perception. It is especially true for sketches, where the goal is often to capture a specific sense or style rather than just technical perfection. In user evaluation, participants are asked to provide feedback for the generated animations based on quality, realism, and consistency. It provides a perceptual validation that automatic metrics may fail to capture fully, especially for subtle attributes like style, expression, and creative interpretation. Since sketches are highly perceptual, user studies are often required:

Mean opinion score Mean Opinion Score (MOS) is a widely used method for getting human ratings on different aspects of an animation. Participants are typically shown a series of animations and asked to rate them on a scale, often

from 1 (poor) to 5 (excellent), for characteristics such as smoothness, quality, and plausibility of motion. The scores from all participants are averaged to get a single, quantifiable MOS for each characteristic, providing insights into how well an animation performs from the user's perspective.

Pairwise preference tests Pairwise preference tests are another powerful way to evaluate animations, especially when comparing two different methods or techniques. Instead of rating animations individually, participants are shown two animations side-by-side and asked to choose which one they prefer, such as, "which animation of a running character looks more natural?" This method is particularly effective because it simplifies the task for the participant. They only have to choose between two options rather than assign a numerical score. Calculating the choices across all participants can determine which method is consistently preferred and by how much, providing a robust comparison between techniques.

Artist evaluation Artist evaluation is crucial for assessing how well an animation tool or technique meets the needs of its intended users. Professional animators, who understand the properties of motion and character performance, are asked to evaluate the system on a variety of factors, including:

- *Usability:* How easy and intuitive the tool is to use. Can an artist quickly create what they envision?
- *Visual Appeal:* Does the tool allow for creating aesthetically pleasing and expressive animations?
- *Workflow Integration:* Can the technique be easily integrated into a professional animation pipeline?

This type of feedback is invaluable because it comes from experts who can identify subtle issues. Their insights can directly inform the development of tools that are not only technically sound but also practical and inspiring for the creative professionals who will use them.

Overall, the practical evaluation of sketch animation requires a balanced combination of evaluation metrics that offer reproducible quantitative analysis and subjective human evaluations, which capture the perceptual qualities essential for the practical use case.

2.4.8 Recent trends

Generative animation Recent animation methods have seen a significant rise in the use of generative models, particularly diffusion models [12, 73, 167].

Previous works relied on Variational Autoencoders (VAEs) [99] and Generative Adversarial Networks (GANs) [57]. The diffusion model provides an edge by facilitating superior image fidelity and stability during training. It is used for sketch synthesis and animation, offering a more flexible and robust way to generate high-quality outputs from sparse or ambiguous sketch sequences.

Cross-model conditioning has recently been highly explored; generative models use text, video, or pose with the sketch to generate plausible animations. Language-guided sketch animation [52, 119] enables users to generate sketch animations from text prompts. On the other hand, video-driven models [75, 218] animate characters based on keypoint motion or optical flow. Using foundation models like CLIP [141] and large-scale text-to-image transformers has made such multimodal control more accessible and effective.

Temporal consistency Another emerging trend is using AI models to generate temporally consistent animations. Deep learning-based architectures use temporal conditioning [76, 134] to ensure smooth and continuous motion for visual coherency. The temporal coherency in these AI models is conditioned on sketch keyframes or motion cues.

3D awareness and depth estimation 3D awareness and depth integration are used to make sketch-based animations more immersive. 2D sketches with depth cues or 3D prior (camera parameters) integration [170] generate an animation that is better suited for augmented reality (AR), virtual reality (VR), and metaverse applications [45, 163]. These models help bridge the gap between artistic 2D inputs and real-world spatial dynamics.

Learning from sparse data In recent years, researchers have increasingly focused on learning from sparse datasets [126], leading to the adoption of few-shot learning techniques to extend sketch animation models to novel characters or artistic styles with limited training examples. These techniques [154] are especially valuable given the independency of large-scale, high-quality sketch animation datasets.

Chapter 3

Literature review

Animating visual content, whether photorealistic imagery, hand-drawn sketches, or 3D models, requires addressing three recurring challenges: synthesizing plausible motion under large pose changes, preserving structural coherence, and enabling controllable, physically plausible dynamics. The difficulty of these challenges mainly depends on the motion source and the source representation (photograph or sketch). Accordingly, this chapter organizes the existing literature along these two axes, as they also reflect the progression of the technical contributions presented in this thesis. This organizing principle is introduced explicitly at the beginning because many existing approaches can be categorized in multiple ways. Section 3.1 presents a taxonomy of video-driven animation systems, while Section 3.2 discusses the generative models and priors used in this research. Sections 3.3 through 3.6, then review the four resulting problem settings in the same sequence as the subsequent thesis chapters. Each section concludes by identifying the key limitations of existing methods, thereby motivating the corresponding technical contributions presented in the following chapters.

3.1 Taxonomy of video-driven animation systems

The literature review in this chapter is organized according to the pairing of the motion source and the motion model, as summarized in Table 3.1. The motion source determines the form of supervision available for animation, such as videos or text prompts.

Table 3.1: Mapping of the literature review to their corresponding motion sources, motion models, and source representations.

Review section	**Motion source**	**Motion model**	**Source representation**	**Chapter**
3.3 Video-driven image animation	Driving video	Single-domain	Image	Chapter 4
3.4 Video-driven sketch animation	Driving video	Cross-domain	Raster sketch	Chapter 5
3.5 Text-driven sketch animation	Text prompt	Diffusion prior	Vector sketch	Chapter 6
3.6 Text-driven 3D animation	Text prompt	Diffusion prior	3D Mesh	Chapter 7

A driving video provides temporally dense observations of motion, but offers limited high-level semantic understanding. On the other hand, a text prompt provides semantic understanding but does not explicitly specify correspondence in spatial or temporal motion. In this context, explicit means that the motion source directly specifies a mapping between a particular source motion and a target structure. A text prompt does not provide this mapping. For example, the prompt "a character walks forward" describes the desired action semantically but does not specify which joints should move, their trajectories, or their timing. A raster image (photograph) provides dense appearance cues, including texture, color, and local visual features, which can facilitate motion estimation and correspondence. However, these cues must be preserved or synthesized consistently under deformation and large pose changes. In contrast, a vector sketch provides explicit geometric structure through contours and strokes but contains substantially less appearance information. Consequently, motion transfer from a driving video to a sketch relies on structural correspondence between these and the interpretation of sparse geometric cues. Consequently, methods designed for one setting cannot be directly transferred to the other. These limitations motivate the progression of the research presented in this thesis and are examined empirically in Chapters 4 and 5.

Within video-driven animation, existing motion models can be broadly categorized into two classes based on whether the source and driving inputs belong to the same or different visual domains.

Single-Domain Motion Model Single-Domain Motion Models (SDMMs) focus on transferring motion within a unified visual domain, in which models are typically trained within constrained distributions, implicitly learn domain-specific correlations between appearance and motion. These methods typically rely on domain-specific motion representations, such as 3D facial landmarks, dense deformation fields, or canonical warping functions, that remain stable across all subjects within the domain. This structural uniformity enables the use of direct pixel warping, appearance-preserving GANs, or diffusion-based motion priors to achieve high-fidelity reenactment while accurately maintaining identity, pose, and temporal smoothness. As a result, single-domain methods have evolved significantly, exhibiting robust performance on tasks such as talking-head generation, portrait animation, and dance reenactment. Their training pipelines are also relatively concise, requiring only homogeneous video datasets and domain-specific discriminators, which further strengthens their realism and stability.

In SDMM, where structural consistency and appearance distributions are homogeneous, different generative architectures exhibit distinct resilience across functional attributes. GAN-based SDMMs are highly effective for real-time or interactive applications due to their low computational cost and fast infer-

ence, while also achieving strong realism through domain-specific discriminators trained on consistent facial or human-body appearance distributions. However, their temporal consistency is moderately delicate, often requiring explicit temporal discriminators or recurrent modules. Diffusion-based SDMMs, by contrast, deliver superior realism, strong temporal coherence, and high robustness to occlusions or fast motion, owing to denoising processes that implicitly regularize content across frames. This comes at the cost of high computational complexity, which reduces its suitability for real-time or mobile deployment. Transformer-based SDMMs offer excellent temporal modeling and scalability by capturing long-range dependencies across frames. It generalizes well within the domain but requires significant computational resources and training costs. Latent representation approaches, such as autoencoders or motion-appearance disentanglement modules, perform particularly well in SDMM because the latent spaces remain compact and domain-stable, enabling automatic, scalable animation pipelines with minimal user input.

Cross-Domain Motion Model Cross-Domain Motion Models (CDMMs) transfer motion between semantically distinct visual domains, such as photographs and cartoons, images and sketches, or object categories with different topologies. Their defining characteristic is the ability to decouple motion from appearance, enabling the transfer of high-level motion dynamics rather than raw geometric displacement fields. In contrast to single-domain models, which are trained on constrained data distributions and implicitly learn domain-specific correlations between appearance and motion, cross-domain models must generalize across substantial differences in structure, texture, and motion semantics. Consequently, they typically adopt one or more of three strategies: (i) representing motion using skeletal or keypoint abstractions that are independent of appearance, (ii) learning disentangled latent spaces in which motion and identity are encoded separately, or (iii) establishing geometric correspondence through shape matching between the source and target. The third strategy forms the basis of the approach presented in Chapter 5, in which motion is transferred by establishing a correspondence between a driving skeleton and a sketch outline rather than learning it from paired training data. The relative strengths and limitations of single-domain and cross-domain motion models, together with the failure of single-domain approaches under domain shift, are examined empirically in Section 3.3.

3.2 Generative priors for animation

Modern animation systems replace manual keyframing, motion-capture retargeting, and procedural animation with generative models that learn motion and appearance distributions directly from data. Among these, diffusion mod-

els [73, 139, 167] have become the dominant paradigm, learning to reverse a noising process through iterative denoising and achieving state-of-the-art results in image synthesis, video generation, and cross-modal content creation. Their iterative formulation supports conditional generation, enabling guidance from keypoints, segmentation masks, or textual prompts. Here, the discussion focuses exclusively on the role of diffusion models as motion priors.

Two distinct uses of diffusion models recur in the animation literature. In the first, the diffusion model acts as a *generative model* that directly synthesizes image or video frames, as exemplified by the video-driven refinement framework presented in Chapter 4. In the second, a pre-trained video diffusion model serves as a *generative prior* that guides the optimization of a non-pixel representation through Score Distillation Sampling (SDS) [139]. In this setting, the diffusion model does not generate pixels explicitly; instead, it provides gradient-based supervision that steers the optimization process. This paradigm underpins the methods used in Chapters 6 and 7, and is discussed in detail in Section 7.2.5.

Diffusion models Diffusion models [73, 139, 167] are a class of generative models that learn to generate data by gradually reversing a diffusion process that adds noise to the data. In the forward diffusion process, a clean data sample $x_0 \sim p_{\text{data}}(x)$ is progressively eroded with Gaussian noise over T time steps, producing a sequence $\{x_1, x_2, \ldots, x_T\}$. This forward process is defined as a Markov chain with fixed variance schedule β_t, such that

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}\, x_{t-1},\, \beta_t \mathbf{I}),$$

where each step adds a small amount of Gaussian noise, and $\mathbf{I}$ is the identity matrix. By the final step T, x_T becomes nearly isotropic Gaussian noise, destroying the structure of the original data. The goal of the diffusion model is to learn the reverse process, parameterized by a neural network θ, which denoises x_T step by step to recover a high-quality sample $\hat{x}_0$. The reverse process is modeled as

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t),\, \Sigma_\theta(x_t, t)),$$

where μ_θ and Σ_θ are learned functions predicting the mean and variance of the reverse Gaussian at each step.

Training a diffusion model typically involves optimizing a variational bound on the negative log-likelihood of the data, which reduces to a simple denoising score-matching objective. Specifically, a loss function is defined as

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{x_0, \epsilon\sim\mathcal{N}(0,\mathbf{I}), t}\Big[\big\|\epsilon - \epsilon_\theta(x_t, t)\big\|^2\Big],$$

where $\epsilon_\theta(x_t, t)$ predicts the noise added at step t, and

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \text{with} \quad \bar{\alpha}_t = \prod_{s=1}^{t} (1-\beta_s).$$

By minimizing this loss, the model learns to iteratively remove noise from a Gaussian sample, generating realistic data that matches the target distribution. Diffusion models have demonstrated state-of-the-art performance in image synthesis, video generation, text-to-image generation, and cross-modal content creation due to their stable training and ability to produce high-fidelity, diverse samples, making them a powerful alternative to GAN-based approaches. The iterative denoising process naturally supports conditional generation, enabling structural guidance through keypoints, segmentation, or textual prompts.

3.3 Video-driven image animation

Image animation and video retargeting have gained widespread attention in recent times. Image or object animation poses a challenge similar to the translation problem across different visual domains. Traditional image animation methods are limited to domain-specific, focusing on areas like the human face [37, 67, 118, 175, 181, 197] and human body [20, 122, 146, 224]. Many of these data-driven methods [20, 54, 135, 224, 242] require supervision from facial landmarks and human pose silhouettes, offering predefined structures but are limited to specific object categories.

Video-driven image animation transfers motion extracted from a driving video to a static source image, synthesizing a temporally coherent animation. In recent years, there has been the emergence of deep learning methods [155, 156, 159, 177, 178, 229] that are not restricted to specific objects and do not rely on labeled data. These methods are applied to arbitrary object categories without prior structure information. For instance, MonkeyNet [155], a self-supervised method, uses a learned, unsupervised, keypoint-based dense motion field to transfer motion from driving videos to images. However, it struggles with large motions in driving videos. FOMM proposed by [156] improves motion transfer quality by adding local affine transformations to the learned keypoints, yet it fails to identify meaningful object regions, locations, and shapes. To overcome this issue, MRAA [159] predicts meaningful regions using affine metrics and transfers dense motion fields to those regions. It also uses a background motion predictor to neglect camera background motion in driving videos. However, MRAA faces challenges with large pose gaps, non-linear complex motions, and rigid motion. Zhao et al. [229] propose a thin-plate spline (TPS) motion estimation method to handle non-linear complex motions and a multi-resolution occlusion mask to prevent occlusion. TPS encounters challenges with non-linear, complex motions, but struggles to accurately estimate smooth motion across large pose gaps. DAM [178] uses a deformable anchor model for motion estimation to identify object structure information, but it struggles with occlusion and fails to align the object shape with the exemplar video object. MT [177] uses a motion transformer that encodes shape tokens from the patch feature of

the input and a motion token that encodes the motion information. Chain-of-Pose (CoP) [49] estimates the intermediate pose from the large pose changes and uses an appearance refinement model to preserve the appearance and structure information. LIA [191, 192] propose an autoencoder-based network that uses a linear motion decomposition approach to model video motion in the source image. All the above methods face the challenge of motion estimation when there are drastic pose changes in the video. Chapter 4 handles the problem of large pose problems and overcomes the unnatural artifacts and shape distortion in the generated animation video.

Open problems (G1) Despite their success, the methods reviewed above share two main limitations. First, motion estimation degrades under large pose variations, leading to unnatural artifacts and geometric distortions because the estimated warping field relies on keypoint configurations that become unreliable during drastic pose change. Chapter 4 handles this issue with a motion condition network integrated with SDMM. Second, and more importantly for this thesis, these methods are inherently single-domain. They assume the availability of dense photometric information for both motion estimation and the synthesis of disoccluded regions. Neither assumption holds for sparse line drawings or hand-drawn sketches. As demonstrated in Chapter 4, directly applying such methods to sketches leads to substantial performance degradation. This negative result motivates the transition from appearance-driven motion transfer to the geometry-aware modeling framework introduced in Chapter 5.

3.4 Video-driven sketch animation

Transferring motion from a driving video to a sketch presents challenges not found in image-based animation. These include geometric deformation without texture cues, partial and self-occlusion, in-plane and out-of-plane rotations, and the lack of dense correspondence between the driving video and the target sketch. A comprehensive review of this problem setting is provided in Chapter 2; here, the discussion is limited to the methods most directly relevant to the approach introduced in Chapter 5.

Significant progress has been made in sketch animation in recent years, with methods that introduce dynamic motion into static sketches. Early work by Bregler et al. [14] proposed a keyframe-based technique that tracks motion in cartoon sequences and transfers it to sketches or line drawings. Further, Okabe et al. [136] animated fluid phenomena using exemplar videos; however, their approach is restricted to fluid animation and incurs a high computational cost. To address this issue, Santosa et al. [149] used optical flow for motion transfer, but their method requires the input sketch to have a structure closely matching that of the exemplar video. Several approaches introduce predefined mo-

tion effects rather than transferring arbitrary motion. For example, Xing and Kazi et al. [211] synthesized animations involving fire, smoke, and water, although their method is limited to a small set of predefined animation styles. Video-driven methods [7, 69, 138] transfer motion from reference videos but require substantial manual intervention throughout the animation process. Among them, TraceMove [138] is a data-driven framework that uses a reference video to guide sketching and a motion-capture (MoCap) dataset [101] to match skeletons and generate intermediate keyframes. Similarly, Ben-Zvi et al. [10] proposed a data-driven approach that focuses on transferring drawing style rather than extracting and transferring motion trajectories from the reference video. To reduce manual interaction, several methods exploit domain-specific priors. Facial animation approaches [17, 181] rely on facial landmarks, but their applicability is limited to facial expressions. Similarly, methods based on motion-capture data [33, 194] are generally restricted to articulated human bodies and do not generalize to arbitrary objects. To overcome this issue, Live-Sketch [170] introduces a generalized approach that extracts sparse motion trajectories from a driving video and transfers them to a sketch via control-point tracking, combined with mesh deformation, to reduce distortion. However, it still requires manual user interaction and struggles to estimate motion reliably in smooth or textureless regions. More recently, Smith et al. [164] presented a video-driven framework for animating hand-drawn characters. Although effective for bipedal characters, the method remains limited by inaccurate motion estimation in smooth regions and does not readily generalize to more diverse object categories.

Open problems (G2) Despite the progress reviewed above, three fundamental limitations remain and are addressed in Chapter 5. First, many existing methods require manual user intervention at multiple stages of the animation pipeline. Second, most approaches are restricted to a single object category, typically the human face or body, limiting their applicability to more diverse subjects. Third, existing methods struggle to handle self-occlusion, in which one body part passes in front of another, and the correct depth ordering must be inferred in the absence of appearance cues. Chapter 5 addresses these challenges through a unified framework that integrates motion extraction, motion transfer, and self-occlusion logic within a single pipeline. The proposed method generalizes across multiple object categories, including bipeds, quadrupeds, and inanimate objects, and supports motion transfer between semantically distinct categories.

3.5 Text-driven sketch animation

Replacing the driving video with a textual description removes the need for a reference motion sequence and provides controllability. Consequently, motion

must be inferred from a learned generative prior capable of generating a diverse range of motion. A comprehensive review of this problem setting is presented in Chapter 2; the discussion here focuses only on the methods most relevant to the work presented in Chapter 6.

Text-to-video generation aims to automatically generate a corresponding video from a given textual description. Early approaches demonstrated the potential of generative adversarial networks (GANs) [113, 182, 239] and autoregressive transformers [204, 217] for video generation. However, these methods were generally limited to fixed domains and struggled to generate diverse, high-quality videos with long-range temporal consistency. Recent advances in diffusion models have substantially improved text-to-video generation. Methods such as [22, 25, 63, 188, 234] utilize Stable Diffusion [134] by incorporating temporal modeling in the latent space, enabling coherent video synthesis from textual prompts. DynamicCrafter [213] further extends this paradigm by generating videos conditioned jointly on an input image and a text prompt. Despite these advances, open-source text-to-video models still face challenges in maintaining semantic consistency and preserving fine structural details throughout long motion sequences. Closely related to text-to-video generation is the image-to-video task, which seeks to animate a single input image according to a textual description. In this direction, Latent Motion Diffusion (LMD) [77] estimates motion by learning optical flow from video sequences and employs a 3D U-Net diffusion model to synthesize animated videos. Further, Make-It-Move [78] adopts an encoder-decoder architecture conditioned on both the input image and text prompt to generate video sequences. Similarly, VideoCrafter1 [22] and LivePhoto [25] preserve the structure and appearance of the input image by conditioning the diffusion model on both image and text inputs. CoDi [176] further generalizes this framework by learning a shared latent representation across multiple modalities, including images, videos, text, and audio. The emergence of diffusion models has also enabled their use as optimization priors rather than direct generators. DreamFusion [139] introduced Score Distillation Sampling (SDS), which enables the optimization of 3D representations using the supervision provided by a pre-trained 2D image diffusion model. Although the SDS objective is derived from the diffusion denoising loss, it omits the U-Net Jacobian during optimization, thereby avoiding the high computational cost of backpropagating through the diffusion model while still guiding the optimization toward samples consistent with the text prompt. SDS loss is also used to optimize the other generative tasks, such as sketches [214], vector graphics [84], and meshes [24]. Building on this paradigm, recent text-driven sketch animation methods optimize the positions of sketch primitives over time and use SDS to align the generated motion with the textual description. Rather than directly synthesizing video frames, the diffusion model operates as a generative prior that provides gradient-based supervision, enabling semantically consistent sketch animation. This property makes diffusion priors applicable to both the vector-stroke

representation used in Chapter 6 and the skeletal parameterization introduced in Chapter 7.

In this direction, LiveSketch [52] presents a training-free framework for animating vector sketches using a pre-trained text-to-video diffusion model. Motion is induced through Score Distillation Sampling (SDS), which optimizes the per-frame displacements of stroke control points under the guidance of the diffusion prior. Because the animation is represented by deforming the original control points rather than synthesizing new pixels, the identity and topology of the vector strokes are preserved throughout the sequence.

Open problems (G3) Directly optimizing vector control points with respect to a diffusion prior leaves the underlying geometry largely unconstrained. The diffusion prior provides no explicit inducement to preserve stroke length, local rigidity, or geometric consistency over time. Moreover, the limited local control of cubic Bézier curves means that even small perturbations of their control points can produce disproportionately large changes in curve shape. Consequently, the resulting animations often exhibit temporal inconsistency and progressive geometric distortion. Chapter 6 addresses these limitations for skeleton-free sparse sketches by introducing explicit geometric constraints that preserve stroke length, swept area, and local rigidity throughout the optimization process.

3.6 Text-driven 3D animation

3D animation has recently seen a paradigm shift driven by the integration of large-scale diffusion priors and unified representation learning. While traditional methods relied heavily on manual rigging or expensive motion capture data. A significant challenge in 3D animation is the lack of high-quality 3D motion data compared to the abundance of 2D video. Consequently, recent works focus on distilling motion priors from pre-trained video diffusion models to drive 3D synthesis. Zhang et al. [227] propose MotionDiffuse, a text-driven approach that learns human skeleton pose with a diffusion model to generate the text-guided 3D animation sequence. To enable the arbitrary object animation, Xie et al. [209] introduce AnimaMimic, which animates static 3D meshes by automatically synthesizing a monocular video to construct a skeleton and skinning weights. It then refines the animation using differentiable rendering and physically grounded soft-tissue simulation. Similarly, Li et al. [110] present Articulated Kinematics Distillation (AKD), which integrates video priors with structural controls. By applying SDS directly to a skeleton-based representation, AKD drastically reduces the degrees of freedom to joint-level controls, distilling complex motions while preserving structural consistency. Expanding this control to arbitrary topologies, Animax [80] bridges video priors with skeleton-based animation by representing 3D motion as multi-view, multi-frame 2D pose maps. Using a

joint video-pose diffusion model, it triangulates these sequences into 3D joints and applies inverse kinematics to animate diverse meshes. To address motion quality, Wang et al. [189] propose BiMotion, a novel approach for text-guided 3D character animation that represents motion as continuous B-spline curves rather than discrete frame sequences. This formulation enables smoother, more coherent, and variable-length motions while improving alignment with textual descriptions. To reduce reliance on the training dataset, Su et al. [172] introduce Animus3D, which uses Motion Score Distillation (MSD). This technique improves upon standard SDS by specifically targeting motion dynamics, resulting in smoother and more coherent animations driven by text prompts. In this direction, Wu et al. [206] introduce AnimateAnyMesh, a feed-forward 4D foundation model. Unlike optimization-based methods that require per-subject tuning, this model learns a universal motion representation applicable to any mesh topology, significantly reducing inference time. Chapter 7 proposes an approach that avoids intensive training over large datasets and overcomes the limitations of controllable motion generation, introducing physics-based regularization and secondary motion to generate natural motion.

Open problems (G4) Despite recent advances, existing methods exhibit several important limitations. Many rely on large-scale motion datasets for training, assume a predefined skeletal template, or provide only limited control over the generated motion. Moreover, few methods incorporate explicit physical constraints, allowing the optimized motion to violate joint limits, penetrate the ground plane, or exhibit foot sliding. These artifacts are not penalized by the diffusion prior. Chapter 7 addresses these limitations by enabling controllable motion generation without requiring large-scale training data. The proposed framework introduces physics-based regularization and secondary motion modeling to produce biomechanically plausible animations for automatically generated, non-standard skeletal rigs.

3.7 Summary and research gaps

The literature reviewed in this chapter indicates a consistent trade-off between appearance, geometry, and controllability. Methods that use dense appearance achieve high visual fidelity but fail under domain shift because they rely on photometric information absent from sketches. In contrast, methods that use explicit geometric structure generalize more effectively across domains but are unable to synthesize appearance and often depend on manual user intervention or category-specific priors. Finally, methods that distill motion from generative priors provide semantic controllability but lack the geometric and physical constraints driven by explicit structural representations. This analysis identifies key research gaps, each addressed in a subsequent technical chapter of this thesis. Ta-

ble 3.2 summarizes the identified research gaps (G1-G4) and their corresponding solutions in the respective chapters.

Table 3.2: Research gaps identified in this chapter and the corresponding contributions of this thesis.

Gap	**Limitation in existing work**	**Addressed in**
G1	Motion estimation degrades under large pose change, producing shape distortion and unnatural artifacts in video-driven image animation (Section 3.3).	Chapter 4: keypoint regularization and adaptive diffusion-based refinement.
G2	Video-driven sketch animation requires manual intervention, is confined to specific object categories, and struggles with self-occlusion (Section 3.4).	Chapter 5: geometric correspondence with skeletal transfer and depth-ordered occlusion handling, generalizing across categories.
G3	Text-driven vector sketch animation leaves stroke geometry unconstrained, yielding temporal inconsistency and progressive shape distortion (Section 3.5).	Chapter 6: length–area and as-rigid-as-possible geometric constraints.
G4	Text-driven 3D animation depends on large-scale training or standard skeletal templates, and lacks physical plausibility constraints and secondary dynamics (Section 3.6).	Chapter 7: motion-targeted score distillation with physics-inspired regularisation on automatically rigged meshes.

These gaps are closely interconnected, as G1 and G2 demonstrate that appearance-driven and geometry-driven approaches address complementary challenges rather than competing ones, thereby motivating the transition from appearance-based to geometry-aware animation. Similarly, G3 and G4 share a common underlying principle: in both settings, a generative prior provides semantically meaningful motion but offers no inherent geometric or physical guarantees. In each case, the solution is to optimize an explicit representation under appropriate constraints rather than relying solely on pixel-level generation. Specifically, Chapter 6 constrains vector stroke control points, whereas Chapter 7 constrains skeletal parameters. Consequently, the four technical chapters follow a coherent progression: from appearance to geometry, from observed motion to user-specified motion, and ultimately from two-dimensional representations to physically plausible three-dimensional animation.

Chapter 4

LiveImage: Motion Condition Guided Diffusion Model for Video Motion Transfer

4.1 Introduction

Traditional tools for image animation are static and non-interactive, requiring users with proficient artistic skills. Image animation using an exemplar video has been a new wave of research in recent years. Video motion transfer onto images has applications across diverse domains, including movie effects, entertainment videos, and video conferencing. Generating high-quality videos from images poses a challenge because it requires a proper representation of the object and accurate mapping of extracted motion from the video to the target object. Researchers have made significant progress over the past years and proposed various methods [51, 54, 146, 158, 175, 190], such as Generative Adversarial Networks [57] and Variational Autoencoders [99], to generate videos and images. Despite their effectiveness, previous approaches are entirely data-driven, rely on large training datasets, and require data annotation for large-scale datasets. Recent unsupervised methods [155, 156, 159, 177, 178, 229] have drawbacks such as self-occlusion, the mismatch between the source object and driving video, topological artifacts, and shape distortion during motion transfer.

This chapter presents LiveImage, a deep-learning method for animating images using a driving video. Single-domain motion models (where the domain is the same for the source image and driving frame) [155, 156, 159, 229] estimate a dense motion field, which is used to warp the source image to generate animated frames using an inpainting network. First, keypoints and structural information are extracted from the input video and fed into a dense motion network. Further, keypoint regularization is used to estimate meaningful keypoints of the frames and capture essential geometric information. To achieve this, a separation loss is incorporated to address challenges such as keypoint collapse during motion in the video frames. The coarse-grained frames are estimated using a single-domain

motion method with keypoint regularization.

Since coarse-grained frames suffer from shape distortion and unnatural artifacts due to large pose changes, Gaussian blur is applied to them. Motion information is then estimated using a motion condition network (MCN), inspired by Zeng et al. [225]. An appearance encoder is used to estimate appearance features, and a motion encoder with a weight amplitude is used to model motion conditions. The appearance features and motion conditions are then integrated through a convolutional layer. Finally, the diffusion-denoising module [73] addresses distortion, jitter, and artifacts by leveraging motion-condition information and generating the final animated sequence.

The main research contributions of this chapter are as follows: (*i*) A keypoint regularization is utilized in the single-domain motion model to estimate smooth and non-jittery motion, enabling the generation of coarse-grained images without loss of information, and (*ii*) a motion condition network (MCN) to overcome the artifacts and distortions by incorporating motion condition within a diffusion model.

4.2 Preliminaries

This section summarizes the two core components of the single-domain motion model upon which the proposed framework is built: the keypoint detector and the dense motion network. Both components were originally introduced by Siarohin et al. [155] and are described here in sufficient detail to support the subsequent developments.

4.2.1 Keypoint detector

Traditional animation frameworks often rely on manually annotated landmarks, such as facial keypoints or skeletal joints, restricting their applicability to predefined object categories. To overcome this limitation, Siarohin et al. [155] introduce a framework that learns motion-aware structural anchors in a fully self-supervised manner by directly detecting keypoints from video sequences, without requiring any human annotations. The keypoint detector uses a fully convolutional U-Net architecture that maps an input frame to a set of K spatial heatmaps, denoted as $H_k(\mathbf{u})$, where each heatmap represents the probability distribution of a single keypoint at pixel location $\mathbf{u} \in \Omega$. The model computes the keypoint location as the spatial expectation of the corresponding heatmap:

$$\mathbf{p}_k = \sum_{\mathbf{u} \in \Omega} H_k(\mathbf{u})\, \mathbf{u}.$$

To further capture the local geometric structure surrounding each keypoint, the model estimates a covariance matrix Σ_k that characterizes the spatial spread and

orientation of the heatmap:

$$\mathbf{\Sigma}_k = \sum_{\mathbf{u}\in\Omega} H_k(\mathbf{u})\,(\mathbf{u}-\mathbf{p}_k)(\mathbf{u}-\mathbf{p}_k)^\top,$$

Unlike traditional keypoint representations that encode only a 2D location, the covariance matrix provides valuable geometric context by describing the local orientation, anisotropy, and uncertainty of the underlying object part. Consequently, each detected keypoint represents not only a position but also the structural characteristics of the associated region, such as the orientation of a limb or other articulated component. During training, the detector is optimized only through the reconstruction objective over consecutive video frames. By observing how object parts move over time, the network naturally learns to associate keypoints with semantically consistent, independently moving regions, eliminating the need for manually labeled landmarks while producing robust structural representations for motion transfer and animation.

4.2.2 Dense motion network

Although sparse keypoints effectively capture the motion of evident object parts, they provide insufficient information to establish dense pixel-wise correspondences required for realistic image warping. To bridge this gap, the framework [155] uses a dense motion prediction network that transforms the sparse keypoint displacements into a dense optical flow field, specifying how every pixel in the source image should be displaced to match the motion observed in the driving frame. The dense motion estimation module assumes that an articulated object can be decomposed into a collection of locally rigid parts, each associated with a learned keypoint. Under this assumption, pixels in the same local region are expected to exhibit similar motion. The displacement of the k_{th} keypoint between the source and driving frames is first computed as

$$\Delta\mathbf{p}_k = \mathbf{p}_k^d - \mathbf{p}_k^s.$$

To propagate these sparse displacements across the image, a dense motion network predicts a set of spatial masks $M_k(\mathbf{u})$, where each mask defines the influence of a particular keypoint over pixel location $\mathbf{u}$. An additional mask is estimated for the background $\mathbf{M}_0(\mathbf{u})$, allowing the network to distinguish foreground motion from static regions. Each keypoint displacement is then broadcast over the image and weighted by its corresponding mask, producing a coarse optical flow that approximates locally rigid motion of an entity. However, many real-world entities, such as humans, animals, and clothing, exhibit significant non-rigid deformations that cannot be adequately represented by rigid-part translations alone. To model these non-rigid deformations, the network simultaneously predicts a residual flow field, denoted by $\Delta\mathbf{F}(\mathbf{u})$, which captures subtle local motions

arising from articulation, stretching, bending, and other complex geometric variations. The final dense optical flow field is obtained by combining the coarse part-based motion with the residual flow:

$$\mathbf{F}(\mathbf{u}) = \sum_{k=1}^{K} \mathbf{M}_k(\mathbf{u})\, \Delta\mathbf{p}_k + \Delta\mathbf{F}(\mathbf{u}).$$

This formulation enables the model to jointly capture large-scale rigid movements and fine-grained non-rigid deformations, resulting in a dense and geometrically coherent motion field. The estimated flow is subsequently passed to the motion transfer network, where it is used to warp the feature representations of the source image, enabling the synthesis of a realistic animated frame that faithfully follows the motion dynamics of the driving video.

4.2.3 Thin-plate spline motion model (TPSMM)

LiveImage adopts TPSMM [229] as its single-domain motion-model backbone. Earlier unsupervised animation methods commonly model motion around each detected keypoint using a local affine transformation. Although affine transformations capture translation, rotation, scaling, and shear, they cannot represent non-linear deformations such as bending and twisting. TPS addresses this limitation by interpolating the motion between corresponding keypoints while minimizing the bending energy

$$min \iint_{\mathbb{R}^2} \left(\left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right) \mathrm{d}x\mathrm{d}y,$$

where f denotes the TPS transformation ($\mathcal{T}^{\text{tps}}$). The resulting transformation consists of a global affine component and a non-linear radial basis component:

$$\mathcal{T}_k^{\text{tps}}(u) = \mathbf{A}_k \begin{bmatrix} u \\ 1 \end{bmatrix} + \sum_{i=1}^{N} \mathbf{w}_{k,i} U(||p_{k,i}^d - u||_2), \qquad U(r) = r^2 \log r^2,$$

where $\mathbf{A}_k \in \mathbb{R}^{2\times 3}$ is the affine component, $p_{k,i}^d$ are the driving-frame keypoints, and $\mathbf{w}_{k,i} \in \mathbb{R}^2$ are the non-linear coefficients. Given corresponding driving-frame keypoints and pixel coordinates $(p_{k,i}^d, u)$, the parameters are obtained analytically by solving the standard TPS linear system. $U(r)$ is a radial basis function that determines the influence of each keypoint on the pixel location. LiveImage estimates K independent TPS transformations, each using N control points, giving a total of $K \times N$ keypoints. These transformations provide local motion components that can specialize to different image regions, although their spatial assignments are learned implicitly rather than explicitly supervised. In parallel, a separate affine transformation models background motion T^{bg}. A dense motion

network combines the K foreground TPS transformations and the background transformation using contribution maps M_k, where $\sum_{k=0}^{K} M_k(p) = 1$:

$$\mathcal{T}(u) = M_0(u)T^{\text{bg}}(u) + \sum_{k=1}^{K} M_k(u)\mathcal{T}_k^{\text{tps}}(u).$$

Thus, different image regions can follow different motion components, while the contribution maps provide smooth transitions between them. The network also predicts multi-scale occlusion masks to identify regions where the warped source features are unreliable, particularly in areas that become newly visible due to disocclusion. These regions are subsequently handled by the inpainting network, which synthesizes the missing content. Despite its greater flexibility than a local affine model, TPSMM remains sensitive to the quality of the detected keypoints. The learned keypoints do not explicitly correspond to anatomical joints, and the resulting transformations do not impose skeletal, depth-ordering, or physical constraints. Moreover, closely positioned or collapsed keypoints can make the TPS system ill-conditioned, leading to unstable deformations and motivating the keypoint regularization introduced in Section 4.3.1. Since TPS can only transform appearance information already present in the source image, large deformations may also produce texture stretching, blurring, and boundary artifacts. This limitation motivates the motion condition network described in Section 4.3.2.

4.3 Methodology

LiveImage is an animation framework designed to streamline video-driven image animation with minimal time and effort. The user provides a source image $\mathcal{I}_s$ and a driving video $\mathcal{V}$ containing a single object; LiveImage produces an animation sequence of the input image similar to the video motion. Figure 4.1 provides an overview of LiveImage, illustrating its operations with an input image and a driving video. A single-domain motion model, such as FOMM [156] or TPS [229], with keypoint regularization is used to estimate smooth motion. Further, a keypoint detector [156] identifies motion-specific keypoints and captures the object's location and motion. A separation loss (Eqn. 4.1) prevents keypoint collapse in successive frames during motion. For example, throughout the motion, keypoints in one frame may erroneously correspond to the same location in the successive frame. Furthermore, keypoints are fed into a dense motion network that produces an optical flow field, which is then used to warp the source image to generate the animated frames. Finally, the generated sequence is passed to the motion condition network to produce a coarse-to-fine animation sequence.

A motion condition network (MCN) is proposed for estimating the coarse-to-fine animated sequence. The appearance and motion encoders estimate features from the coarse-grained and driving frames. Further, an MLP block consisting of linear layers (discussed in subsection 4.3.2) estimates the motion weight

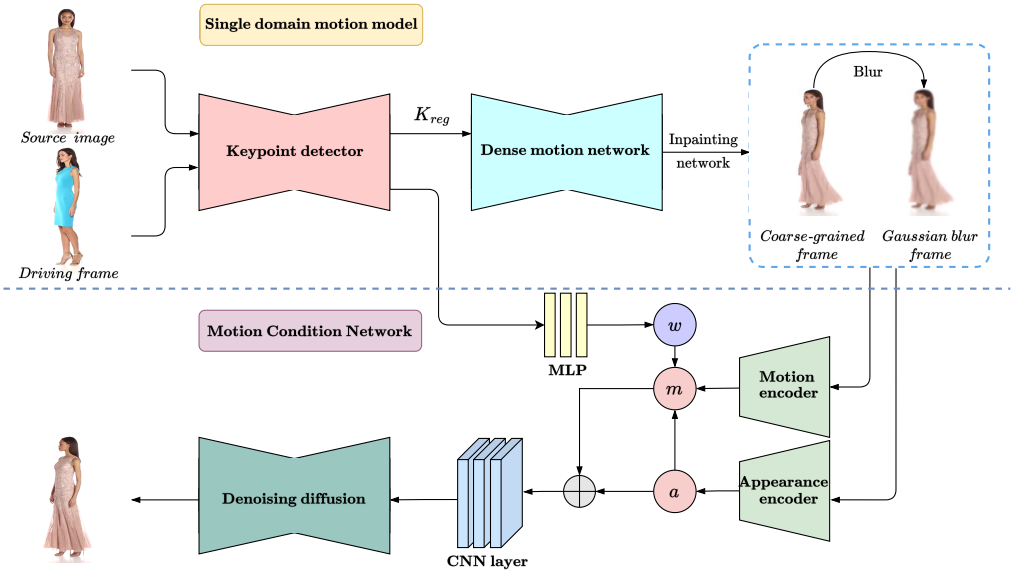


Figure 4.1: Overview of the proposed motion model for image animation. The proposed motion model (LiveImage) has two modules: SDMM and MCN. SDMM detects keypoints, estimates dense optical flow with keypoint regularization, and generates coarse images via inpainting. MCN refines motion using keypoint differences based on weight amplitude, and the diffusion model synthesizes coarse-to-fine frames.

amplitude that is determined based on the difference between the keypoints of the source and the driving frames. Since single-domain motion models can introduce shape distortions under large pose changes, Gaussian blur is applied to the coarse-grained images to handle these artifacts. The weights of the coarse-grained and Gaussian-blurred frames are then adjusted based on the magnitude of the pose changes. Higher weights are assigned to the coarse-grained frames for small pose changes, whereas larger weights are assigned to the Gaussian-blurred frames for large pose changes. Finally, a diffusion denoising model [167] is applied to generate a smooth and non-jittery animated video. The motion condition and appearance information are fed into the denoising diffusion model to generate coarse-to-fine animation frames.

4.3.1 Keypoint regularization

Single-domain motion model comprises a keypoint detector, a dense motion network, and an inpainting network. The transformation of keypoints effectively handles the distortion such that $\mathcal{T}(\mathbf{p}_s) = \mathbf{p}_d$, where $\mathbf{p}_s$ and $\mathbf{p}_d$ are keypoints of source and driving frames, respectively, and $\mathcal{T}$ represents the affine transformations.

The keypoint detector network estimates the keypoints of both the source and driving frames. These keypoints are then fed into a dense motion network that estimates the optical flow field and occlusion mask. Finally, the inpainting network utilizes the flow field and occlusion mask to generate the final source animation video. The above methods need to estimate smooth, accurate motion.

Inspired by Suwajanakorn et al. [173], keypoint regularization is performed to extract meaningful motion information from the input video by adding a separation loss as

$$\mathcal{L}_{sep} = \frac{1}{k^2} \sum_{p=0}^{k-1} \sum_{q=0, p\neq q}^{k-1} \sum_{i=0}^{n-1} \max\left(0, \delta - ||\mathbf{p}_i^p - \mathbf{p}_i^q||^2\right), \tag{4.1}$$

where k and n represent the number of keypoints and frames respectively, and p, q denote the keypoint indices. The separation loss captures meaningful semantic information and penalizes keypoints that are close to each other. A threshold δ is set to avoid the collapse of two keypoints and maintain a proper separation.

These keypoints are passed into a dense motion network that produces an optical flow field, which is then fed to the inpainting network that warps the source image $\mathcal{I}_s$ to generate the animated frames $\mathcal{I}_i^g$. The generated frames give prominent results, however these suffer from artifacts in regions of large pose changes. The Gaussian blur image $\mathcal{I}_i^b$ is computed by convolving the coarse-grained image with a Gaussian as

$$\mathcal{I}_i^b = G(\sigma) * \mathcal{I}_i^g,$$

where G is the standard 2D Gaussian function with variance σ. This set of frames is used to refine visual quality in case of large pose changes in the driving video, as described next.

4.3.2 Motion condition network

A motion condition network is introduced to predict motion-condition information for the diffusion model, thereby refining the initially generated coarse-grained results. Applying the single-domain motion model demonstrates that animated images encounter artifacts and distortion when faced with large pose changes. Conversely, small pose changes yield satisfactory results. Correspondingly, low-resolution images exhibit fewer artifacts with large pose changes but fail to generate good visual quality results.

An MLP is used to introduce a motion weight amplitude determined by the keypoint difference between the image and driving frame given by: $w = \text{MLP}((\mathbf{p}_s - \mathbf{p}_d), t)$ for cross-identity and $w = \text{MLP}((\mathbf{p}_s - \mathbf{p}_{s\to d}), t)$ for self-identity, where $\mathbf{p}_s$, $\mathbf{p}_d$ are the source and driving keypoints, $\mathbf{p}_{s\to d}$ represents source-to-driving keypoint, and t denotes the diffusion timestep. The keypoints

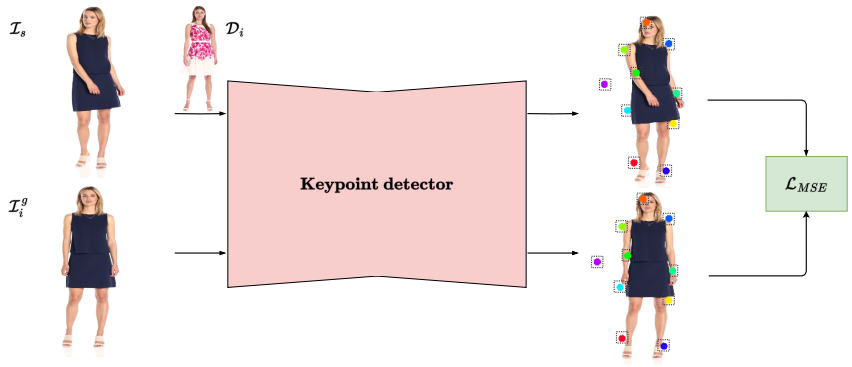


Figure 4.2: During inference, fixed-size patches from the source image and Gaussian-blurred frames are extracted from motion keypoints. MSE loss aligns appearance features for cross-identity motion transfer.

are estimated using the keypoint detector, similar to the single-domain motion model [156]. For same-identity reenactment, the keypoint difference is computed between the source frame and the source-to-driving frame. For cross-identity reenactment, it is computed between the source and driving frames. Using the source and source-to-driving keypoint can lead to failure when there is a disparity in the shape or topological structure of objects between the source and driving frames.

Further, an appearance encoder $a = \mathcal{E}(\mathcal{V})$ is used to estimate the appearance feature information of the driving frames during training time, where $\mathcal{E}$ represents the appearance encoder. During the inference, appearance is used as $\mathcal{E}(\mathcal{I}_i^g)$ and aligns features of the coarse-grained frame with the source image, as shown in Figure 4.2. Fixed-size image patches are extracted around the motion keypoints, and an MSE loss (Eqn. 4.2) is used to align the relevant appearance information.

$$\mathcal{L}_{MSE} = \frac{1}{n}\sum_{i=1}^{n} \|\mathcal{P}(\mathcal{I}_s) - \mathcal{P}(\mathcal{I}_i^g)\|^2, \tag{4.2}$$

where n is the number of frames, and $\mathcal{P}$ denotes patch extraction. The motion weight is used to assign a higher weight to the original image $\mathcal{I}_i^g$ when small pose changes occur. If there are drastic changes in pose, a larger weight is assigned to the Gaussian-blurred image $\mathcal{I}_i^b$. It is achieved by analyzing the keypoint differences between frames, providing insights into whether significant pose changes have occurred. The motion weight amplitude is estimated similar to the approach of Zeng et al. [225] as $w_i = e^{(w-\lambda)}$ to $\mathcal{I}_i^b$ when large pose occurs and $w_i = e^{(-w+\lambda)}$ to $\mathcal{I}_i^g$ when small pose change occurs and vice-versa. Here, w denotes the weight, and the hyperparameter $\lambda = 0.4$.

Further, a motion encoder $\mathcal{M}$ is used to leverage appearance information from the appearance encoder $\mathcal{E}$ and the coarse-grained image. Motion condition

information is further estimated by determining the weight value for the coarse-grained frames and blurred image $\mathcal{I}_i^b$. The motion condition is assigned as per the following criteria:

$$m = \begin{cases} w_i \cdot \mathcal{M}(\mathcal{I}_i^b, a), & \text{for large pose gap,} \\ w_i \cdot \mathcal{M}(\mathcal{I}_i^g, a), & \text{for small pose changes.} \end{cases}$$

ALGORITHM 1: Motion condition network

Input: Coarse-grained frames $\{\mathcal{I}_i^g\}_{i=1}^n$, Gaussian-blurred frames $\left\{\mathcal{I}_i^b\right\}_{i=1}^n$

Output: Conditional inputs (a, m) for DDIM

for $i = 1$ **to** n **do**
  Appearance encoder: $a = \mathcal{E}(\mathcal{V})$
  if $w = \text{MLP}(\mathbf{p}_s - \mathbf{p}_{s\to d}, t)$ **or** $w = \text{MLP}(\mathbf{p}_s - \mathbf{p}_d, t)$ **then**
    `// Case 1`
    Large $\mathbf{p}$ translation:
    Assign $\mathcal{I}_i^b \leftarrow (w_i \uparrow)$
    `// Case 2`
    Small $\mathbf{p}$ translation:
    Assign $\mathcal{I}_i^g \leftarrow (w_i \uparrow)$
    where $(\uparrow)$ denotes the higher value
    Motion condition:

$$m = \begin{cases} w_i \cdot \mathcal{M}(\mathcal{I}_i^g, a), \\ w_i \cdot \mathcal{M}(\mathcal{I}_i^b, a). \end{cases}$$

    Fuse appearance and motion: $a \oplus m$
  end
end
Conditional inputs (a, m) for DDIM

Here, m and a are determined as the motion condition and appearance, $\bigoplus$ denotes the channel-wise concatenation in Algorithm 1, and this information is fused in the diffusion process. Furthermore, LiveImage uses motion-condition information to add noise to coarse-grained frames. Subsequently, the frames are iteratively denoised via the diffusion process with motion conditioning. A few-step sampling process [167] is utilized to generate outputs conditioned on a, m. The optimization function for the denoising diffusion model is

$$\mathcal{L}_d = E_{x_0,(a,m),z\sim N(0,1),t}\left[\|\epsilon - \epsilon_d(x_t, t, (a, m))\|_2^2\right], \tag{4.3}$$

where $\mathcal{L}_d$ represents the diffusion loss, ϵ_d is the denoising diffusion model.

4.4 Experiments

This section describes the benchmark datasets and evaluation metrics used to assess the proposed approach. Various components are compared with the previous state-of-the-art methods to perform an ablation study of the proposed method.

4.4.1 Benchmarks

Training is conducted on four primary datasets: Voxceleb [131], TaiChiHD [156], MGif [156], and Fashion [224]. The data is preprocessed and split into training and test sets as in FOMM [229]. To assess the results, the following metrics are used from the image animation literature [155, 156, 159]

- **L_1**: The L_1 loss measures the video reconstruction loss by comparing pixel values between the generated and ground truth frames.
- **AED:** The Average Euclidean Distance (AED) calculates the feature representation within the reconstructed video along with the ground truth. Additionally, it estimates the preservation of identity in the reconstructed video. Primarily, it uses a person re-identification network [70] trained on the human pose dataset.
- **AKD:** The average Keypoint Distance (AKD) calculates the distance between the keypoints in the generated video and those in the ground truth. To achieve this, a pre-trained keypoint detector network is used, similar to FOMM [156], trained on human poses. It helps to identify whether the generated video's motion matches the ground truth video motion.

4.4.2 Training

Training is performed on an NVIDIA A100 GPU with 80 GB of memory. LiveImage uses 62GB of memory over the training time of 72 hours. TPS [229] is used as a single-domain motion model for coarse-grained image generation. Keypoint regularization uses separation loss with the hyperparameter value for separation loss set to δ=0.1, while the value of the hyperparameter λ is set to 0.4. The diffusion denoising model incorporates conditional diffusion loss, as summarized in subsection 4.3.2. The appearance branch uses a 7×7 convolution followed by two residual blocks, producing 16-channel features at 256×256. Motion attributes are combined with the timestep embedding through an MLP. The coarse warped frames, converted to Gaussian-blur frames, are resized to 256×256, projected to 16 channels, and fused using motion-dependent weighting. The fused representation is combined and then processed by residual blocks to produce a 3-channel conditioning image c. The model undergoes training for 100 epochs using the



Figure 4.3: LiveImage results on VoxCeleb [131], TaiChiHD [156] Fashion[224] and MGif[156] dataset.

Adam optimizer [98] with a learning rate of $\eta = 10^{-4}$. Table 4.1 summarizes the complete training configuration.

Table 4.1: Training configuration.

Group	Parameter	Value
Optimisation		
	Optimiser	Adam
	Learning rate	10^{-4}
	Batch size	8
	Epochs	100
Loss weights		
	λ_{sep} , λ_{mse}	1.0, 1.0
	Separation threshold δ	0.1
	Weight offset λ	0.4
Diffusion		
	Training timesteps	100
	Variance schedule	10^{-4} to 0.02
Data		
	Resolution	256×256
	Patch size	16×16
Hardware		
	Device	NVIDIA A100, 80 GB
	Peak memory	62 GB
	Training time	72 h approx

4.4.3 Results and comparison

LiveImage is compared with state-of-the-art methods, such as FOMM [156], MRAA [159], TPS [229], MT [177], DAM [178] and LIA [191, 192]. The proposed method is evaluated quantitatively and qualitatively on VoxCeleb, TaichiHD, Fashion, and MGif datasets. The quantitative comparison is presented in Table 4.2 on the given benchmarks.

LiveImage surpasses previous state-of-the-art methods and produces high-quality results. Reconstruction loss, AKD, and AED are computed for quantitative evaluation, demonstrating the superior performance of the proposed method. Results are detailed in Figure 4.3, and qualitative comparison is illustrated in Figure 4.4. LiveImage demonstrates superior appearance and motion quality compared to other baseline methods.

Qualitative comparison LiveImage is qualitatively evaluated on four benchmarks: VoxCeleb [131], TaiChiHD [156], Fashion [224], and MGif [156]. LiveImage demonstrates superior performance compared to state-of-the-art methods, as

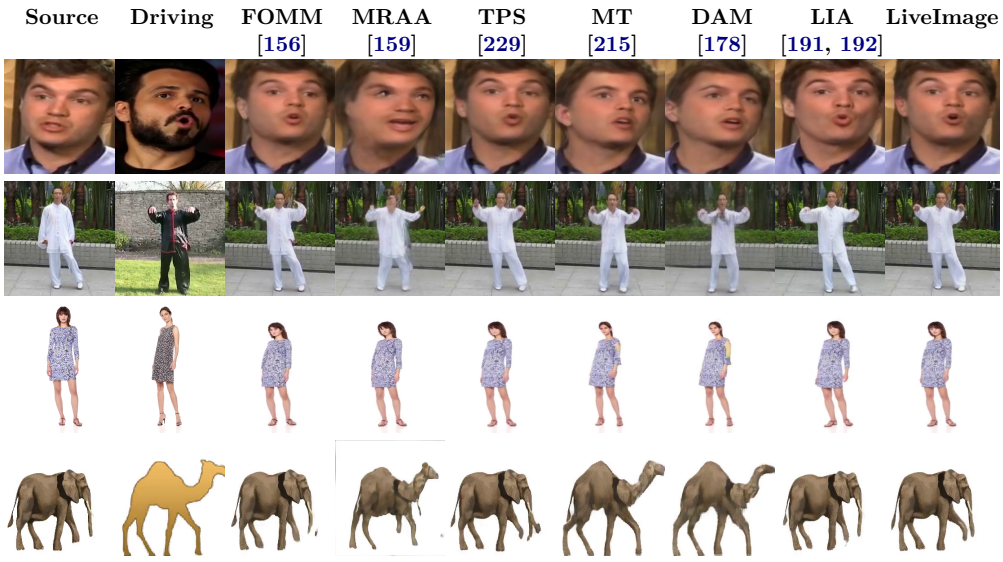


Figure 4.4: Comparison of LiveImage with state-of-the-art methods on VoxCeleb, TaiChiHD, Fashion and MGif datasets.

shown in Figure 4.4. FOMM [156] and MRAA [159] struggle to capture motion information, resulting in poor visual quality. Although TPS [229], MT [177], DAM [178] and LIA [191, 192] generate promising results, there are artifacts in the generated sequence if a large motion occurs. LiveImage captures motion information and generates better visual quality animated sequences as per the diving video motion while preserving the original shape, as depicted in Figure 4.3.

Quantitative comparison LiveImage is compared against FOMM [156], MRAA [159], TPS [229], MT [177], DAM [178], and LIA [191, 192]. In the Table 4.2, LiveImage outperforms the given benchmarks. L_1, AED, and AKD are used as evaluation metrics for the VoxCeleb [131], TaiChiHD [156], and Fashion [224] datasets; on the other hand, L_1 is used as an evaluation metric for the MGif dataset. LiveImage outperformed the VoxCeleb, TaiChiHD, and Fashion datasets quantitatively; L_1, AED, and AKD values are lower than those of state-of-the-art methods (lower is better). On the other hand, for the MGif dataset, LiveImage surpasses the FOMM [156], MRAA [159], TPS [229], DAM [178], and LIA [191, 192], except MT [177]. However, on the MGif dataset, LiveImage achieves an $\mathcal{L}_1$ loss comparable to that of MT [177]. These results demonstrate that the proposed method outperforms state-of-the-art methods in quantitative evaluation.

Table 4.2: Quantitative comparison of the proposed approach with state-of-the-art methods.

	VoxCeleb			**TaiChiHD**			**Fashion**			**MGif**
	$L_1\downarrow$	AED↓	AKD↓	$L_1\downarrow$	AED↓	AKD↓	$L_1\downarrow$	AED↓	AKD↓	$L_1\downarrow$
FOMM [156]	0.043	0.140	1.294	0.063	0.179	6.862	0.0155	0.0683	1.490	0.0225
MRAA [159]	0.040	0.133	1.280	0.047	0.152	5.580	0.0148	0.0645	1.181	0.0265
TPSMM [229]	0.039	0.125	1.220	0.045	0.151	**4.570**	0.0126	0.0574	0.928	0.0212
DAM [178]	0.039	0.124	1.240	0.044	0.146	4.790	0.0112	0.0540	1.041	0.0201
MT [177]	0.038	0.116	**1.180**	0.045	0.148	4.670	0.0118	0.0532	1.020	**0.0200**
LIA [191, 192]	0.041	0.132	1.256	0.054	0.162	4.843	0.0142	0.0626	1.275	0.0258
LiveImage	**0.036**	**0.112**	1.190	**0.044**	**0.146**	4.590	**0.0112**	**0.0528**	**0.904**	0.0201

4.4.4 Ablation Study

With and w/o keypoint regularization

Keypoint regularization is used to estimate smooth, accurate motion, resulting in non-jittery animated frame generation. A separation loss is used to achieve this. The separation loss prevents the keypoint collapse during motion. The single-domain motion model produces jittery frames and topological artifacts without keypoint regularization. As depicted in Figure 4.5, the absence of keypoint regularization led to jittered hair collapse with the body parts. The keypoint regularization results, shown in Figure 4.5, generate a smooth and non-jittery frame, and the hair and mouth are generated precisely. With the keypoint regularization, the generated frame prevents information loss and quantitatively (Table 4.3) and qualitatively improves the results compared to frames generated without keypoint regularization.

With and w/o motion condition

Without the motion condition, the generated frames struggle to handle the large pose change. Additionally, without this condition, the jittering of parts and the appearance of artifacts undergo substantial changes. As shown in Figure 4.5, the results suffer from artifacts if a large pose change occurs. The diffusion model effectively addresses large pose changes by incorporating the motion condition. To achieve this, a motion weight is assigned to prioritize the motion encoder. If drastic pose changes occur, it assigns a higher weight to the blurred frame. The resulting frame preserves the source's appearance information and the driving frame's motion. The motion-conditioned model achieves lower loss than the model without motion conditioning in both quantitative (Table 4.3) and qualitative evaluation. The motion-conditioned generated frame visually contains richer details than unconditioned generated frames.

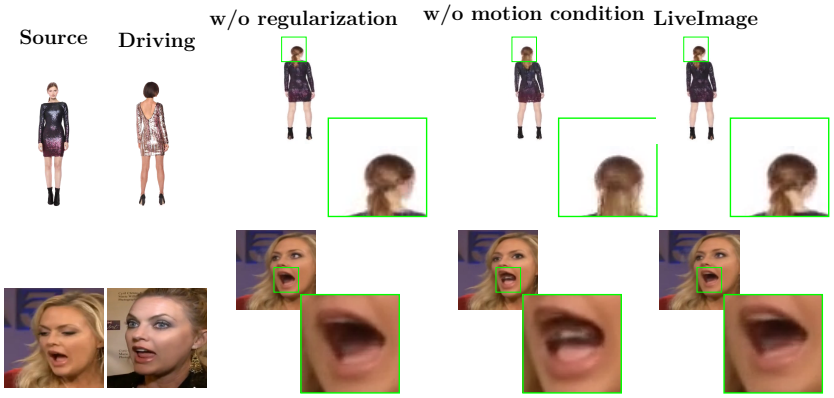


Figure 4.5: The qualitative results on different settings and generated results w/o motion condition, w/o keypoint regularization, and with the LiveImage method.

Table 4.3: Ablation results—the quantitative evaluation on w/o motion condition, w/o keypoint regularization, and the proposed method.

	Vox			**Fashion**		
	L_1	AED	AKD	L_1	AED	AKD
W/o keypoint reg.	0.039	0.125	1.210	0.0118	0.0547	0.910
W/o motion condition	0.041	0.134	1.260	0.0122	0.0568	0.916
LiveImage	**0.036**	**0.112**	**1.190**	**0.0112**	**0.0528**	**0.904**

4.5 Limitations

LiveImage performs well compared to state-of-the-art methods, but it does have some limitations. One notable issue is the occurrence of false correspondences between the video object and the source image, even when both belong to the same category. Additionally, if the input image lacks specific details, LiveImage does not automatically add those details when applying motion. LiveImage also encounters difficulties with complex background motion and self-occlusion, as illustrated in Figure 4.6. Moreover, LiveImage is constrained to a single-domain motion model, meaning it may not accurately estimate motion from a cartoon or sketch video, and vice versa (see Figure 4.6).

4.6 Conclusion

This Chapter presents LiveImage, a novel image animation framework with potential applications in video editing, animation, and entertainment content generation. It takes an image and a driving video as input and generates the animation video of the image based on the driving video's motion. LiveImage addresses the challenge posed by a single-domain motion model due to large pose changes in

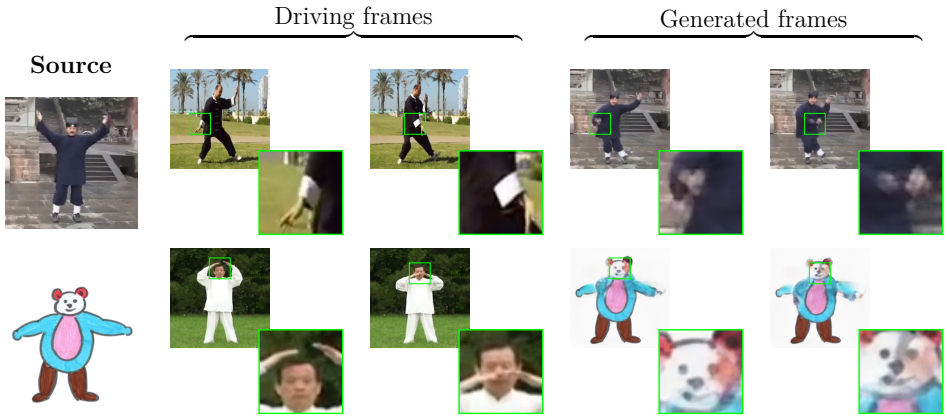


Figure 4.6: LiveImage struggles with self-occlusion, false correspondence, and cross-domain (sketch) motion transfer during animation.

videos. It introduces keypoint regularization to prevent keypoint collapse, estimate smooth, accurate motion, and achieve significant results. The proposed motion condition network helps to generate coarse-to-fine animation sequences and produces results better than the state-of-the-art methods.

Chapter 5

SketchAnim: Real-time sketch animation transfer from videos

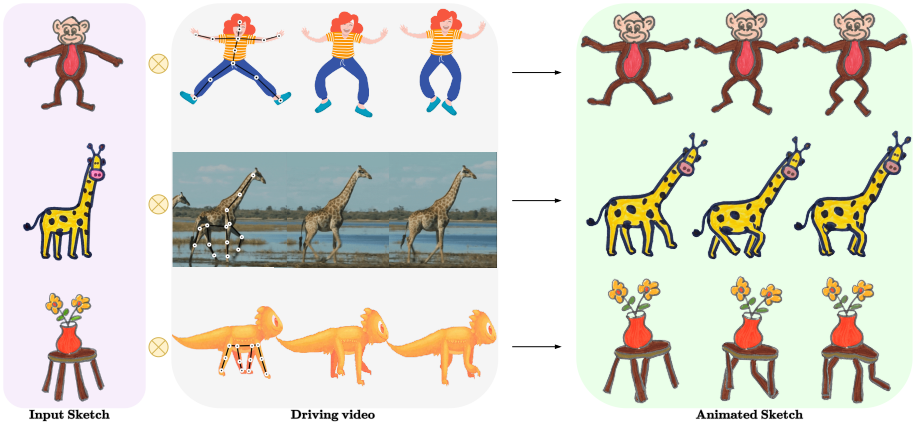


Figure 5.1: Sketch animation using motion from exemplar video.

5.1 Introduction

Sketches are a great medium for visual communication, and animating them can add dynamism and interactivity. Traditional tools for animating sketches require a unique set of skills to bring hand-drawn characters to life. Artists often refer to some existing videos and try to replicate a similar video motion to the sketch character when attempting to animate it. However, this process is often time-consuming, and an inaccurate replication of motion may make the character's movement look non-fluent or unnatural, especially for those with little to no experience in animation.

Character animation has been an area of extensive research in recent years. Initial work includes that of Xing et al. [212] who proposed a method that auto-completes a set of continuously repeated strokes during sketching and performs

a keyframe-based animation. TraceMove [138] provides a data-assistant user interface that enables novice animators in sketching and 2D character animation. It takes skeleton points as user input and utilizes a motion capture dataset to predict the skeleton in the subsequent frames. Live-Sketch [170] extracts dynamic deformation from videos as moving control points and transfers the motion to sketches while addressing certain challenges, such as extracting motion from videos, video-to-sketch alignment, and motion transfer.

Recent advancements in sketch animation utilize neural network-based techniques like CharacterGAN [72], which relies on keypoints added manually by the user. Another approach, AnimationDrawing [164], uses video motion and applies it to the sketch. However, it has limitations, supporting only the animation of biped characters and generating artifacts in instances of inaccurate joint estimation and occlusion.

This chapter presents SketchAnim, a sketch animation method that transfers motion from a driving video to a hand-drawn sketch. The proposed method takes as input a sketch to be animated, a driving video from which the motion is to be applied, and a skeleton of the character from the sketch. It then generates an animated sketch with the motion of the video applied to it. SketchAnim offers multiple advantages at various stages. It supports animation of sketch characters with different structures, and is not just restricted to biped or quadruped characters. For instance, this method seamlessly handles inanimate object sketches.

The proposed approach generates high-quality animations compared to other recent techniques. The main contributions of this chapter are as follows:

- An automatic, user-friendly, and efficient sketch animation framework capable of handling different types of motions such as biped, quadruped, and inanimate. The proposed framework offers a two-stage approach to animating sketches from videos: motion extraction, and motion transfer.
- A novel motion transfer approach is introduced to map video skeleton motion to sketch skeleton. A depth-order-based approach to handle self-occlusion during sketch deformation.
- The proposed method shows superior performance, both quantitatively and qualitatively, compared to state-of-the-art methods.

5.2 Methodology

SketchAnim is a robust and easy-to-use framework for animating sketches using motion from a driving video. The algorithm begins by obtaining user-provided input, including a sketch $\mathcal{S}$, a driving video $\mathcal{V}$ containing a moving object, and a user-provided skeleton $\mathcal{K}_v^1$ of the object in the first frame of the driving video.

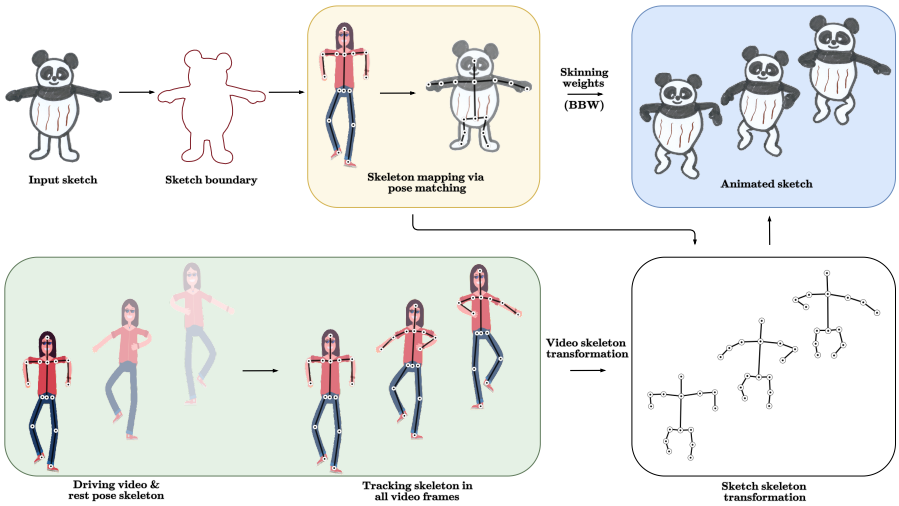


Figure 5.2: Overview of the SketchAnim framework for sketch-based animation. The method involves taking a sketch and a video as input. The user then draws a skeleton on the initial frame of the video. Mapping of this skeleton to the input sketch is done by shape matching with MVC. Thereafter, the transformation of skeletal bones is estimated by tracking the skeletal joints across the video frames. Finally, the sketch is animated using linear blend skinning with BBW weights.

The driving video may be an animated video or a real-world video that provides realistic character motion for the sketch. The video skeleton $\mathcal{K}_v^1$ is tracked across all video frames. Subsequently, bone-joint angles are estimated for the motion skeleton. The video skeleton is mapped to the sketch to compute a sketch skeleton $\mathcal{K}_s$, and skinning weights are computed using bounded biharmonic weights [83]. The proposed algorithm then computes the bone transformations required for sketch animation through Linear Blend Skinning (LBS). Finally, the transformations are applied to the sketch to generate an animated output video $\mathcal{V}_s$, effectively portraying the given sketch in accordance with the dynamics of the driving video sequence. The output video is presented to the user as the animated result, encapsulating the essence of the original sketch within the context of the driving video. Algorithm 2 outlines the primary steps of the proposed approach, and Figure 5.2 shows the overall sketch animation framework.

5.2.1 Skeleton mapping

The Segment Anything Model (SAM) [100] is used to segment the characters in the sketch and the first frame of the driving video. The boundaries of these segmentation regions are resampled with points spaced d distance apart as polygons $\mathcal{B}_s$ and $\mathcal{B}_v$ respectively for the sketch image and the video frame. At this stage, the character poses represented by $\mathcal{B}_s$ and $\mathcal{B}_v$ may not exactly match.

ALGORITHM 2: SketchAnim: Video-to-Sketch motion transfer

Input: Sketch $\mathcal{S}$, driving video $\mathcal{V}$, and video skeleton $\mathcal{K}_v^1$
Output: Sketch video $\mathcal{V}_s$
Segment sketch boundary polygon $\mathcal{B}_s$
Compute triangulation $\mathcal{T}_s$ (CDT) of $\mathcal{B}_s$
Segment the first frame of the video $\mathcal{B}_v$
Compute sketch skeleton $\mathcal{K}_s$ from $\mathcal{K}_v^1$ using MVC (Sec. 5.2.1)
Tracking video skeleton $\mathcal{K}_v^1$ across all frames
Motion transfer between video and sketch skeleton(Sec. 5.2.2)
Compute BBW weights for $\mathcal{T}_s$ and $\mathcal{K}_s$ (Sec. 5.2.6)
Deform $\mathcal{S}$ using LBS to produce $\mathcal{V}_s$

The skeleton $\mathcal{K}_v^1$ of the first frame of the video is provided by the user as input. The skeleton vertices are then automatically mapped to construct a sketch skeleton $\mathcal{K}_s$ with the topology of the former. The proposed skeleton mapping approach is based on shape matching between boundaries $\mathcal{B}_v$ and $\mathcal{B}_s$. To start with, $\mathcal{B}_s$ and $\mathcal{B}_v$ do not have much correlation as they were constructed independently of each other. The neural deformation pyramid approach of Li and Harada [112] is used to deform the video boundary $\mathcal{B}_v$ to obtain a close approximation of $\mathcal{B}_s$. This results in a new sketch boundary $\tilde{\mathcal{B}}_s$ whose points are in correspondence with $\mathcal{B}_v$ and $|\tilde{\mathcal{B}}_s| = |\mathcal{B}_v|$. Once the shape correspondence between the two boundaries is established, the vertices of the video skeleton $\mathcal{K}_v^1$ are transformed to those of $\mathcal{K}_s$ via barycentric coordinates. To perform the same, mean value coordinates (MVC) [46] are used to calculate the barycentric coordinate $\alpha_v \in \mathbb{R}^{|\mathcal{B}_v|}$ of a vertex $b_v \in \mathcal{K}_v^1$ with respect to the boundary polygon $\mathcal{B}_v$. The corresponding skeleton point p_s (see Figure 5.3) in the sketch skeleton $\mathcal{K}_s$ w.r.t. $\tilde{\mathcal{B}}_s$ can be calculated as

$$p_s = \sum_{j=1}^{|\tilde{\mathcal{B}}_s|} b_{s,j}\, \alpha_{v,j}, \forall b_{s,j} \in \tilde{\mathcal{B}}_s. \tag{5.1}$$

Transferring video motion to the sketch requires tracking the motion of $\mathcal{K}_v^1$ across the video frames. This can be achieved with point tracking in the video; the primary challenge here comes from occlusions and error propagation across frames. It requires a sparse tracking method that is reasonably resilient to occlusions and propagated errors. CoTracker [90] tracks vertices of $\mathcal{K}_v^1$ across video frames, while keeping the skeleton topology fixed. This results in a sequence of video skeletons $\{\mathcal{K}_v^i\}$ that are used for motion mapping to the sketch.

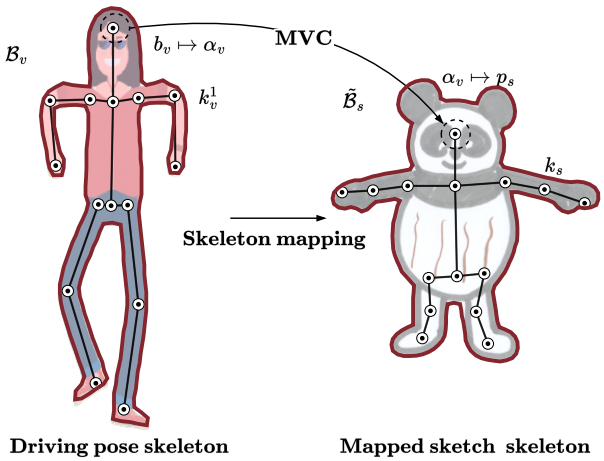


Figure 5.3: Video skeleton mapping to sketch via mean-value coordinates.

5.2.2 Motion extraction and motion transfer

The sequence of sketch skeletons $\{\mathcal{K}_s^i \mid i = 1, \ldots, n\}$ is derived from the previously computed sequence of video skeletons $\{\mathcal{K}_v^i\}$. While there are many ways to do so, for the best results, one should ensure that the skeleton motion of the video is captured by the sketch accurately and that the sketch proportions do not get deformed unnaturally.

5.2.3 Skeleton tree node hierarchy

Consider the video skeleton $\mathcal{K}_v^1$ as a tree, with the deepest vertex designated as the root. The deepest vertex is determined by considering distances from the leaf nodes in the skeleton, with edge weights set to the edge lengths. The root node is then selected as the non-leaf vertex that minimizes the sum of shortest-path distances to all reachable leaf nodes (as shown in Figure 5.4). In the case of a tie, the vertex with the highest degree is selected as the root node. The tree topology is preserved across all video frames. Since the video and sketch skeletons are isomorphic, the corresponding vertex in the sketch skeleton is selected as the root node.

5.2.4 Initial pose matching

The skeleton of the sketch $\mathcal{K}_s$ thus obtained may not have the same pose as that of the first frame of the video skeleton $\mathcal{K}_v^1$. Therefore, the pose of $\mathcal{K}_s$ is first aligned with that of $\mathcal{K}_v^1$ by matching their corresponding edges while keeping the root node fixed. The pose matching is performed in a breadth-first search (BFS) manner starting from the root node. Each edge connected to the current

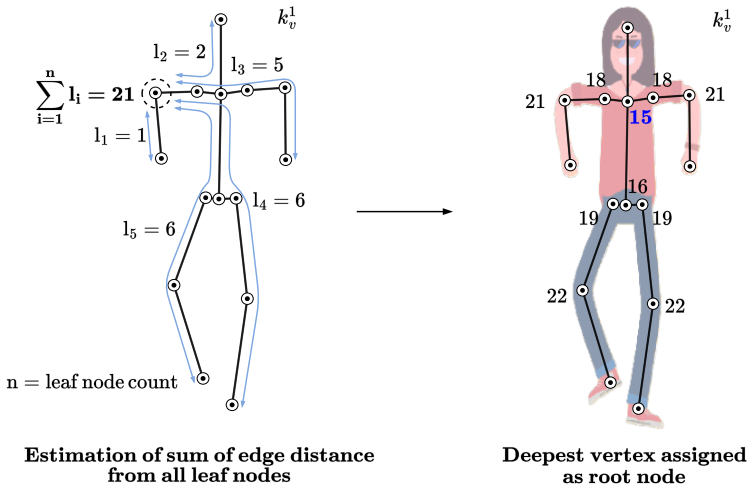


Figure 5.4: Estimation of the root node as the deepest vertex. The blue color number denotes the smallest sum from the leaf nodes and the corresponding node as the root.

node is reoriented to match the direction of its corresponding edge in $\mathcal{K}_v^1$. More specifically, for a skeleton node p_v under consideration in $\mathcal{K}_v^1$, let a connected edge be (p_v, q_v). The corresponding node q_s connected to p_s in $\mathcal{K}_s$ is modified as

$$q_s' = p_s + \frac{\|q_s - p_s\|}{\|q_v - p_v\|}(q_v - p_v). \tag{5.2}$$

After completing the BFS traversal with these modifications, the pose-matched sketch skeleton $\mathcal{K}_s^1$ is obtained, as shown in (see Figure 5.5).

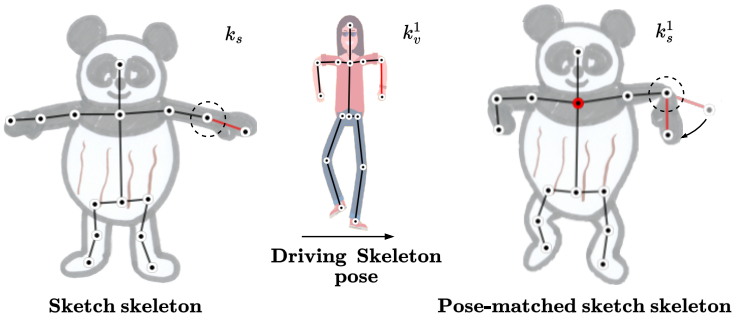


Figure 5.5: Pose matching of the sketch skeleton to the video skeleton. The skeleton is traversed in a BFS manner starting from the root node (shown in red). At a current node, transformations are applied to edges in its 1-ring to match the orientation of the corresponding source edges.

5.2.5 Motion mapping

The motion of the sequence of video skeletons $\{\mathcal{K}_v^i\}$ is mapped to the sketch skeleton $\mathcal{K}_s^1$ sequentially. The proposed approach is to compute transformations between skeletons of two successive video frames and apply that to the starting skeleton in the sketch skeleton sequence, starting from the first frame. A breadth-first traversal is again performed to compute the transformation parameters for each edge and apply them to the corresponding edges of the sketch skeleton.

During motion mapping at timestep t, the root node r_v^t undergoes only translation, which applies to the corresponding sketch skeleton as $r_s^{t+1} = r_s^t + (r_v^{t+1} - r_v^t)$. For a skeleton node p_v^t under consideration, let a connected edge be (p_v^t, q_v^t). The rotation and scaling transformations for the corresponding sketch edge are computed as follows:

$$\theta = \cos^{-1}\left(\frac{(q_v^{t+1} - p_v^{t+1}) \cdot (q_v^t - p_v^t)}{\|(q_v^{t+1} - p_v^{t+1})\|\|(q_v^t - p_v^t)\|}\right),$$
$$s = \frac{(q_v^{t+1} - p_v^{t+1})}{(q_v^t - p_v^t)}.$$

The entire BFS pass yields the sketch skeleton $\mathcal{K}_s^{t+1}$ at the next timestep. The computed transformation parameters are also retained as bone transformations for linear blend skinning in the subsequent stage.

5.2.6 Resolving self-occlusions

Self-occlusions in 2D animation are difficult to handle if the depth ordering of different parts of the object is not known. To resolve this issue, depth ordering is first established for the skeleton vertices and then propagated to the mesh vertices during skinning. Each skeleton vertex is assigned a discrete depth value, starting with zero at the root. Every other vertex is given a value equal to the number of edges it is away from the root. A negative sign is assigned to vertices whose corresponding video skeleton vertices are identified as occluded in one or more frames during tracking. This procedure establishes the depth ordering of the sketch skeleton vertices. During skinning, the depth values are propagated from the skeleton vertices to the mesh vertices using the skinning weights, following a procedure similar to that of linear blend skinning, and these depth values are used as the z-coordinate of the mesh vertices in rendering.

After transferring the video motion to the sketch skeleton, the sketch is animated by first triangulating the sketch boundary polygon $\mathcal{B}_s$ using constrained Delaunay triangulation (CDT). Additional points are added to the triangulation by Poisson disk sampling the interior of the boundary. Smooth skinning weights for this triangulation and the skeleton $\mathcal{K}_s^1$ are computed using the Bounded biharmonic

weights (BBW) [83] method. The triangulation is then deformed using linear blend skinning (LBS) and texture-mapped with the sketch image to generate each frame of the animated sketch. Algorithm 3 outlines the entire approach to motion extraction and motion mapping from video to sketch.

ALGORITHM 3: Motion transfer between video and sketch skeletons

Input: Sketch skeleton $\mathcal{K}_s$ and sequence of video skeletons $\{\mathcal{K}_v^i \mid i = 1, \ldots, n\}$
Output: Sequence of sketch skeletons $\{\mathcal{K}_s^i \mid i = 1, \ldots, n\}$
Define the deepest vertex (skeleton joint) as root
foreach skeleton $\mathcal{K}_v^i$ in the sequence **do**
  foreach joint in $\mathcal{K}_v^1$ **do**
    Initial pose matching $\mathcal{K}_v^1$ to $\mathcal{K}_s^1$
    Traverse the skeleton joint in BFS manner
    Estimate the sketch skeleton transformation using $\mathcal{K}_v^i$
    Assign depth order to each joint (root = 0)
  end
end
Compute BBW weights for $\mathcal{T}_s$ and $\mathcal{K}_s$ and deform using LBS

5.3 Results

The performance of SketchAnim is evaluated on a system equipped with an Intel Xeon CPU and an Nvidia Quadro P6000 GPU featuring 24 GB of memory. For the skeleton tracking, co-tracker [90] is used, and shape matching is performed using the deformation pyramid method of Li and Harada [112]. The proposed approach incorporates novel techniques for motion extraction, motion transfer, and occlusion handling that enhance the quality of the resulting animation. The framework is implemented in Python and source code for this implementation is available for research purposes.

In terms of computational time, SketchAnim takes approximately 30 seconds to generate an animation sequence of 24 frames with a skeleton consisting of 12-15 vertices. This duration may vary depending on the number of skeleton vertices and the frames in the driving video. In this method, the motion transfer phase is the most time-consuming, primarily at the mesh deformation and animated video rendering stages. On the other hand, in the motion extraction phase, the time taken by skeleton mapping depends upon the mesh configuration.

The sketch animation capabilities of SketchAnim are demonstrated with approximately 50 hand-drawn, colored sketches across three categories: bipeds, quadrupeds, and inanimate objects. Corresponding driving videos are collected from online sources under Creative Commons licenses. The selected cartoon and

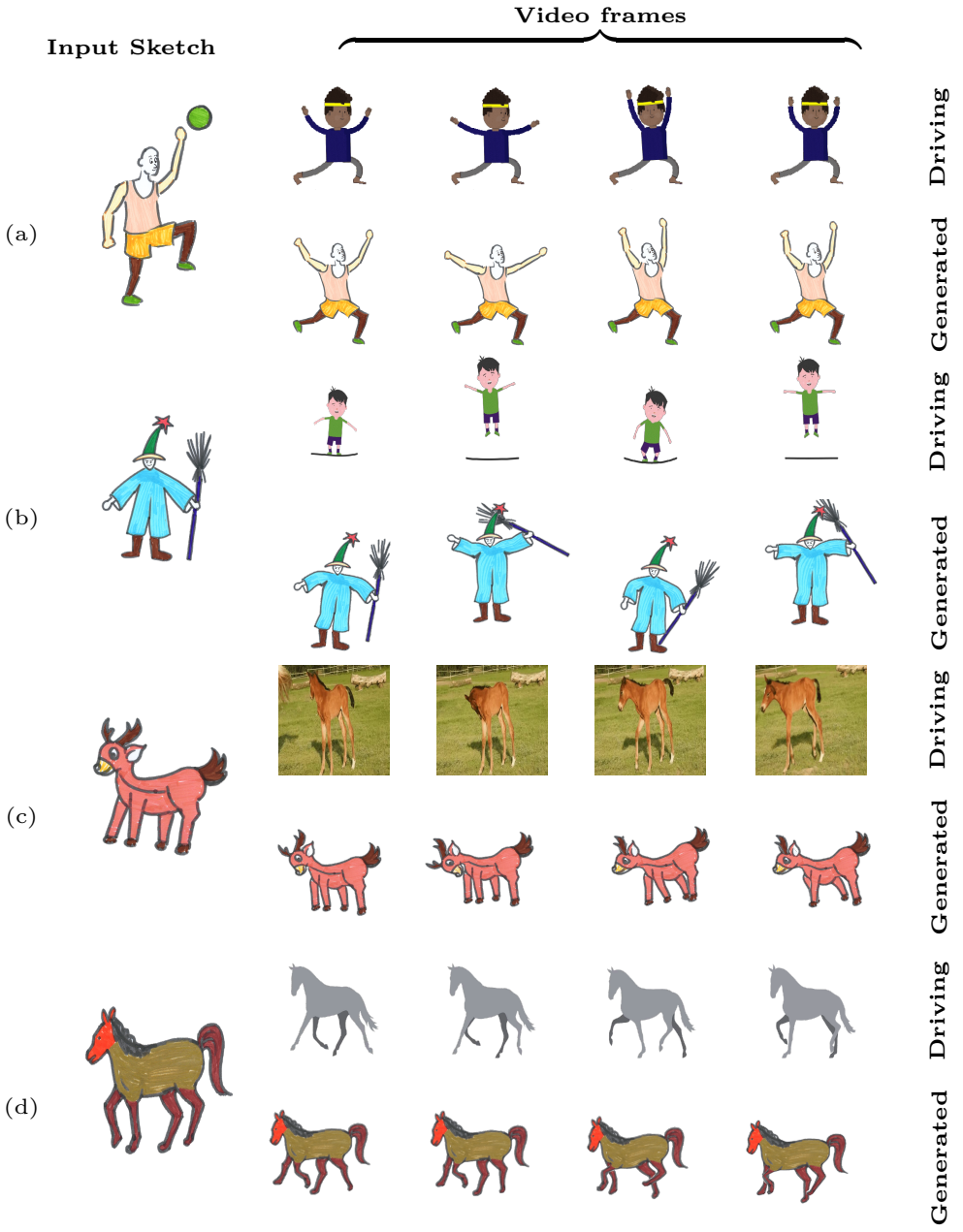


Figure 5.6: SketchAnim results on biped and quadruped sketches.

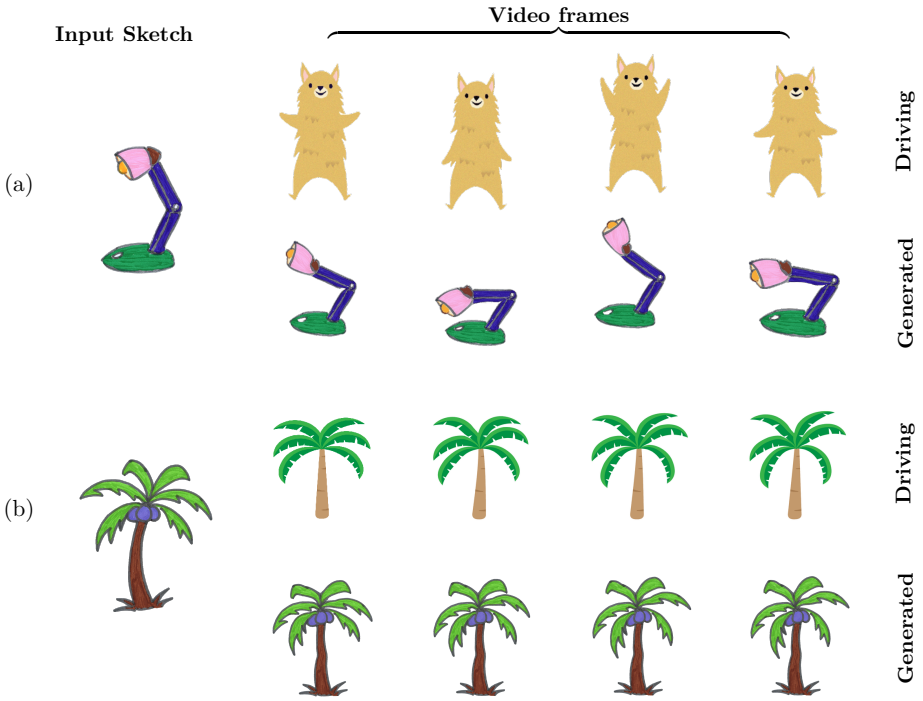


Figure 5.7: SketchAnim results on sketches of inanimate objects.

natural videos contain a central character/object visible in all the frames. Some quadruped samples are also selected from the MGif [156] dataset. The sketch dataset used in this work is available publicly for research purposes and can be accessed at https://github.com/graphics-research-group/SketchAnim/.

5.3.1 Evaluation matrices

The following metrics are used to quantitatively evaluate and compare SketchAnim with state-of-the-art methods.

- **FID:** Frechet Inception distance (FID) [71] is used to estimate the overall quality of the generated frame. It measures the similarity between the distribution of real images and generated images.
- **AED:** Average Euclidean distance (AED) estimates the identity similarity of the ground truth and generated frame. This implies that the generated frame shares the same identity as the ground truth image. It uses a person re-identification [70] network similar to TPS [229] to measure the identity features of the human body.

The input sketch is used as the real sample and the animated sketch sequence as the generated sample to evaluate FID. For estimating AED, the driving video

is considered the real sample, and an animated sketch sequence is the generated sample.

5.4 Comparisons with the state-of-the-art

5.4.1 Qualitative evaluation

The performance is assessed on animation across 50 sketch samples across different categories. Motion transfer and appearance consistency within the animated sequences are evaluated qualitatively. Figures 5.6 and 5.7 show animated sequences for three different categories of sketches: biped, quadruped, and inanimate. In the biped example (see Figure 5.6(a)), the hand movements of the sketch character precisely correspond to those in the driving videos.

Figure 5.6(b) illustrates the appropriate handling of self-occlusion as the broom passes behind the character's head without any distortion or artifact. In the quadruped sample, the deer's motion aligns with the exemplar horse's movement in Figure 5.6(c). Additionally, illustrated in Figure 5.6(d), the motion of a horse's legs is depicted, showcasing its running movement. In non-living sketches, Figure 5.7(a), the lamp exhibits jumping and bending motions in accordance with the hand movements of a bear. In another scenario (see Figure 5.7(b)), the motion of tree leaves mirrors the similar motion style of the tree in the driving video.

The method of Siarohin et al. [156] uses keypoint transformation for animation, which fails in cross-domain motion transfer and generates artifacts during sketch animation. In Figure 5.8, the hand motion is distorted during the deformation, and the motion consistency is not stable. Similarly, the leg motion of the quadruped character distorts and collapses during animation, as depicted in Figure 5.9. The approach of Smith et al. [164] is close to SketchAnim and generates promising results, but it is limited to some kind of motion (biped motion), whereas SketchAnim seamlessly handles other motion categories very well. In Figure 5.8, the bat moves behind the character's body; it generates the artifact during deformation. On the other hand, SketchAnim maintains appearance consistency and smooth deformation. Also, Figures 5.8 and 5.9 show qualitative comparison with these methods. The subtle motion of body parts in the characters is very well captured by SketchAnim, whereas the other two fail to model these. Occlusion cases are also handled very well with SketcAnim.

5.4.2 Quantitative evaluation

Quantitative evaluation is conducted on 15 bipeds, 15 quadrupeds, and 10 inanimate sketch samples. SketchAnim demonstrates higher efficiency and accuracy

compared to other state-of-the-art techniques. Table 5.1 shows a quantitative comparison between these methods and SketchAnim. SketchAnim achieves a better FID score (lower is better), indicating higher quality in the generated sketches. Additionally, the proposed method demonstrates a superior performance on the AED metric, reflecting its robustness and capability. AnimationDrawing [164] supports only biped animation and is therefore not applicable to quadruped and inanimate categories. For these two motion categories, SketchAnim is compared against FOMM [156]. Table 5.1 presents the comparison results, demonstrating that SketchAnim outperforms FOMM. The proposed approach is more robust and accurate for motion transfer, with reduced distortion and precise motion estimation, especially in handling occlusions.

Table 5.1: Quantitative comparison with state-of-the-art methods.

Method	**Biped**		**Quadruped**	**Inanimate**
	FID↓	AED↓	FID↓	FID↓
Siarohin et al. [156]	186.37	0.4826	160.01	–
Smith et al. [164]	201.65	0.4775	–	–
SketchAnim	**150.25**	**0.4430**	**140.00**	**176.89**

5.5 Limitations and future work

The proposed framework has limitations, such as it fails to handle motion without correspondence between the video object and the sketch. If the skeleton topologies differ, it cannot transfer human motion to an animal or vice versa. Additionally, if details are missing from the input sketch, SketchAnim will not autocomplete them during deformation. SketchAnim struggles to handle frontal motion, such as a human moving toward the camera's direction (see Figure 5.10(a)). In the rest pose, the body part should not be self-occluded, as the tracking method cannot handle such occlusion. In some cases where the body parts may be occluded in the rest pose, the generated output tends to have distortion, as shown in Figure 5.10(b), where the legs of the horse are occluded in the rest pose, which leads to the artifacts in the animated frames. Error propagation poses a challenge in this method; if tracking fails or a shape mismatch occurs in the initial frames, errors will accumulate in the final animated sequence. Additionally, incorrect skeleton mapping might reorient the animated object incorrectly. In Figure 5.10(c), the tracking point of the fish's mouth shifts toward the wing, distorting the final animated sketch.

Future work will focus on improving motion extraction by incorporating 3D context to address the aforementioned limitations. A keypoint-based tracking approach can be developed to estimate the motion details (such as co-part motion) to enhance performance. Furthermore, depth cues and camera parameters can

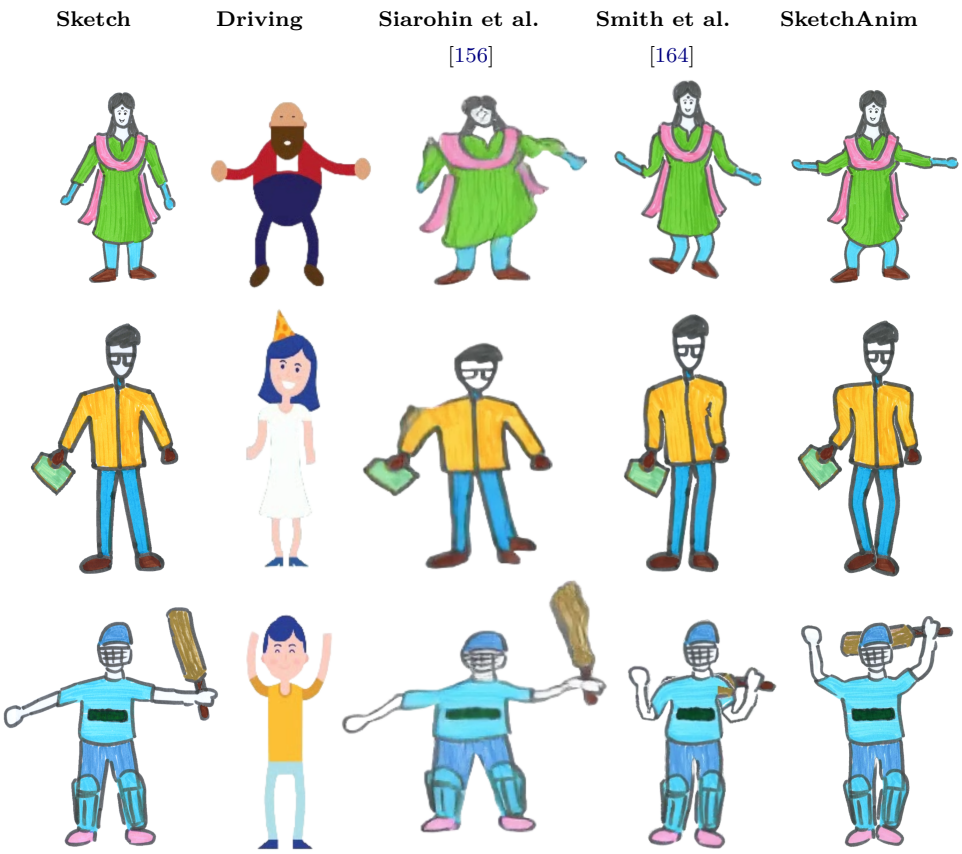


Figure 5.8: Comparison of biped samples with the state-of-the-art methods.

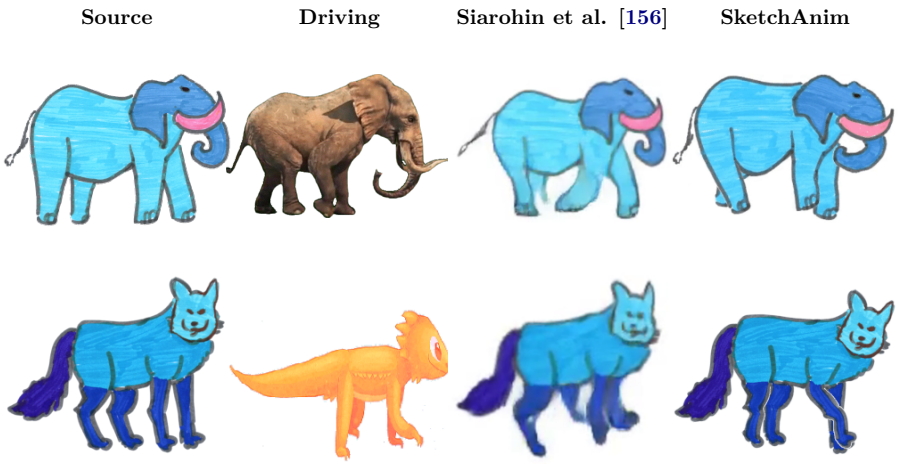


Figure 5.9: Comparison of quadruped animation with the state-of-the-art methods.

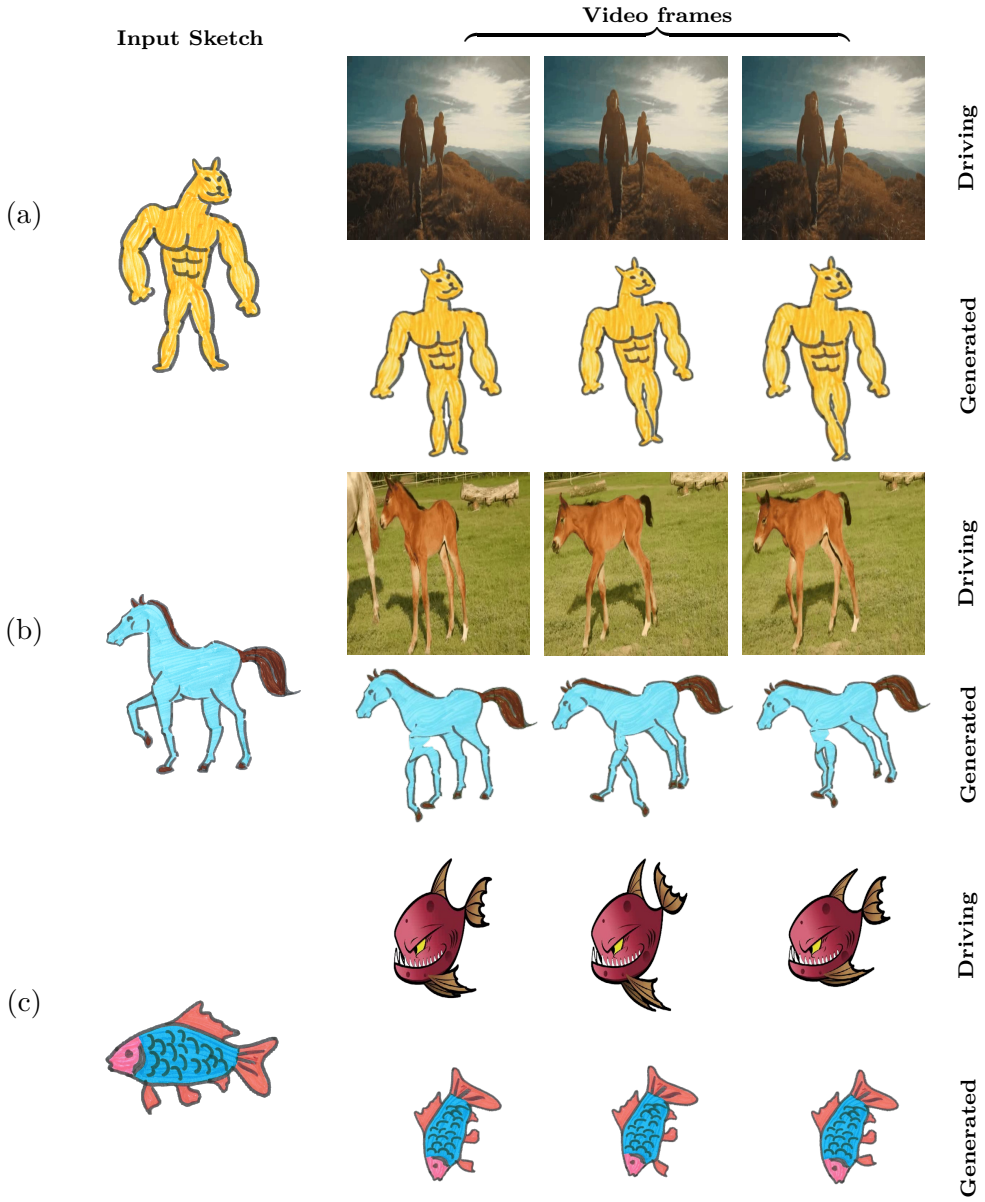


Figure 5.10: Failure cases such as (a) frontal motion, (b) self-occlusion in rest pose, and (c) error propagation in shape mapping.

be integrated to enable 3D handling capabilities. An automatic skeleton extraction and matching approach can be explored to enhance the proposed method for efficiency and robustness.

5.6 Conclusions

This chapter presents a method for animating a static sketch using a reference video. Users have the flexibility to choose a specific animation for the sketch, providing a direct reference video input. Furthermore, this method requires minimal user input and provides motion controllability: drawing a skeleton on a video frame. This approach enables the animation of sketches with arbitrary shapes, removing restrictions to specific categories such as biped or quadruped. SketchAnim successfully captures subtle motion from the given video and outperforms similar methods in animation quality. While the proposed framework exhibits notable strengths, it does have certain limitations. For instance, it may encounter challenges in handling motion when there is no direct correspondence between the video object and the sketch. Tracking failures can arise in complex motion scenarios within the video and instances of occlusion. In such cases, errors may propagate in skeleton mapping and motion transfer. Although SketchAnim effectively handles self-occlusion cases, it does not autonomously complete missing sketch details in specific scenarios when there is overlap between parts of the body.

Chapter 6

Enhancing Sketch Animation: Text-to-Video Diffusion Models with Temporal Consistency and Rigidity Constraints

6.1 Introduction

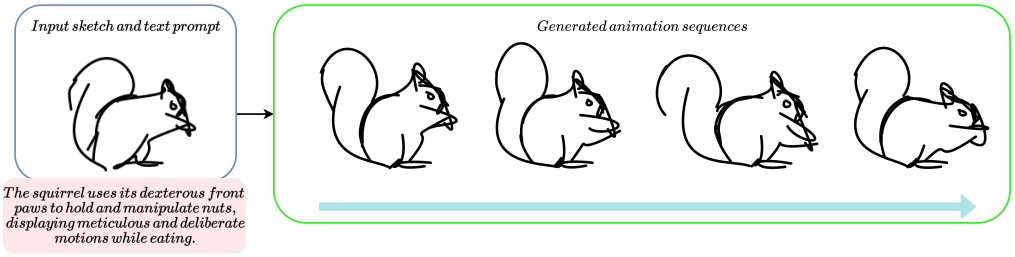


Figure 6.1: Problem with the LiveSketch [53] method. This example illustrates the lack of temporal consistency and the shape distortions that occur during motion.

Sketches serve as a medium for communication and visual representation. Animating 2D sketch illustrations using traditional tools is tedious, cumbersome, and requires significant time and effort. Keyframe-based animation is highly labor-intensive, while video-driven animation methods are often restricted to specific motions. In recent years, sketch animation has emerged as a significant area of research in computer animation, with applications in video editing, entertainment, e-learning, and visual representation. Previous sketch animation methods, such as that in [138, 212], require extensive manual input and artistic skill, presenting challenges for novice users. Traditional methods are limited to specific types of motion, such as facial and biped animation. More recent techniques,

like Su et al. [170], animate sketches based on a video, but still require manual input. AnimationDrawing [164] is a sketch animation technique that does not rely on manual input, generating animations using pose mapping, but is limited to biped motion. In contrast, LiveSketch [53] is a learning-based approach that takes a sketch and text prompt to produce an animated sketch. While it generates promising results, it faces challenges with temporal consistency and shape preservation (see Figure 6.1). To address these issues, this chapter presents Enhancing Sketch Animation (ESA), a method for animating input sketches based solely on a text description, with no manual input required. This method represents each stroke as a Bézier curve, similar to LiveSketch [53], and extends LiveSketch's capabilities by a novel Length-Area minimization and rigidity loss. Furthermore, local and global paths are utilized for motion estimation and apply Score Distillation Sampling (SDS) loss [139] for optimization. A length–area (LA) minimization objective is introduced to preserve temporal consistency and produce smooth, accurate motion in the animated sketch. Further, the As-Rigid-As-Possible (ARAP) loss [82] preserves local rigidity in the sketch's shape during animation. The proposed method outperforms state-of-the-art techniques in both quantitative and qualitative evaluations. This method achieves better sketch-to-video consistency and text-to-video alignment compared to the previous method. The main contributions of this chapter are as follows:

- A Length-Area minimization objective to maintain temporal consistency across animated sequences and generate smooth animations.
- A shape-preserving ARAP loss to preserve local rigidity in sketch strokes during animation. The rigidity loss overcomes shape distortion during animation.

6.2 Preliminaries

LiveSketch [52] addresses the challenge of automatically animating static, single-subject hand-drawn sketches without requiring extensive manual intervention, such as user-defined skeletal rigging, motion exemplars, or domain-specific training. Rather than operating in raster space, the framework represents sketches directly in the vector domain as a collection of k strokes, each parameterized by a cubic Bézier curve $f(u)$. The control points of a single frame are collected as $\mathcal{B} = \{p_j\}_{j=1}^{4k}$ with $p_j \in \mathbb{R}^2$, and an animation of n frames is defined by the set of moving control points $\mathcal{Z} = \{\mathcal{B}_i\}_{i=1}^{n}$, where $\mathcal{Z} \in \mathbb{R}^{4k\times n\times 2}$. This vector-based representation is inherently resolution-independent, enabling high-quality editing and animation while preserving the structural characteristics of the original drawing. Unlike raster-based approaches that often suffer from pixelization, aliasing, and blurring artifacts under repeated geometric transformations, the

sparse vector formulation maintains crisp stroke boundaries. It provides a mathematically continuous representation of temporal deformation.

To preserve structural integrity throughout the animation sequence, LiveSketch explicitly parameterizes the temporal evolution of each control point. The position of a control point $p_{i,j}$ at frame i evolves as a disentangled motion model,

$$p_{i+1,j} = M_i\, p_{i,j} + \Delta p_{i,j},$$

where M_i denotes the global transformation matrix estimated for frame i, encoding scaling, shear, rotation, and translation, and capturing the overall object trajectory, while $\Delta p_{i,j}$ represents the local control-point displacements responsible for fine-grained articulations and non-rigid deformations. The two terms are produced by separate branches: a global motion predictor M_g, which regresses the transformation parameters, and a local motion predictor M_l, implemented as an MLP F_θ that outputs unconstrained offsets. This decomposition separates coarse object motion from subtle stroke-level dynamics, thereby producing temporally coherent animations while preserving the topological consistency and visual identity of the input sketch.

To bridge the gap between continuous vector graphics and neural video generation models, the evolving Bézier strokes are projected into a raster video sequence through a differentiable rasterizer $\mathcal{R}$, such that

$$x_{1:n} = \mathcal{R}(\mathcal{Z}),$$

which allows gradients to propagate seamlessly from the image domain back to the vector control points. Instead of learning motion from paired sketch-video datasets, LiveSketch uses the rich temporal priors embedded in a large pre-trained text-to-video (T2V) diffusion model to synthesize animations directly from user-provided text prompts. This training-free paradigm eliminates the need for costly data collection while leveraging the semantic understanding and motion knowledge already encoded in modern diffusion models. Motion generation is formulated as an optimization over the motion parameters $\phi = \{M_{1:n}, \Delta\mathcal{Z}\}$, which are iteratively refined using the Score Distillation Sampling (SDS) objective conditioned on a text prompt y. During optimization, Gaussian noise ϵ is added to the rasterized video at diffusion timestep γ to obtain the noisy sequence x_γ. The predicted diffusion score is back-propagated through the differentiable rasterizer to update the vector control points, enabling the synthesized motion to align with the semantic intent of the textual description. By combining vector-based geometric parameterization with diffusion-based semantic supervision, LiveSketch generates smooth, expressive, and text-driven animations while preserving the original topology, stroke connectivity, and visual fidelity of the input sketch.

6.3 Methodology

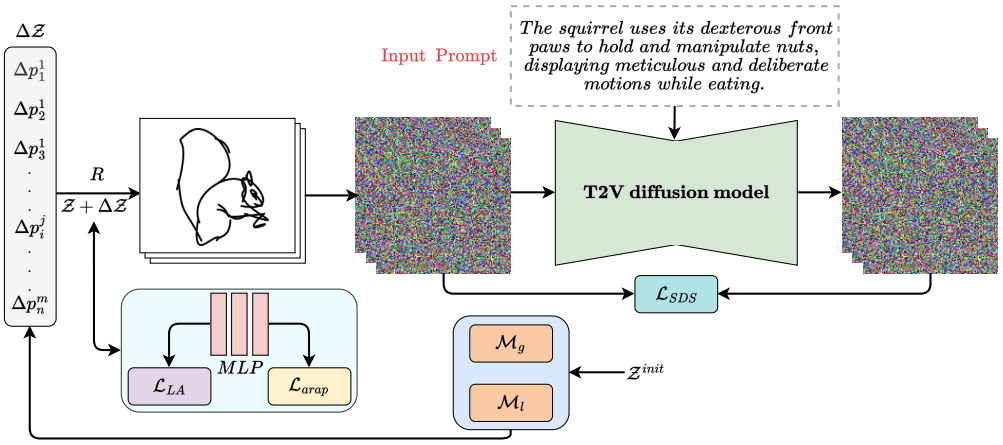


Figure 6.2: Network architecture of the proposed framework. The framework takes an initial set of control points, $\mathbf{Z}^{\text{init}}$, as input and predicts the corresponding displacement field, $\Delta\mathbf{Z}$, for the local ($\mathcal{M}_l$) and global ($\mathcal{M}_g$) paths under the guidance of a frozen text-to-video (T2V) diffusion model. The Score Distillation Sampling (SDS) loss, $\mathcal{L}_{\text{SDS}}$, aligns the predicted motion with the input text prompt by backpropagating gradients through the differentiable rasterizer R. For each frame, a Length-Area loss ($\mathcal{L}_{LA}$) is used to maintain temporal consistency and avoid drastic shape changes. Further, ARAP loss ($\mathcal{L}_{arap}$) maintains the sketch stroke's rigidity and prevents shape distortions during motion.

This methodology extends the framework introduced by Gal et al. [53], which produces animations from sketches guided by textual descriptions. Each sketch consists of a set of strokes, represented as cubic Bézier curves. The set of control points within a frame is represented as $\mathrm{B} = \{\mathbf{p}_i\}_{i=1}^{4k}$ where $\mathbf{p}_i \in \mathbb{R}^2$ and k is the total number of strokes. Further, a sketch video comprising n frames is represented by a set of moving control points $\mathrm{Z} = \{\mathrm{B}_i\}_{i=1}^{n}$, where $\mathrm{Z} \in \mathbb{R}^{4k\times n\times 2}$.

Animation of a sketch requires the user to provide a text prompt, which is passed to the network as input along with the sketch. Similar to LiveSketch [53], it takes an initial set of control points, Z^{init}, as input and produces the corresponding set of displacements, $\Delta\mathrm{Z}$. For each frame, Z^{init} is initialized to the set B. Each control point is first projected onto a latent space using a mapping function, $g_{shared} : \mathbb{R}^2 \to \mathbb{R}^{\mathbb{D}}$. This function takes the initial point set $\mathrm{Z}^{init} \in \mathbb{R}^2$ and projects it into a higher-dimensional space enriched with positional encoding, thereby generating point features. These features are processed through two branches: a local motion predictor, $\mathcal{M}_l$, implemented as a multi-layer perceptron (MLP) F_θ, which computes unconstrained local motion offsets, and a global motion predictor, $\mathcal{M}_g$, which estimates transformation matrices $\mathbf{M}_i$ for scaling, shear, rotation, and translation, yielding the global motion offsets T_i. The

generated animation sequence suffers from a lack of temporal consistency and degradation of sketch identity during motion. To address these issues, a novel length–area (LA) minimization framework is introduced to improve temporal coherence and preserve sketch identity throughout the animation. The proposed approach estimates the Bézier curve length and the area between consecutive frames. Furthermore, a shape-preserving As-Rigid-As-Possible (ARAP) loss is used, leveraging a mesh constructed via Delaunay triangulation [35] of control points in each frame. Unlike existing methods, ARAP loss is explicitly designed to maintain local shape consistency, thereby addressing critical challenges in handling deformation and ensuring robust animation fidelity. Figure 6.2 provides a detailed illustration of the proposed network architecture, highlighting its key components. The performance of the proposed method is evaluated using different learning-rate configurations and multiple optimization iterations to systematically refine the model.

6.3.1 Length-Area minimization

The LA loss is designed to minimize abrupt changes in stroke lengths between consecutive frames, ensuring smoother transitions and preserving structural consistency by maintaining stable stroke lengths across the animation. To mitigate error propagation, the length minimization for a stroke in a given frame is computed relative to its length in the initial frame. The stroke length is estimated as the curve length, $L = \int_0^1 |\dot{\mathbf{f}}(u)|\,\mathrm{d}u$ of the Bézier curve $\mathbf{f}(u)$.

Bézier curves offer limited local control, as even small changes in control point positions can cause substantial variations in the resulting curve. To mitigate this issue, an area loss term is introduced to minimize the area swept by a stroke between consecutive frames, thereby improving temporal stability and reducing undesirable deformations. To compute this area, a stroke in an intermediate frame i is represented by the Bézier curve $\mathbf{f}_i(u)$ defined by the control points $\mathbf{p}_{i,j}$, where $j = \{0 \dots 3\}$, for an intermediate frame i. Let the estimated global transformation matrix of control points for frame i be denoted as $\mathbf{M}_i$, and the corresponding local motion offsets as $\Delta\mathbf{p}_{i,j}$. The control points for the next frame are determined as $\mathbf{p}_{i+1,j} = \mathbf{M}_i\mathbf{p}_{i,j} + \Delta\mathbf{p}_{i,j}$. The space-time Bézier surface $\mathbf{f}(u,t)$ for $t \in [t_i, t_{i+1}]$ is defined by time-varying control points $\mathbf{p}_j(t) = \mathbf{M}(t)\mathbf{p}_{i,j} + \Delta\mathbf{p}_{i,j}(t - t_i)/(t_{i+1} - t_i)$, where $\mathbf{M}(t)$ is obtained by interpolating the transformation parameters appropriately over time. The surface area swept by the stroke between frames i and $i+1$ is computed as

$$A_i = \int_{t_i}^{t_{i+1}} \int_0^1 \left\| \frac{\partial \mathbf{f}}{\partial u} \times \frac{\partial \mathbf{f}}{\partial t} \right\| \mathrm{d}u\,\mathrm{d}t.$$

The LA loss, denoted as $\mathcal{L}_{LA}$ is defined as

$$\mathcal{L}_{LA} = \sum_{i=0}^{n-1} \left(\lambda_l \left| L_{i+1} - L_i \right| + \lambda_a A_i \right), \tag{6.1}$$

where $\mathcal{L}_{LA}$ represents the length-area loss function. This formulation aims to minimize both the variation in stroke length and the swept area between consecutive frames, ensuring temporal coherence and stability in animation. The loss is minimized with respect to the parameters of an MLP by gradient descent, with values of hyperparameters λ_l and λ_a set to 0.1 and $1e-5$, respectively.

LiveSketch [53] uses the SDS loss to train its model, which has separate blocks for optimizing the global and local motion. The SDS loss is defined as

$$\bigtriangledown_\phi \mathcal{L}_{sds} = \left[w(\gamma)(\epsilon_\theta(x_\gamma, \gamma, y) - \epsilon) \frac{\delta x}{\delta \phi} \right], \tag{6.2}$$

where $\epsilon_\theta(x_\gamma, \gamma, y)$ is the output of the diffusion model, ϵ denotes the actual noise, γ represents the timestep, and $w(\gamma)$ is a constant term that depends on the noising schedule. The SDS loss is calculated at every step of the diffusion generation process for all frames, guiding the training of these blocks and the overall generation process. During each generation step, the optimization occurs after completing the SDS loss-based optimization for both blocks of the base model, resulting in updated control points. These control points are used as input for the optimization procedure.

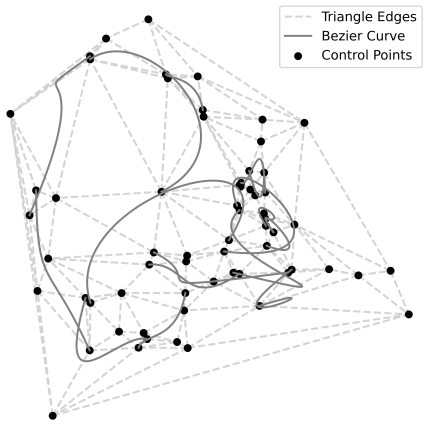


Figure 6.3: Cubic Bézier curve of each stroke and their corresponding control points. Delaunay triangulation of Bézier control points.

6.3.2 Shape preservation

As-rigid-as-possible deformation enables point-driven shape deformation by moving anchor points, which act as constraints within the model. This deformation

Table 6.1: Comparison with state-of-the-art methods.

	Sketch-to-video consistency (↑)	**Text-to-video alignment** (↑)
VideoCrafter1 [22]	0.7064	0.0876
LiveSketch [53]	0.8287	0.1852
ESA	**0.8561**	**0.1893**

framework maintains the rigidity of each element of the mesh as closely as possible, ensuring that transformations are smooth and visually coherent. The ARAP method leverages a two-step optimization algorithm. In the first step, an initial rotation is estimated for each triangle in the mesh. This involves computing the optimal rotation matrix that best approximates the transformation required to map the vertices of each triangle from their initial positions to their target positions while minimizing distortion. The second step involves adjusting the scale, ensuring that the transformation adheres to an as-rigid-as-possible model by minimizing the amount of stretch that would distort the original shape. The approach minimizes distortion across the triangular mesh by optimizing each triangle's local transformations while maintaining global consistency across the mesh.

In this proposed approach, the standard ARAP loss [82] is extended by formulating it as a differentiable function, thereby enabling gradient-based optimization and backpropagation within the network. This differentiable ARAP, with respect to the deformed control points, is minimized, enabling adaptive and flexible shape deformation.

The ARAP loss is computed based on a global mesh structure formed by triangulating Bézier control points (see Figure 6.3) within each frame. Calculating the ARAP loss relative to a similar triangulated mesh for the next frame ensures stroke preservation, which is essential for generating smooth and consistent animations. The ARAP loss $\mathcal{L}_{ARAP}$ is computed by identifying all triangles in the mesh formed by the control points of a given frame, with the triangulation topology $\mathcal{T}$ remaining fixed across all frames. The same is defined as

$$\mathcal{L}_{ARAP} = \lambda_{arap} \sum_{e \in \mathcal{T}} \alpha_e \left\| e' - \mathbf{D}e \right\|^2, \tag{6.3}$$

where $\mathbf{D}$ is the ARAP transformation matrix, e represents an edge of a triangle, estimated from the control points of the initial sketch, and e' denotes the corresponding deformed edge of the triangle of the subsequent frames. α_e denotes the weight, usually proportional to the edge length.

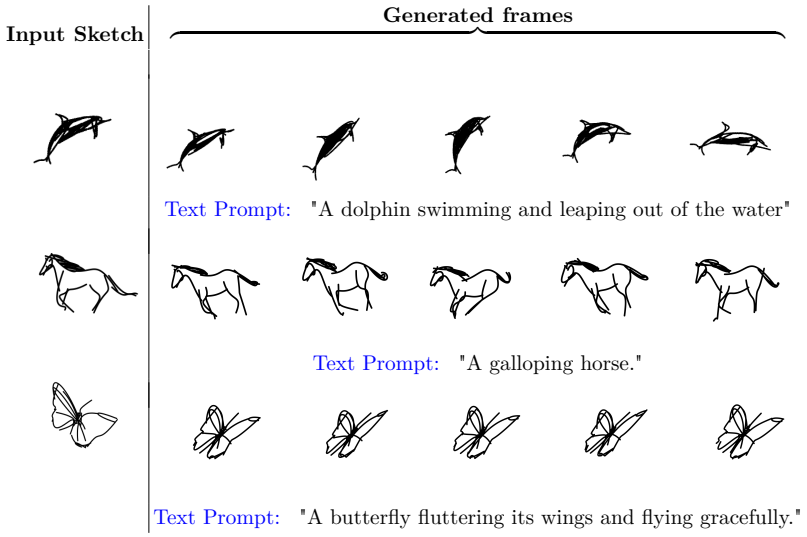


Figure 6.4: Qualitative results of the proposed method and generated animated sketch sequences from input text prompts.

6.4 Experiments and results

6.4.1 Implementation details

A text-to-video diffusion model [188] is used, as in LiveSketch [53], to generate the required motion in pixel space. Furthermore, the generated frames are used to apply SDS loss training at a given timestep to update the control points. These updated control points are then further optimized using the learning procedure. It takes the updated control points of the current frame as input and outputs the optimized control points. The model is optimized for 1,000 iterations, with λ_l, λ_a and λ_{arap} as 0.1, $1e-5$, and 0.1 respectively. Furthermore, the same parameter settings used in LiveSketch [53] are adopted for both the local and global paths. The proposed method takes approximately 2 hours to generate a sequence of 24 animated sketches.

6.4.2 Results and comparison

Quantitative evaluation

The proposed approach is compared with previous baseline methods, including VideoCrafter1 [22], and LiveSketch [53]. Sketch-to-video consistency and Text-to-video alignment are used as evaluation metrics, similar to LiveSketch [53], which uses CLIP [141] to estimate sketch-to-video consistency and X-CLIP [133] for text-to-video alignment. For quantitative evaluation, 20 unique sketch samples and their corresponding text prompts are used. VideoCrafter1 [22] is an

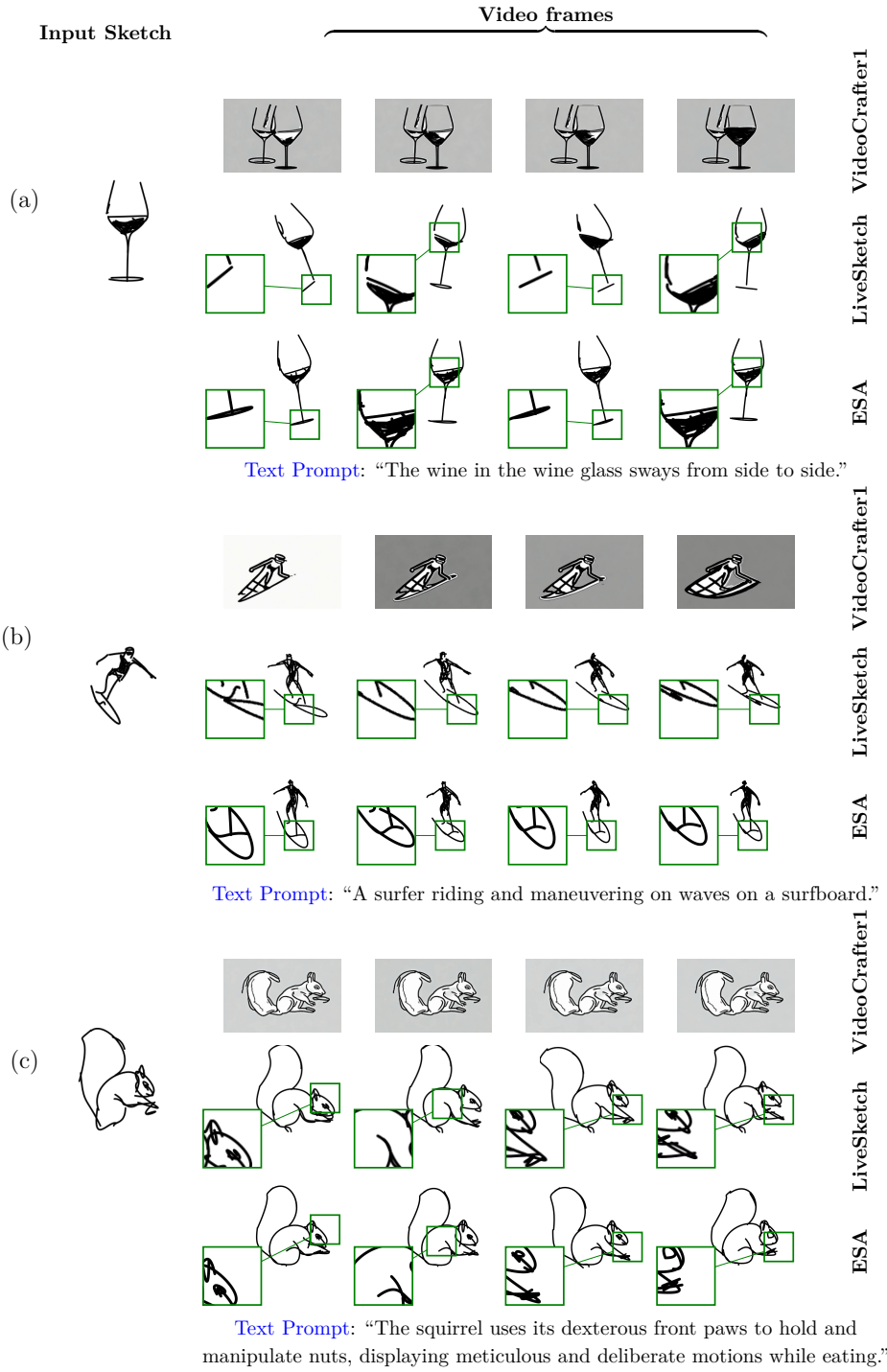


Figure 6.5: Comparison with state-of-the-art methods (VideoCrafter1 [22] and LiveSketch [53]). In the figure above, the base of the wine glass is distorted in the previous methods, and in the surfer base, the original shape is missing compared to the proposed method (ESA). The squirrel's tail and body shape preserve the original topology in the proposed method.

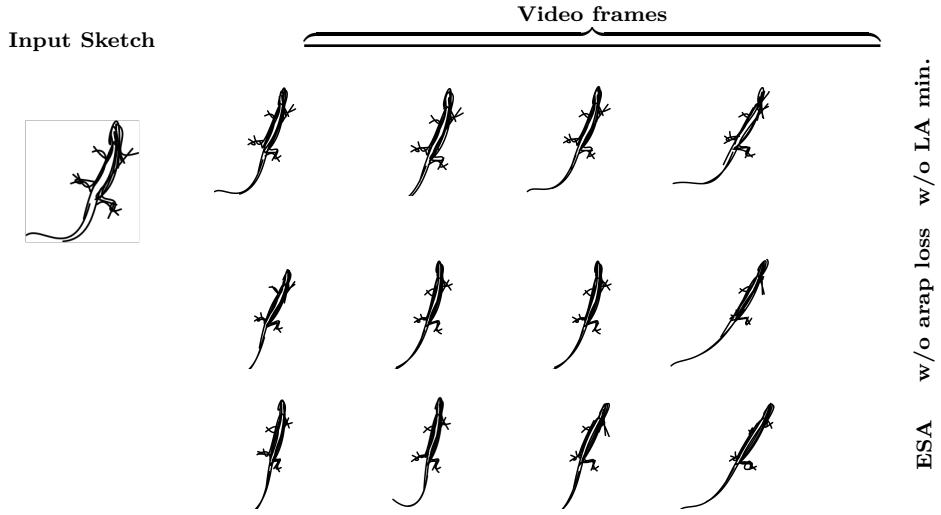


Figure 6.6: Ablation study on different setting such as w/o LA minimization, w/o shape preserving ARAP, and the full method.

image-to-video generation model and the conditions on image and text prompts. Table 6.1 depicts that the proposed method outperforms the quantitative analysis compared to previous methods.

Qualitative evaluation

In the qualitative comparison, sketch-to-video consistency and text-to-video alignment are evaluated. Figure 6.4 demonstrates improvements in temporal consistency and shape preservation. Sketch-to-video consistency refers to the temporal consistency of generated sketch sequences. Figure 6.5 shows that the bottom of the wine glass and squirrel are temporally consistent in all the frames compared to VideoCrafter1 [22], and LiveSketch [53]. On the other hand, the surfer and squirrel examples demonstrate shape preservation throughout the animation. The mesh-based rigidity loss helps produce smoother deformation compared to baseline methods. The proposed method maintains temporal consistency, preserves shape during animation, and outperforms baseline methods.

6.4.3 Ablation study

With and without LA minimization

The proposed method is evaluated without LA minimization and is found to fail to maintain temporal consistency. LA minimization helps to address the

Table 6.2: Ablation results—the quantitative evaluation on w/o LA minimization, w/o shape preserving ARAP, and the proposed method.

	Sketch-to-video consistency (↑)	**Text-to-video alignment** (↑)
w/o LA minimization	0.8306	0.1864
w/o Shape-preserving ARAP	0.8489	0.1891
ESA	**0.8561**	**0.1893**

issue of drastic changes in stroke. In Figure 6.6, the lizard's tail and legs move rapidly, and the stroke length varies excessively. The proposed method produces smooth motion with minimal changes in stroke length. Table 6.2 shows the quantitative results w/o the LA minimization and the proposed method with the LA minimization.

With and without shape-preserving ARAP

The proposed method is evaluated without shape preservation and observes shape distortion during animation. The animated sketch shows distortion when local motion increases, as topology is not preserved in the animated sketch video. In Figure 6.6, the lizard body distorts during the motion, compared to the proposed method with shape-preserving ARAP loss. Quantitatively (see Table 6.2), the performance without shape preservation is comparable, but the complete method gives better results.

6.5 Limitations

The proposed method relies on a pre-trained text-to-video prior [239], which may struggle with certain types of motion, leading to errors that propagate and manifest as noticeable artifacts in some cases of the generated animations. Improvements could be made by employing more advanced text-to-video priors capable of handling text-to-video alignment with higher accuracy. Additionally, this approach faces challenges in animating multi-object scenarios, particularly when functional relationships exist between objects. Designed primarily for single-object animations, the method experiences a decline in quality when dealing with such cases. For example, as shown in Figure 6.7, the human and the bicycle are incorrectly separated, resulting in unnatural motion during the animation. Future work could address this limitation by implementing object-specific translations rather than relying on relative motion.

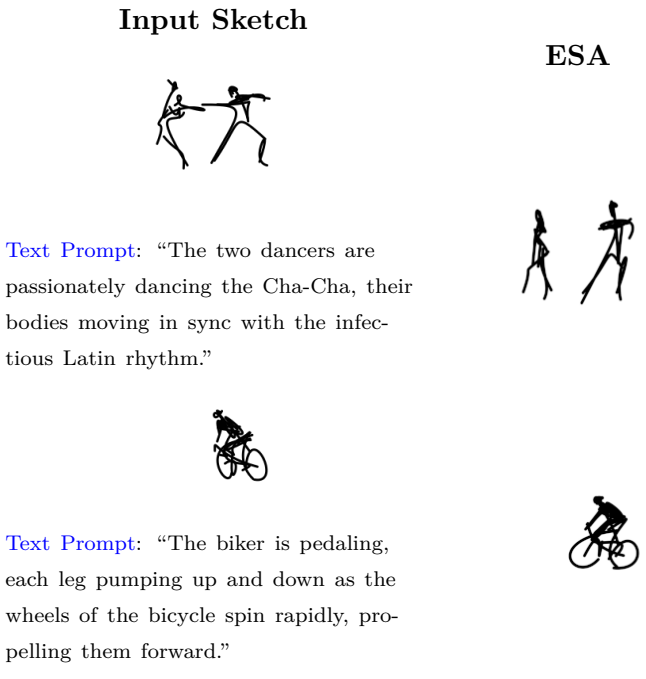


Figure 6.7: Failure cases.

6.6 Conclusion

This work presents a method for generating animated sketches from a combination of sketch inputs and text prompts. To ensure temporal consistency in the animations, a Length-Area (LA) minimization objective is introduced, while a shape-preserving ARAP loss is utilized to preserve the topology of the original shape. The proposed approach delivers superior performance both quantitatively and qualitatively, addressing challenges in animation generation. However, the method has certain limitations, including its inability to handle multi-object scenarios and its reliance on a pre-trained text-to-video prior. Future work will focus on addressing these limitations to further enhance the method's capabilities.

Chapter 7

Sketch2Motion: Text-driven 2D Sketch to 3D Animation via Diffusion-guided Skeleton Optimization

7.1 Introduction

Animating 2D hand-drawn sketches poses technical challenges arising from the inherent ambiguity and sparsity of 2D representations. Unlike photographs or 3D scans, sketches lack explicit depth cues, surface geometry, and consistent topology, making it difficult to infer plausible 3D structure and motion. The primary challenge lies in estimating and simulating complex 3D motion from inherently 2D inputs, which requires bridging a significant dimensional gap while preserving the artistic intent, stylization, and abstraction characteristic of hand-drawn content. Minor errors in depth inference or articulation can easily lead to visually implausible or artistically inconsistent animations. Another major challenge arises from occlusions or self-intersections commonly present in sketches, where

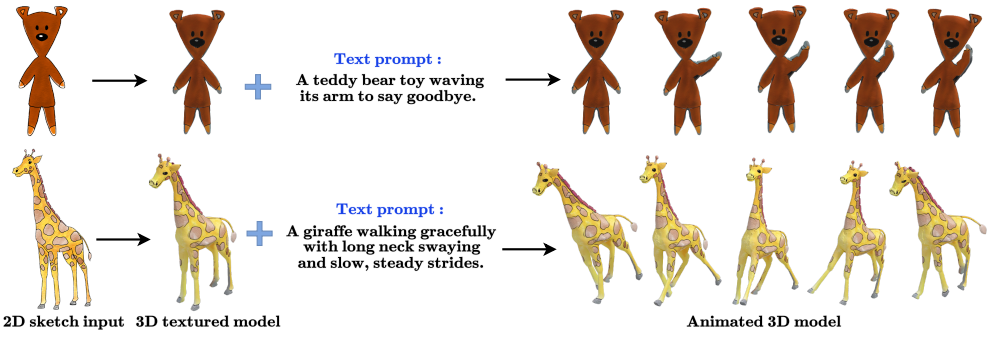


Figure 7.1: Overview of the proposed method for 2D sketch to 3D animation.

overlapping strokes may represent different body parts or structural components. Such occlusions introduce ambiguity in part ordering and connectivity, complicating both motion estimation and temporal coherence during animation. Additionally, sketches often exhibit varying levels of detail and incomplete contours, further increasing uncertainty in part segmentation and motion interpretation. As a result, many existing sketch animation methods [142, 170] are constrained to simple motions or rely on strong assumptions about object structure and pose.

Traditional approaches to sketch animation have therefore been limited to specific motion categories [164] or predefined templates [65, 138], restricting their applicability to a broader range of characters and motion styles. To address these limitations, recent methods for 2D sketch-to-3D animation [45, 195, 221, 236] maintain the texture style of drawing but rely on template-based motion for animation. Animating 3D characters with realistic motion remains a challenging problem. Classical motion capture pipelines rely on expensive hardware setups and carefully curated datasets. Moreover, adapting captured motion to new characters requires non-trivial retargeting procedures. Learning-based motion synthesis methods offer a promising alternative. However, they typically depend on large-scale paired datasets for training. In addition, these methods often struggle to generalize across varying skeletal topologies and non-standard character shapes. These limitations are particularly apparent when animating sketch-derived or stylized 3D models, for which large-scale motion datasets are unavailable. Recent advances in diffusion models have demonstrated impressive performance in image and video generation, suggesting their potential as strong priors for motion synthesis [139]. Text-to-video (T2V) diffusion models, in particular, encode rich semantic and temporal information about motion patterns. However, directly applying diffusion models to 3D animation is non-trivial due to the high dimensionality of articulated motion spaces, the lack of paired supervision between text and skeletal motion, and the need to preserve physical plausibility and temporal topological consistency.

Recent, Text-to-3D model animation methods lack physics-based constraints [189, 206] and soft-body dynamics [110, 130] and are limited to a specific kind of motion [64]. To address these limitations, this chapter presents Sketch2Motion, a framework that begins by inflating the input 2D sketch into a 3D representation [109], and generates a 3D mesh that preserves the visual appearance and stylistic characteristics of the original sketch. This work bridges the classical animation pipelines and modern generative modeling by introducing a diffusion-guided, skeleton-based motion synthesis framework. It uses a pretrained T2V diffusion model [188] as a high-level semantic prior and explicitly optimizes skeletal motion parameters. Score distillation sampling (SDS) [139] is enhanced with motion-aware SDS (MoSDS), extracting gradients from the diffusion model to guide motion optimization toward text-aligned, visually plausible animations without updating the diffusion model itself. To ensure stability and realism, physics-inspired regularization terms are incorporated, including tem-

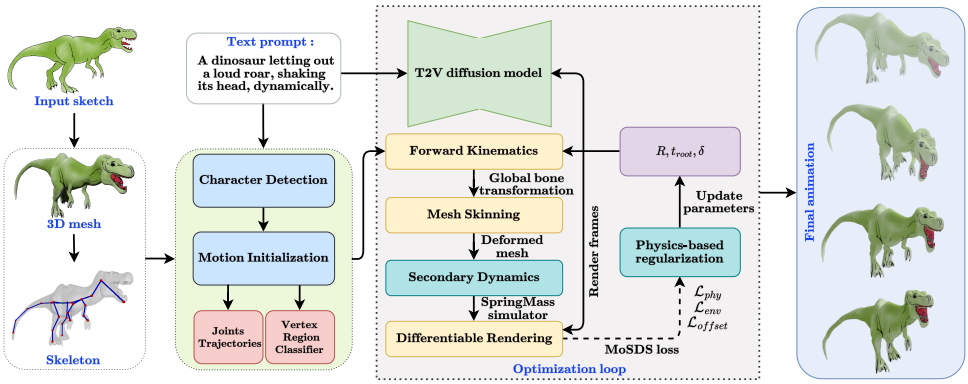


Figure 7.2: Overview of pipeline architecture of the proposed method for 2D sketch to 3D model animation.

poral smoothness, topological joint limits, symmetry constraints, and ground contact penalties. A spring-mass simulator is integrated to add secondary motion effects that make a character feel alive. The proposed framework is fully differentiable, does not require paired motion supervision, and generalizes across a wide range of character types, including bipeds, quadrupeds, flying, aquatic, and non-living objects. Together, these components provide a principled, extensible solution for controllable, realistic skeleton-driven motion synthesis as illustrated in Figure 7.1. The main contributions of this chapter are as follows:

- A diffusion-guided framework that directly optimizes skeletal motion using MoSDS, enabling text-driven animation without paired motion supervision.
- Kinematic NURBS conditioning and physics-inspired constraints for stable, coherent, and physically plausible motion optimization.
- A spring-mass model for secondary motion and a dataset of 100 hand-drawn sketches spanning diverse categories and structural complexities.

7.2 Methodology

7.2.1 Overview

A 2D sketch S is used as input, and first inflated into a textured 3D mesh. Thereafter, a kinematic skeleton and joint hierarchy are automatically inferred from the 3D geometry. The extracted skeleton defines joint locations, parent-child relationships, and a rest pose. Motion initialization is subsequently performed to identify the object class and determine an appropriate motion type, enabling the assignment of a valid range of motion for each joint type in the skeleton.

To generate animations, a text-to-video (T2V) diffusion model is conditioned on the textured mesh and the extracted skeleton. The skeletal parameters are optimized in the latent space of a pre-trained T2V diffusion model [188] guided by a user-specified text prompt describing the desired motion. Motion-aware score distillation sampling (MoSDS) is utilized to ensure semantically aligned, meaningful, and temporally coherent motion generation. By parameterizing the shape using a kinematic skeleton, the method constrains the generated motion to preserve structural rigidity while enabling flexible and controlled articulation. Rather than using the T2V model as a direct video generator, it serves as a differentiable semantic prior. Given the text prompt y, joint rotations are optimized over time such that the rendered sequence of the rigged mesh aligns with the learned motion priors of the diffusion model. This optimization-by-synthesis approach enables the generation of complex, non-linear motions that are semantically faithful to the text while remaining kinematically grounded in the 3D structure.

Further, to generate physically plausible motion, physics-based regularization and a spring-mass simulator are used to model secondary motion effects. Figure 7.2 illustrates the pipeline of the proposed method. Algorithm 5 provides a set of steps of the proposed method.

7.2.2 Skeleton representation

An articulated object is represented by a kinematic skeleton composed of J joints organized in a hierarchical tree structure with a root joint. Each joint j is defined by its rest-pose position and a parent index $p(j)$. To facilitate efficient kinematic computations, the rest-pose offsets (bone vectors) are precomputed as $\mathbf{o}_j = \mathbf{x}_j^{\text{rest}} - \mathbf{x}_{p(j)}^{\text{rest}}$ for all non-root joints, while the root joint's offset is defined relative to the global origin, where $\mathbf{x}$ denotes a joint's coordinate. The topological order of the hierarchy is computed via depth-first search to ensure that parent transformations are always processed before their children during the forward pass.

To generate dynamic, expressive animations, motion is parameterized over a sequence of T frames using three distinct sets of learnable variables. First, the global root translation $\mathbf{t}_{\text{root}} \in \mathbb{R}^{T\times 3}$ controls the character's global trajectory and positioning in world space. Second, joint rotations $\mathbf{R} \in \mathbb{R}^{T\times J\times 3}$ are represented in axis-angle form (rotation represented as $\theta\hat{\mathbf{n}}$ with angle of rotation $\theta \in \mathbb{R}$ and direction unit-vector $\hat{\mathbf{n}} \in \mathbb{R}^3$) and ensure smooth gradient flow. These define the relative orientation of each joint with respect to its parent. Finally, local offsets $\boldsymbol{\delta} \in \mathbb{R}^{T\times J\times 3}$ are introduced as per-joint translational residual that models non-rigid secondary dynamics that cannot be captured by only rigid skeletal rotation. Given the rest-pose offset $\mathbf{o}_j$ of joint j relative to its parent, a local rigid transformation $\mathcal{T}(\mathbf{R}_{t,j}, \mathbf{o}_j) \in \mathbb{R}^{4\times 4}$ is defined using homogeneous coordinates.

7.2.3 Motion initialization

T2V diffusion models provide strong semantic priors for motion synthesis but struggle to directly supervise 3D articulated motion from an uninitialized state. When skeletal parameters are optimized from scratch using SDS, the optimization landscape becomes non-convex and under-constrained, often resulting in degenerate local minima and severe visual artifacts such as floating geometry, foot-skating, jittery motion, and anatomically implausible joint configurations [3]. To address this limitation, a motion initialization module is introduced to provide a structured, physically plausible prior, thereby significantly stabilizing diffusion-guided optimization. The initialization is procedural and training-free, requiring neither motion-capture data, learned pose priors, nor category-specific models. Moreover, it is adaptive rather than template-based. A rig automatically generated from an amateur sketch may contain an arbitrary number of joints organized in an unknown hierarchy, with non-standard limb proportions and no consistent naming convention. Consequently, the proposed classifier infers anatomical roles directly from skeletal topology and geometric relationships, rather than matching the rig to a predefined reference skeleton.

This subsection fully specifies the anatomical prior, including how joints are assigned to anatomical categories, how these categories determine the joint limits used by the anatomical regularizer, and how these categories determine the joint limits used by the anatomical regularizer.

Morphological joint classification

The automatically generated skeletons derived from sketch-driven meshes may exhibit irregular topologies, arbitrary joint hierarchies, inconsistent naming conventions, and substantial variations in limb proportions. Consequently, traditional template-matching approaches that rely on predefined skeletons or semantic joint names cannot be applied reliably. To address these limitations, a topology-driven morphological joint classifier is utilized to assign an anatomical role to each joint based on the skeletal graph and its geometric configuration. Let the rig consist of J joints with rest positions $\mathbf{p}_j \in \mathbb{R}^3$ and parent indices $\pi(j)$, where $\pi(j) = -1$ denotes the root. The children of a joint are represented by $\mathcal{C}(j)$, while the set of leaf joints is denoted by $\mathcal{L}$. Since the reconstructed meshes may vary significantly in size, all vertical coordinates are first normalized, producing scale-independent heights $\hat{y}_j \in [0, 1]$. This normalization ensures that the classifier operates consistently regardless of the absolute dimensions of the inflated mesh. The classifier predicts one of eleven anatomical categories, namely *root*, *hip*, *upper_leg*, *lower_leg*, *foot*, *spine*, *body*, *head*, *arm*, *tail*, and *wing*, through a deterministic four-stage procedure in which *body* serves as the default residual category. Because labels assigned during earlier stages are preserved, the ordering of the stages establishes a clear precedence that ensures deterministic,

reproducible classification. The upward traversals required during classification are implemented using two variants of a tracing operator: TraceUp^{+} includes the first branching ancestor in the traced chain, making it suitable for recovering complete leg chains ending at the hip, whereas TraceUp^{-} excludes the branching ancestor, preventing torso joints from being incorrectly absorbed into head or arm chains. In both cases, traversal continues until the root or the designated stopping condition is reached, enabling robust analysis of skeletons with arbitrary connectivity.

The classification proceeds sequentially through four stages. First, all joints with no parent are assigned the *root* label, while the remaining joints are initialized as *body*. In Stage 1, leaf joints located below the normalized height threshold θ_{low} are treated as potential feet. Each candidate is traced upward using the inclusive TraceUp^{+} operator, and the resulting chain is labeled according to the fixed sequence *foot*, *lower_leg*, *upper_leg*, *hip*. Assigning labels according to ordinal position rather than absolute geometric distance makes the method robust to exaggerated limb proportions commonly found in freehand sketches and naturally accommodates incomplete or simplified skeletons. Stage 2 detects head structures by tracing upward from high leaf joints using the exclusive TraceUp^{-} operator, thereby preventing torso joints from being incorporated into the head chain while deliberately merging the neck into the *head* category because their motion characteristics are nearly identical for the animations considered in this work. Stage 3 refines the remaining unlabeled joints using any informative joint names when available, identifying *wing*, *tail*, or *arm* joints through case-insensitive substring matching, while shallow hierarchy nodes are classified as *spine*. Finally, Stage 4 identifies arm chains that cannot be recognized through naming alone by tracing intermediate-height leaf joints using topology, requiring both a minimum chain length and an upper-body attachment constraint to distinguish arms from lower-body structures. The complete classification procedure is summarized in Algorithm 4. Together, these stages provide a robust, anatomy-aware initialization procedure that remains effective despite inconsistent rig structures, missing joint names, and the large geometric variations inherent in sketch-derived character models.

Category-specific motion heuristics

Following morphological classification, each joint category is associated with a predefined set of motion heuristics that encode coarse biomechanical priors for procedural animation. Rather than learning motion from data, the proposed framework synthesizes plausible articulation by assigning category-specific rotation amplitudes, rotation axes, phase offsets, and frequencies to each joint. For cyclic locomotion, the rotation of joint j about axis $a \in x, y, z$ at normalized

ALGORITHM 4: Morphological joint classification

Input: Rest joints $\{\mathbf{p}_j\}_{j=1}^{J}$, parents π, names $\{n_j\}$
Output: Category labels $\{\tau(j)\}_{j=1}^{J}$

Initialization
$\hat{y}_j \leftarrow$ normalised; $\mathcal{L} \leftarrow \{j : \mathcal{C}(j) = \emptyset\}$
$\tau(j) \leftarrow$ `body` $\forall j$; $\tau(j) \leftarrow$ `root` where $\pi(j) = -1$
Step 1: Leg chains
foreach $\ell \in \mathcal{L}$ with $\hat{y}_\ell < \theta_{\text{low}}$ **do**
  $c \leftarrow \textsc{TraceUp}^{+}(\ell)$; `// branching ancestor included; root included if reached`
  if $|c| \geq 2$ **then**
    foreach i **do**
      if $\tau(c_i) =$ `root` **then**
        continue ; `// label kept, ordinal position consumed`
      end
      $\tau(c_i) \leftarrow \sigma_i$ if $i < |\sigma|$, else `body`
    end
    mark all c_i as leg-consumed
  end
end
Step 2: Head chain
foreach leg-unconsumed $\ell \in \mathcal{L}$ with $\hat{y}_\ell > \theta_{\text{high}}$ **do**
  $c \leftarrow \textsc{TraceUp}^{-}(\ell)$; `// branching ancestor excluded`
  $\tau(c_0), \tau(c_1) \leftarrow$ `head`; $\tau(c_{i\geq 2}) \leftarrow$ `body` ; `// skipping root and leg-consumed joints`
end
Step 3: Name and depth refinement
foreach j with $\tau(j) =$ `body` **do**
  $\tau(j) \leftarrow$ `wing` | `tail` | `arm` on first substring match in n_j
  else if $\text{depth}(j) \leq 1$ **then**
    $\tau(j) \leftarrow$ `spine`
  end
end
Step 4: Topological arm detection
foreach $\ell \in \mathcal{L}$ with $\tau(\ell) =$ `body` and $\theta^{-}_{\text{arm}} \leq \hat{y}_\ell \leq \theta^{+}_{\text{arm}}$ **do**
  $c \leftarrow \textsc{TraceUp}^{-}(\ell)$
  $q \leftarrow \pi(c_{|c|-1})$
  if $|c| \geq 2$ **and** $\big(q = -1$ **or** $\hat{y}_q > \theta_{\text{branch}}\big)$ **then**
    $\tau(c_i) \leftarrow$ `arm` for all i
  end
end
return $\tau(j)_{j=1}^{J}$.

time t is defined by,

$$\theta_{j,a}(t) = A\,\alpha_{\tau(j),a}\,\sin\big(\omega_{\tau(j),a}t + \phi_j + \delta_{\tau(j),a}\big),$$

where the global amplitude scale A controls the overall motion intensity, ϕ_j specifies the joint-dependent phase, and $\alpha_{\tau(j),a}$, $\delta_{\tau(j),a}$, and $\omega_{\tau(j),a}$ represent the amplitude coefficient, phase offset, and oscillation frequency associated with the predicted anatomical category $\tau(j)$. Most joints oscillate at the fundamental gait frequency ω; however, the vertical root motion, spinal undulation, and head stabilization during running naturally occur at twice the stride frequency (2ω) because they complete two oscillations per locomotion cycle. Contralateral coordination is achieved by assigning opposite phase offsets according to,

$$\phi_j = s_j\frac{\pi}{2}, \qquad s_j = \begin{cases} +1 & x_j > 0 \\ -1 & \text{otherwise,} \end{cases}$$

where the lateral position of a joint in the rest pose determines whether its motion leads or lags by π. Since the mesh inflation procedure consistently produces characters in a canonical orientation, the world x-axis coincides with the lateral direction. For rigs generated at arbitrary orientations, the inferred lateral axis should replace the fixed world axis to ensure correct synchronization of limb motion between the left and right limbs.

The motion heuristics encode fundamental biomechanical characteristics while remaining computationally efficient for procedural animation. Hip flexion provides the primary driving force for leg swing, knee motion is constrained to unidirectional flexion to prevent anatomically implausible hyperextension, and ankle plantarflexion is phase-shifted to reproduce realistic push-off. The arms swing in antiphase with the legs to maintain dynamic balance, the spine counter-rotates with respect to the pelvis to improve stability, and the root undergoes vertical oscillation at twice the gait frequency. Walking and running are represented by distinct parameterizations rather than simple amplitude scaling, allowing differences in joint coordination, phase relationships, and oscillation frequencies to be captured more faithfully. The same formulation is extended to quadrupedal, flying, aquatic, serpentine, insect, and non-living characters, each with category-specific heuristic parameters. Non-cyclic actions, including jumping, kicking, waving, and lifting, are modeled using phase-masked motion segments with predefined temporal boundaries and smooth easing functions to ensure continuous transitions between successive motion phases.

The anatomical categories predicted by morphological classification are further used to impose joint-specific rotational constraints in motion optimization. Instead of treating every joint uniformly, each anatomical category τ is assigned a maximum permissible rotation magnitude λ_τ, reflecting its expected range of motion. These limits are incorporated through the anatomical regularization, which applies a one-sided quadratic hinge penalty whenever the rotation

magnitude of a joint exceeds its assigned threshold. Here, $\mathbf{r} * t, j$ denotes the rotation parameter of joint j at frame t. Under the axis-angle representation, $|\mathbf{r} * t, j|$ directly corresponds to the rotation angle. Unlike a hard constraint that clips rotations and eliminates gradients beyond the admissible range, the proposed hinge formulation preserves differentiability by producing a restoring force that increases smoothly with the magnitude of the violation. Consequently, the optimizer is encouraged to recover from anatomically implausible poses while retaining sufficient flexibility to satisfy competing optimization objectives.

Joint trajectory / NURBS reparameterization

To initialize skeletal kinematics, dense frame-by-frame trajectories are generated for the joint rotations $\tilde{\mathbf{R}} \in \mathbb{R}^{T\times J\times 3}$ and root translation $\tilde{\mathbf{t}}_{\text{root}} \in \mathbb{R}^{T\times 3}$ using parameterized periodic functions, where T is the total number of frames and J is the total number of joints. Drawing inspiration from classic procedural animation and Fourier-based kinematic modeling [185], joint rotations for cyclic motions, such as walking and running, are modeled using sinusoidal primitives:

$$\tilde{\mathbf{R}}_{t,j} = \mathbf{A}_j \sin\Big(2\pi\omega t + \boldsymbol{\varphi}_j\Big) + \mathbf{b}_j, \tag{7.1}$$

where ω is the global temporal frequency dictating the overall cycle speed, while $\mathbf{A}_j$ (amplitude), $\boldsymbol{\varphi}_j$ (phase offset), and $\mathbf{b}_j$ (resting bias) are joint-specific parameters. These joint-specific parameters are explicitly retrieved from a predefined heuristic dictionary, which is mapped to the semantic action extracted from the user's prompt. This procedural generation naturally encodes key biomechanical behaviors. For example, assigning a phase difference of π between the left and right hip joints enforces phase-shifted limb coordination, while specific amplitude and bias tunings ensure periodic foot-ground contact and pelvis-spine counter-rotation. For non-cyclic or transient motions (e.g., jumping or striking), piecewise parametric curves with phase-dependent parameter scheduling are used to model standard acceleration and deceleration patterns.

While the dense trajectories $\tilde{\mathbf{R}}$ provide a meaningful initialization, directly optimizing in the high-dimensional space $\mathbb{R}^{T\times J\times 3}$ leads to temporal instability under SDS gradients. To implement temporal smoothness and reduce optimization complexity, the dense trajectories are projected onto a low-dimensional manifold represented by Non-Uniform Rational B-Splines (NURBS) [44]. For each joint j, the dense trajectory $\tilde{\mathbf{R}}_{:,j}$ is uniformly sampled to extract $K+1$ control points $\mathbf{c}_{j,k} \in \mathbb{R}^3$. The continuous rotational trajectory $\mathbf{R}_j(s)$ over normalized time $s \in [0,1]$ is then represented as follows:

$$\mathbf{R}_j(s) = \frac{\sum_{k=0}^{K} \bar{B}_{k,d}(s) w_{j,k} \mathbf{c}_{j,k}}{\sum_{k=0}^{K} \bar{B}_{k,d}(s) w_{j,k}}, \tag{7.2}$$

where $\bar{B}_{k,d}(s)$ denotes the standard B-spline basis function of degree d (cubic splines, $d = 3$), and $w_{j,k} \in \mathbb{R}^+$ are the strictly positive rational weights assigned to each control point. It provides a compact, differentiable representation that inherently enforces temporal smoothness and reduces high-frequency artifacts during optimization.

A key advantage of the NURBS formulation is the inclusion of the rational weights $w_{j,k}$, which are critical for modeling non-uniform kinematic transitions. Weights are assigned according to each joint's anatomical function and the motion phase. In particular, during foot-contact phases, significantly larger weights ($w_{j,k} \gg 1$) are assigned to the corresponding control points of the lower-extremity joints. This effectively pulls the interpolated curve closer to these specific points, creating the sharp transitions necessary to capture the impact dynamics of foot-ground interaction. In contrast, during swing phases, uniform weights ($w_{j,k} = 1$) are assigned to control points to encourage smooth, pendular motion. Similarly, torso and spine joints are assigned moderate weights to preserve stability while allowing subtle compensatory motion. The final continuous trajectories are evaluated via the standard Cox-de Boor recursion algorithm [30] to yield the initialized discrete global root translations $\mathbf{t}_{\text{root}}^{(0)}$ and joint rotations $\mathbf{R}^{(0)}$. The localized translation offsets (discussed in Sec. 7.2.4) are initialized strictly to zero ($\boldsymbol{\delta}^{(0)} = \mathbf{0}$). By conditioning the optimization loop on these low-dimensional NURBS parameters, Sketch2Motion ensures that the T2V diffusion model refines a physically plausible, structurally aligned motion prior rather than hallucinating kinematics from random noise. This significantly improves convergence stability, reduces artifacts, and leads to more physically consistent and semantically aligned animations.

7.2.4 Forward kinematics and skinning

To render the final animation, the learned local motion parameters, root trajectory, joint rotations, and non-rigid offsets should be mathematically mapped into global world coordinates. A differentiable forward kinematics layer propagates local motion parameters into global transformations. First, axis-angle rotations $\mathbf{R}_{t,j}$ are converted into rotation matrices $\mathbf{M}_{t,j} \in SO(3)$. A local transformation matrix $\mathbf{L}_{t,j} \in \mathbb{R}^{4\times4}$ is then constructed for each joint, with the learnable local offsets $\boldsymbol{\delta}_{t,j}$ incorporated directly into the translation component of this transformation to model secondary motion:

$$\mathbf{L}_{t,j} = \begin{bmatrix} \mathbf{M}_{t,j} & \mathbf{o}_j + \boldsymbol{\delta}_{t,j} \\ \mathbf{0} & 1 \end{bmatrix}. \tag{7.3}$$

The global transformation $\mathbf{G}_{t,j}$ for each joint is computed recursively by traversing the pre-computed topological order:

$$\mathbf{G}_{t,j} = \begin{cases} \mathcal{T}(\mathbf{t}_{\text{root}}) \cdot \mathbf{L}_{t,j}, & \text{if } j \text{ is the root,} \\ \mathbf{G}_{t,p(j)} \cdot \mathbf{L}_{t,j}, & \text{otherwise.} \end{cases}, \tag{7.4}$$

where $\mathcal{T}(\mathbf{t}_{\text{root}})$ applies the global root translation. This formulation ensures structural consistency while remaining fully differentiable with respect to $\mathbf{R}$, $\mathbf{t}_{\text{root}}$, and $\boldsymbol{\delta}$.

Linear Blend Skinning (LBS) is used to deform the surface mesh according to the computed skeletal motion. The mesh consists of V vertices with rest positions $\mathbf{v}_i \in \mathbb{R}^3$. Each vertex is associated with a set of skinning weights $w_{i,j}$ satisfying $\sum_j w_{i,j} = 1$, which determine the influence of each joint j on vertex i. These weights are pre-computed using puppeteer [166] to ensure smooth deformations. Let $\mathbf{B}_j = (\mathbf{G}_j^{\text{rest}})^{-1}$ denote the inverse bind matrix, which transforms points from the global rest pose into the local coordinate frame of joint j. The deformed position $\mathbf{v}_{t,i}$ of vertex i at time t is computed as a weighted sum of the transformed rest positions:

$$\mathbf{v}_{t,i} = \sum_{j=1}^{J} w_{i,j} \left(\mathbf{G}_{t,j} \cdot \mathbf{B}_j \cdot \mathbf{v}_i \right). \tag{7.5}$$

By incorporating the local offsets $\boldsymbol{\delta}$ into $\mathbf{G}_{t,j}$ before skinning, the proposed formulation enables the optimization process to learn subtle, non-rigid surface deformations that blend seamlessly across the mesh.

7.2.5 Diffusion-based motion-aware score distillation guidance

To incorporate high-level semantic motion information, a pre-trained T2V diffusion model [188] is used as a motion prior. SDS operates by rendering a sequence of images and independently adding Gaussian noise to each frame. The diffusion model then predicts the noise per frame. Consequently, the appearance gradients are temporally uncorrelated; the diffusion model may hallucinate disparate textures, colors, or geometric topologies across adjacent frames to satisfy the text prompt. When these independent gradients are backpropagated to the skeletal and local offset parameters, it results in high-frequency surface collapses and structural flickering.

Given a sequence of rendered frames $\mathbf{r} \in \mathbb{R}^{T \times C \times H \times W}$ and a text prompt y, the motion is guided using video motion-aware Score Distillation (MoSDS), inspired by Animus3D [172]. The diffusion model components, including the text encoder and UNet, remain frozen throughout the optimization process. This ensures that the rich semantic knowledge encapsulated in the pre-trained weights is preserved while providing gradients that encourage the rendered animation

to align with the text-conditioned motion distribution. To maintain stability and focus on motion dynamics rather than high-frequency noise or structural hallucination, diffusion timesteps are sampled from the range $\tau \in [0.02, 0.50]$ of the total training steps. The MoSDS process operates in the latent space of the diffusion model. Rendered frames $\mathbf{r}$ are first resized to 256×256, normalized to the range $[-1, 1]$, and encoded into a latent video sequence $\mathbf{z} = \mathrm{E}(\mathbf{r})$. Gaussian noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is added at a randomly sampled timestep τ, producing noisy latents $\mathbf{z}_\tau$. The UNet then predicts the noise residual $\hat{\epsilon}_\theta(\mathbf{z}_\tau; \tau, y)$ conditioned on the text prompt. A Classifier-Free Guidance (CFG) is used with a scale of $w_{\text{cfg}} = 10.0$ to strengthen the fidelity to the text prompt, computing the final noise prediction as $\hat{\epsilon}_{\text{total}} = \hat{\epsilon}_{\text{uncond}} + w_{\text{cfg}}(\hat{\epsilon}_{\text{text}} - \hat{\epsilon}_{\text{uncond}})$.

Unlike SDS, which applies independent gradient updates per frame and frequently destroys temporal coherence, MoSDS decouples the raw noise residual $\Delta\epsilon = \hat{\epsilon}_{\text{total}} - \epsilon$ along the temporal axis. The appearance gradient $\Delta\epsilon_{\text{appear}}$ is extracted by computing the temporal mean of the noise across all N frames, and the motion gradient $\Delta\epsilon_{\text{motion}}$ as the frame-wise deviation from this mean, such that $\Delta\epsilon_{\text{appear}} = \frac{1}{N}\sum_{t=1}^{N} \Delta\epsilon_t$ and $\Delta\epsilon_{\text{motion}} = \Delta\epsilon - \Delta\epsilon_{\text{appear}}$. The final MoSDS gradient with respect to the latent is formulated as a weighted combination of these decoupled signals:

$$\nabla_{\mathbf{z}} \mathcal{L}_{\text{MoSDS}} = \bar{w}(\tau) \left(\lambda_{\text{appear}} \Delta\epsilon_{\text{appear}} + \lambda_{\text{motion}} \Delta\epsilon_{\text{motion}} \right), \tag{7.6}$$

where $\bar{w}(\tau)$ depends on the noise schedule, and $\lambda_{\text{appear}}, \lambda_{\text{motion}}$ are scalar weights. By assigning a high weight to λ_{motion} and suppressing λ_{appear}, the diffusion model serves primarily as a kinematic critic, guiding updates to the skeletal parameters based on motion dynamics while preserving the underlying 3D geometry.

To propagate these gradients back to the skeletal parameters, a proxy loss is defined directly in the latent space [139]. The MoSDS gradient is treated as a constant, and the target latent sequence is constructed as $\mathbf{z}_{target} = \mathbf{z} - \eta \nabla_{\mathbf{z}} \mathcal{L}_{\text{MoSDS}}$. A Mean Squared Error (MSE) loss is computed between the encoded latents $\mathbf{z}$ and $\mathbf{z}_{target}$. Minimizing this proxy loss allows smooth gradients to flow backward through the frozen VAE encoder and the differentiable renderer, directly updating the skeletal kinematics and local offsets.

7.2.6 Physics-inspired motion regularization

While diffusion guidance provides strong semantic priors, it struggles to ensure temporally stable and physically plausible motion. To enforce physics-inspired constraints and exhibit temporal coherence, a comprehensive set of physics-based regularization terms is minimized during animation generation. This objective is crucial for preventing artifacts such as jitter, foot sliding, and unnatural joint rotations.

Temporal smoothness Motion continuity is enforced through hierarchical derivatives of the kinematic parameters to model inertia and damping [74, 203]. The velocity loss ($\mathcal{L}_{vel}$) penalizes the first-order temporal differences of both joint rotations and root translations. Minimizing the squared L_2 norm of the velocity helps to prevent high-frequency jitter and rapid, unnatural pose changes between consecutive frames. Furthermore, an acceleration loss ($\mathcal{L}_{accel}$) is also integrated to minimize the second-order differences of the motion parameters. This term ensures smooth velocity transitions, effectively adding simulated weight to the character and preventing the weightless float often seen in generated animations. Let $\Phi = \{\Phi_t\}_{t=1}^{T}$ denote the sequence of kinematic parameters, including joint rotations and root translations. The loss is defined as the mean squared norm of the velocity $\Delta\Phi_t = \Phi_t - \Phi_{t-1}$. The unified hierarchical smoothness loss is defined as:

$$\mathcal{L}_{\text{smooth}} = \lambda_{\text{vel}} \left(\frac{1}{T-1} \sum_{t=2}^{T} \|\Delta\Phi_t\|_2^2 \right) + \lambda_{\text{accel}} \left(\frac{1}{T-2} \sum_{t=3}^{T} \|\Delta^2\Phi_t\|_2^2 \right),$$

where λ_{vel} and λ_{accel} control the relative strength of velocity and acceleration respectively.

Topological constraints To prevent unnatural poses during optimization, soft constraints based on the character's skeletal structure are applied, inspired by SMPLify [13]. Joint limits ($\mathcal{L}_{\text{rom}}$) are defined by classifying joints into anatomical categories and assigning specific rotational thresholds θ_{limit}. For instance, joints identified as part of the spine are assigned stricter rotational limits (e.g., 0.4 radians) to maintain torso stability. In contrast, hinge-like joints such as knees or elbows are permitted a wider range of motion (e.g., 1.5 radians). A ROM loss function is utilized that penalizes the magnitude of rotations $\mathbf{R}_{t,j}$ only when they exceed these predefined limits, such as:

$$\mathcal{L}_{\text{rom}} = \frac{1}{TJ} \sum_{t=1}^{T} \sum_{j=1}^{J} \max(0, (\|\mathbf{R}_{t,j}\|_2 - \theta_{\text{limit},j}))^2. \tag{7.7}$$

ROM loss function allows free motion within the natural range while heavily penalizing hyperextension.

Additionally, a symmetry loss ($\mathcal{L}_{\text{sym}}$) is applied to characters with bilateral structures to encourage symmetric motion. By identifying paired joints (e.g., left and right legs), the difference in motion magnitude between corresponding joints $\bar{\text{C}}$ is minimized. This regularization prevents asymmetric gaits where one limb moves significantly more than the other, ensuring balanced and coordinated

locomotion [74] such as:

$$\mathcal{L}_{\text{sym}} = \frac{1}{T|\bar{\mathbb{C}}|} \sum_{t=1}^{T} \sum_{(\text{left},\text{right})\in\bar{\mathbb{C}}} (\|\mathbf{R}_{t,\text{left}}\|_2 - \|\mathbf{R}_{t,\text{right}}\|_2)^2. \tag{7.8}$$

Cyclic consistency For animations intended to loop seamlessly, such as walk or run cycles, cyclic consistency ($\mathcal{L}_{\text{cyclic}}$) is achieved via a boundary constraint. This loss minimizes the L_2 distance between the pose (joint rotations and root translations) of the first frame ($t = 1$) and the last frame ($t = T$), inspired by [169]. Beyond positional matching, boundary velocities are also constrained to ensure a seamless transition from the end of the loop back to its beginning:

$$\mathcal{L}_{\text{cyclic}} = \|\Phi_1 - \Phi_T\|_2^2 + \|\Delta\Phi_1 - \Delta\Phi_T\|_2^2. \tag{7.9}$$

The combined physics-based regularization is formulated as:

$$\begin{aligned}\mathcal{L}_{phy} = \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_{\text{rom}}\mathcal{L}_{\text{rom}} \\ + \lambda_{\text{cyclic}}\mathcal{L}_{\text{cyclic}} + \lambda_{\text{sym}}\mathcal{L}_{\text{sym}},\end{aligned}$$

where, λ_{smooth}, λ_{rom}, λ_{cyclic} and λ_{sym} denotes the corresponding weighting coefficients.

Environment interaction Grounding the character in a physical environment is essential for realistic animation. To enable this, a ground penetration ($\mathcal{L}_{\text{ground}}$) term is introduced that strictly penalizes the vertical position $v_{t,i}^{z'}$ of any vertex i that falls below the ground plane ($z' = 0$) [179, 223]. This ensures the character remains above the surface and does not clip through the floor:

$$\mathcal{L}_{\text{ground}} = \frac{1}{TV} \sum_{t=1}^{T} \sum_{i=1}^{V} \max(0, -v_{t,i}^{z'})^2. \tag{7.10}$$

Furthermore, to eliminate the common artifact of foot sliding (where feet glide across the ground while holding weight), a foot-contact ($\mathcal{L}_{\text{contact}}$) loss is used. This term dynamically detects contact phases by monitoring the height of foot joints; when a foot is near the ground threshold, the loss minimizes its horizontal velocity [13]. It effectively plants the foot in place during the stance phase of locomotion, simulating proper friction and weight transfer. Let $\mathbf{p}_{f,t} \in \mathbb{R}^3$ denote the 3D position of a designated foot joint f at frame t, and define its horizontal velocity $\mathbf{vel}_{f,t}^{xy} = (\mathbf{p}_{f,t} - \mathbf{p}_{f,t-1})_{xy}$. A binary contact indicator $h_{f,t} \in \{0, 1\}$ is defined, which is activated when the foot joint is sufficiently close to the ground as $h_{f,t} = \mathbb{I}(p_{f,t}^z < \tau)$, where $p_{f,t}^z$ is the vertical coordinate and τ is a predefined

ALGORITHM 5: Diffusion-guided skeleton-based text-to-3D animation

Input: 2D sketch S and text prompt y
Output: Optimized joint rotations $\{\mathbf{R}_{t,j}\}$, root translations $\{\mathbf{t}_{\text{root},t}\}$, local offsets $\{\boldsymbol{\delta}_{t,j}\}$, and final deformed mesh sequence $\mathbf{V}_{\text{seq}}$

Step 1: 3D Initialization
Estimate 3D mesh $\mathbf{V}^0$ from 2D sketch S, extract skeleton hierarchy $\mathcal{H}$ and skinning weights $\mathbf{w}$.
Step 2: Motion initialization
$\{\mathbf{R}_{t,j}, \mathbf{t}_{\text{root},t}, \boldsymbol{\delta}_{t,j}\} \leftarrow \text{MotionInitializer}(\mathcal{H}, y, T)$; `// Initialize base motion (NURBS)`
$\Psi \leftarrow \text{InitSpringMassSim}(\mathbf{V}^0, \mathcal{D}_{\text{mask}})$; `//` $\mathcal{D}_{\text{mask}} \rightarrow$ `Dynamic region mask`
foreach iteration $u = 1, \ldots, M$ **do**
  Step 3: Forward kinematics with local offsets
  foreach frame $t = 1, \ldots, T$ **do**
    foreach joint $j \in \mathcal{H}$ (topological order) **do**
      $\mathbf{G}_{t,j} \leftarrow \mathbf{G}_{t,\text{p}(j)} \cdot \mathcal{T}(\mathbf{R}_{t,j}, \mathbf{o}_j + \boldsymbol{\delta}_{t,j})$
    end
  end
  Step 4: Mesh skinning & secondary dynamics
  foreach frame $t = 1, \ldots, T$ **do**
    foreach joint $j \in \mathcal{H}$ **do**
      $\mathbf{M}_{t,j} \leftarrow \mathbf{G}_{t,j}\mathbf{B}_j^{-1}$
    end
    $\mathbf{V}_t^{\text{LBS}} \leftarrow \sum_j \mathbf{w}_j \mathbf{M}_{t,j} \mathbf{V}^0$; `// LBS`
  end
  $\mathbf{V}_{\text{seq}} \leftarrow \Psi(\mathbf{V}^{\text{LBS}}, \mathcal{D}_{\text{mask}})$; `// Spring-Mass force for secondary motion effects`
  Step 5: Differentiable rendering & MoSDS guidance
  Render $\mathbf{V}_{\text{seq}}$ to generate 2D frames $\mathbf{r}$.
  Compute MoSDS gradient via decoupled appearance and motion noise residuals:

$$\nabla_{\text{MoSDS}} \leftarrow \lambda_{\text{appear}} \Delta\epsilon_{\text{appear}} + \lambda_{\text{motion}} \Delta\epsilon_{\text{motion}}$$

  Step 6: Physics regularization
  Physics-based losses ($\mathcal{L}_{\text{phy}}$), and physical contact ($\mathcal{L}_{\text{env}}$).
  Step 7: Parameter update
  Minimize $\mathcal{L}_{\text{total}}$ by utilizing iteration-dependent weights $\lambda(u)$:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{MoSDS}}(u)\mathcal{L}_{\text{proxy}} + \sum_k \lambda_k(u)\mathcal{L}_k$$

  Update $\{\mathbf{R}_{t,j}, \mathbf{t}_{\text{root},t}, \boldsymbol{\delta}_{t,j}\}$ via AdamW optimizer.
end
return $\mathbf{V}_{\text{seq}}$

contact threshold. The contact loss then penalizes horizontal motion during

these contact phases:

$$\mathcal{L}_{\text{contact}} = \frac{1}{\sum_{t=1}^{T} h_{f,t} + \epsilon} \sum_{t=1}^{T} h_{f,t} \left\| \mathbf{vel}_{f,t}^{xy} \right\|_2^2, \tag{7.11}$$

where ϵ is a constant for numerical stability.

All environmental constraints are integrated into the generative optimization process:

$$\mathcal{L}_{\text{env}} = \lambda_{\text{ground}} \mathcal{L}_{\text{ground}} + \lambda_{\text{contact}} \mathcal{L}_{\text{contact}}.$$

7.2.7 Secondary motion effects

To model subtle, non-rigid secondary motions, learnable per-joint local translations are introduced. However, without proper constraints, these offsets can distort the underlying skeletal structure or introduce unrealistic deformations. To address this, an offset regularization ($\mathcal{L}_{\text{offset}}$) is applied to penalize the squared L_2 norm of the local offsets and constrain them to represent small, detailed deformations rather than large-scale structural changes. Additionally, temporal velocity regularization is applied to these offsets to ensure that the secondary motion is temporally coherent and does not flicker or jitter independently of the primary skeletal animation. Let $\boldsymbol{\delta}_t \in \mathbb{R}^{J \times 3}$ denote the set of local offsets for all joints at frame t. The offset regularization loss is defined as:

$$\mathcal{L}_{\text{offset}} = \frac{1}{T} \sum_{t=1}^{T} \|\boldsymbol{\delta}_t\|_2^2 + \lambda_\delta \left(\frac{1}{T-1} \sum_{t=2}^{T} \|\boldsymbol{\delta}_t - \boldsymbol{\delta}_{t-1}\|_2^2 \right), \tag{7.12}$$

where λ_δ controls the strength of the temporal smoothness regularization.

Before physics-based simulation, the system automatically partitions the mesh into rigid and deformable regions, determining which vertices are driven solely by skeletal motion (rigid) and which are controlled by physics (soft). This classification is derived from the skinning weights $\mathbf{w}$ and skeletal topology, without requiring manual annotations. Vertices with high weight concentration on core structural bones (e.g., spine, skull) are designated as anchor vertices and remain rigidly controlled via LBS, serving as attachment points for the physics system. In contrast, vertices associated with extremities or non-structural joints (e.g., hair, cloth, tail) are classified as dynamic vertices and are simulated as soft bodies. Additionally, the mesh is partitioned into semantically meaningful regions, enabling region-specific masks (e.g., for hair or cloth), which allow the physics module to apply distinct dynamic behaviors to different parts of the character. Once the vertices are classified, the Spring-Mass Simulator module assigns secondary motion to the dynamic region inspired by [153, 160, 200].

For each dynamic vertex i at a given time step, the system computes the total force $\mathbf{F}_i$ acting on the vertex as the sum of a position spring force, a structural spring force, velocity damping, and gravity. Specifically, the position spring

force $\mathbf{F}_{\text{spring}} = -k_{\text{pos}}(\mathbf{q}_i - \hat{\mathbf{q}}_i)$ binds the simulated position $\mathbf{q}_i$ to its underlying kinematic LBS target $\hat{\mathbf{q}}_i$, ensuring the soft tissue follows the primary skeletal animation. To prevent volume collapse, the structural spring force $\mathbf{F}_{\text{struct}}$ penalizes deviations from the rest-pose edge lengths l_{ij}^0 for adjacent vertices j, formulated as:

$$F_{\text{struct}} = \sum_j k_{\text{struct}} \cdot \Delta l_{ij} \frac{q_j - q_i}{\|q_j - q_i\|},$$

where the stretch Δl_{ij} is clamped to a maximum of 30% of the rest length to maintain stability. Finally, a damping force $F_{damp} = -d \cdot vel_i$, and an gravitational force $F_{grav} = [0, -g \cdot m_i, 0]^\top$, are applied, yielding the total acceleration $a_i = (F_{\text{spring}} + F_{\text{struct}} + F_{\text{damp}} + F_{\text{grav}})/m_i$. The system state is updated over discrete substeps Δt using a semi-implicit Euler integration scheme. Because physics simulations embedded in gradient-based optimization are highly susceptible to numerical instability, strict clamping mechanisms are enforced during integration. First, the updated velocity $vel_i \leftarrow vel_i + a_i \Delta t$ is strictly bounded by a maximum magnitude vel_{max} to prevent drastic acceleration spikes, expressed as:

$$vel_i \leftarrow vel_i \cdot \min(1, \frac{vel_{max}}{\|v_i\|}).$$

Subsequently, the newly integrated position $q_i \leftarrow q_i + vel_i \Delta t$ undergoes displacement clamping, ensuring its Euclidean distance from the LBS target $\hat{q}_i$ never exceeds a predefined threshold d_{max}, which strictly prevents mesh tearing. To ensure a seamless visual transition between the simulated extremities and the rigid kinematic core, the final output position q_i^{out} is computed via linear interpolation using a spatially varying blend weight $w_i \in [0, 1]$. This is formulated as $q_i^{out} = \hat{q}_i + w_i(q_i - \hat{q}_i)$, allowing the generative MoSDS loss to evaluate the visual quality of the secondary dynamics and backpropagate gradients directly through the simulation to optimize the underlying skeletal parameters.

The final optimization objective is:

$$\begin{aligned}\mathcal{L}_{total} = \lambda_{\text{MoSDS}}\mathcal{L}_{\text{MoSDS}} + \lambda_{\text{phy}}\mathcal{L}_{\text{phy}} \\ + \lambda_{\text{env}}\mathcal{L}_{\text{env}} + \lambda_{\text{offset}}\mathcal{L}_{\text{offset}}.\end{aligned}$$

7.3 Experiment and results

7.3.1 Implementation details

Sketch2Motion uses PyTorch3D for differentiable rendering and geometric operations to optimize text-driven motion for a rigged 3D mesh. The animation is parameterized as a sequence of $T = 48$ frames with learnable joint rotations

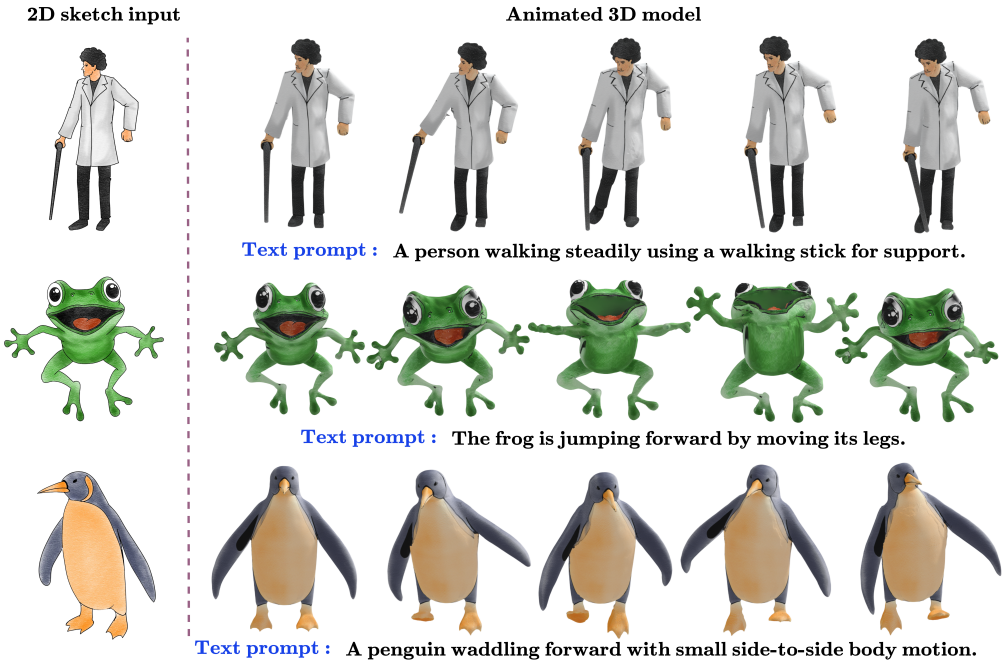


Figure 7.3: Qualitative results of the proposed method and generated 3D animation sequences from input text prompts.

Table 7.1: Analysis of text-to-3D model animation techniques.

Methods	Deformation type	Representation	Model	Dataset	Inference time
MotionDiffuse [227]	Skeleton-driven	Mesh/Pose	Motion Diffsion	HumanML3D & KIT-ML	20 seconds
AKD [110]	Skeleton-driven	3DGS	SDS	Not required	25 hours
Animus3D [172]	Pose-driven	3DGS	MSD	Not required	-
AnimateAnymesh [206]	Pose-driven	Point cloud	AutoEncoder	Objaverse	6-10 seconds
AnimaX [80]	Skeleton-driven	Mesh/Pose	MV Video-Pose Diffusion	Custom dataset	6 minutes
BiMotion [189]	Pose-driven	Point cloud	AutoEncoder	Objaverse	4-8 seconds
Sketch2Motion	Skeleton-driven	Mesh	MoSDS	Not required	5 minutes

$R \in \mathbb{R}^{T\times J\times 3}$ (axis-angle), root translations $t_{\text{root}} \in \mathbb{R}^{T\times 3}$, and per-joint local offsets $\delta \in \mathbb{R}^{T\times J\times 3}$ to model secondary non-rigid motion. Global joint transforms are computed via forward kinematics, and vertices are deformed using linear blend skinning. Text-to-video supervision is provided by the frozen text-to-video diffusion model [188] using MoSDS, and CFG scale set to 10.0; SDS [139] gradients are computed in latent space and backpropagated via an MSE loss on decoded rendered frames. Table 7.2 lists all optimizer and guidance hyperparameters. Optimization uses AdamW [123] with weight decay 10^{-5}, learning rates of 8.0×10^{-3}, 1.0×10^{-2}, and 5.0×10^{-3} for rotations, root translations, and offsets respectively. The timestep curriculum progressively shifts the diffusion sampling window from $[0.25, 0.50]$ at the start of optimization to $[0.02, 0.30]$ later in optimization, allowing early iterations to capture coarse motion while later iterations refine fine-grained dynamics. The MoSDS noise is temporally correlated using a coefficient of $\rho = 0.7$, producing the blended noise term $\boldsymbol{\epsilon}_t$. This temporal cor-

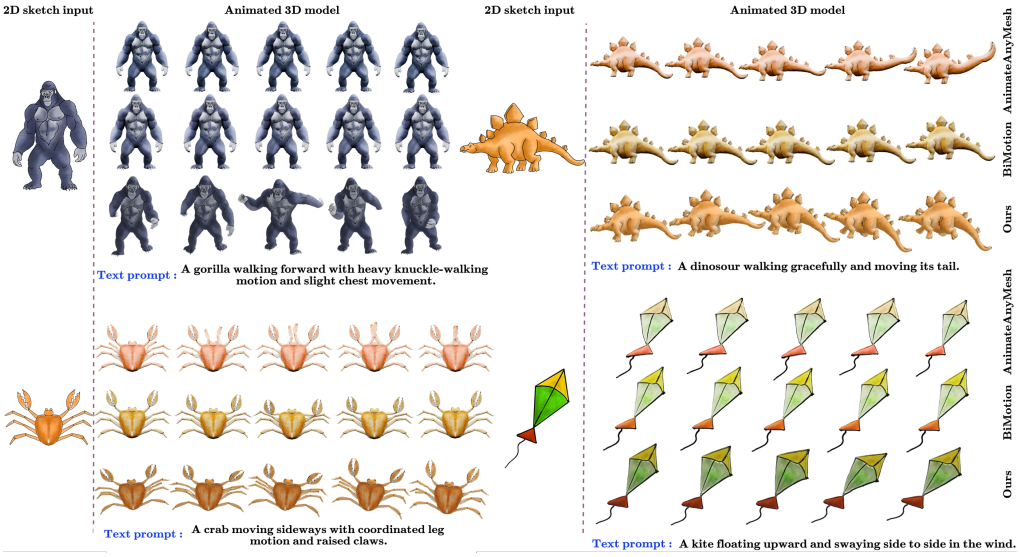


Figure 7.4: Qualitative Comparison with state-of-the-art methods AnimateAnyMesh [206] and BiMotion [189]).

relation provides the denoiser with more realistic motion-consistent inputs than independently sampled frame-wise noise.

7.3.2 Dataset

A dataset comprising 100 hand-drawn sketches is created to assess the performance of the proposed method. The dataset is designed to cover a diverse range of articulation structures and object types, and is divided into three categories: biped (40 sketches), quadruped (40 sketches), and non-living objects (20 sketches). This categorization enables an assessment of the generalization capability of the Sketch2Motion approach across different kinematic topologies and motion complexities, ranging from articulated characters to rigid or semi-rigid objects. For each sketch, Step1X-3D [109] is used to inflate the 2D drawing into a textured 3D mesh representation. The resulting meshes serve as canonical 3D representations for subsequent animation. Furthermore, Puppeteer [166] is used to extract skeletal structures and corresponding skinning weights from the reconstructed meshes automatically. The extracted skeletons, together with the textured meshes, are subsequently used as inputs to the proposed diffusion-guided animation pipeline.

7.3.3 Evaluation matrices

The following metrics are used to quantitatively evaluate the performance of the proposed method.

Table 7.2: Hyperparameter values used in the experiments.

Group	**Parameter**	**Value**
Motion representation		
	Frames T	48
	Rotation parameterisation	Axis-angle
	Trajectory initialisation	NURBS
	NURBS control points N / degree p	8 / 3
	Cyclic motion	enabled
Optimiser		
	Algorithm	AdamW
	Iterations	100
	Learning rate: joint rotations	8×10^{-3}
	Learning rate: root translation	1×10^{-4}
	Learning rate: local offsets	5×10^{-4}
	β_1, β_2	0.9, 0.999
	Weight decay	1×10^{-5}
	Scheduler	cosine annealing, $T_{\max} = 100$, $\eta_{\min} = 1 \times 10^{-4}$
Score distillation		
	Backbone	ModelScope T2V
	Guidance scale	10.0
	Motion weight λ_{motion}	2.0
	Temporal noise correlation ρ	0.7
	Timestep window (start)	$[0.25, 0.50] \times T_{\text{train}}$
	Timestep window (end)	$[0.02, 0.30] \times T_{\text{train}}$
	Absolute timestep bounds	$[0.02, 0.50] \times T_{\text{train}}$
Rendering		
	Optimisation resolution	320×320
	Output resolution	512×512
	Output frame rate	16 fps
Skinning		
	Method	Puppeteer [166]

Table 7.3: Quantitative comparison and user-evaluation with state-of-the-art methods.

	Quantitative Evaluation			User Study		
Methods	**T2VA (↑)**	**MLD (↓)**	**MVFC (↑)**	**T2VA**	**MN**	**SC**
AnimateAnymesh [206]	0.1967	0.6796	0.8810	2.053	2.003	3.033
BiMotion [189]	0.1891	0.0102*	0.8902	1.873	1.950	3.233
Sketch2Motion	0.2120	0.3523	0.8933	3.760	3.267	3.903

* Such a low value is a result of negligible motion in the mesh sequence.

Text-to-video alignment (T2VA) (↑) Text-to-video alignment evaluates the semantic consistency between the input text prompt and the generated animation sequence. This is achieved using X-CLIP [124], text embedding ϕ_{text}, and frame embeddings Θ_{frame} are projected into a shared embedding space, and their similarity is measured using cosine similarity:

$$\hat{S}_{\text{X-CLIP}} = \frac{1}{T}\sum_{t=1}^{T}\frac{\langle \phi_{\text{text}}, \Theta_{\text{frame}} \rangle}{\|\phi_{\text{text}}\|\|\Theta_{\text{frame}}\|},$$

where $\hat{S}_{\text{X-CLIP}}$ denotes the cosine similarity and $\langle \cdot, \cdot \rangle$ denotes the inner product. A higher score indicates better semantic alignment between the prompt and the generated video frames.

Mesh laplacian distortion (MLD) (↓) Mesh Laplacian Distortion measures the geometric stability of a deforming mesh by preserving the local structural relationship between vertices and their neighbors. For a vertex v_i with neighbors $n(i)$, the Laplacian coordinate is defined as $\rho_i = v_i - \frac{1}{|n(i)|}\sum_{\tilde{k}\in n(i)} v_{\tilde{k}}$. The distortion between the deformed mesh and the rest pose is computed as:

$$E_{\text{lap}} = \sum_{i=1}^{V_{\text{total}}} \|\rho_i - \bar{\rho}_i\|_2^2,$$

where lower values indicate better preservation of surface structure during deformation.

Multi-view feature consistency (MVFC) (↑) Multi-view feature consistency evaluates the 3D coherence of the generated animation by comparing features extracted from renderings of the mesh across different viewpoints. Let $\hat{M}_t$ represent the predicted 3D mesh at time frame t, and let $\hat{R}_k$ denote the rendering function configured for the k-th camera viewpoint, out of K total views. Given $\mathcal{K}$ camera views, features are extracted using a pre-trained encoder μ (CLIP), and the consistency is measured as the average pairwise cosine similarity:

$$C_{MV} = \frac{1}{\mathcal{K}(\mathcal{K}-1)}\sum_{k\neq k'}\frac{\langle \mu(\hat{R}_k(\hat{M}_t)), \mu(\hat{R}_{k'}(\hat{M}_t)) \rangle}{\|\mu(\hat{R}_k(\hat{M}_t))\|\|\mu(\hat{R}_{k'}(\hat{M}_t))\|},$$

Higher scores indicate stronger alignment across the rendered views, reflecting a more coherent and geometrically consistent 3D structure.

7.3.4 Comparison

Sketch2Motion generates suitable and text-aligned motion, as illustrated in Figure 7.3. Furthermore, Sketch2Motion is compared with state-of-the-art methods for text-driven 3D animation, including AnimateAnyMesh [206] and BiMotion [189], which are trained on the Objaverse dataset [34]. Although AKD [110] and AnimaX [80] are relevant skeleton-driven approaches, their code implementations are not publicly available, which prevents direct experimental comparison. Therefore, Sketch2Motion is evaluated against the above baselines, which span diverse paradigms in text-driven 3D model animation and provide a comprehensive benchmark for assessing the proposed method. Additionally, a comparative analysis of the baseline methods against Sketch2Motion is provided in Table 7.1. The table compares existing text-to-3D animation methods across deformation types, representations, underlying models, dataset requirements, and inference times. Overall, existing approaches primarily rely on skeleton or pose-driven deformation and use representations such as meshes, 3D Gaussian splats, and point clouds. Their computational costs vary substantially, while some methods require task-specific datasets. In contrast, Sketch2Motion uses skeleton-driven mesh deformation with MoSDS, requires no training dataset, and produces animations in approximately five minutes, providing a balance among dependency, representation, and computational cost.

Qualitative comparison

Sketch2Motion is qualitatively compared with state-of-the-art methods, including AnimateAnyMesh [206], and BiMotion [189], visual results provided in Figure 7.4. AnimateAnyMesh [206] enables motion generation on arbitrary meshes by leveraging diffusion priors and deformation fields. While it significantly improves generalization across diverse mesh topologies, it primarily operates in a deformation space and does not enforce explicit skeletal or kinematic constraints. Consequently, it produces temporally smooth but structurally inconsistent motions shown in Figure 7.4, particularly for articulated objects where joint coherence and bone-length preservation are critical. Although this method is effective for subtle movements, its approach does not generalize well to large articulated movements or complex full-body animation. Meanwhile, BiMotion [189] introduces a bidirectional motion representation that jointly models motion dynamics and geometry for text-driven animation. Although it improves motion diversity and temporal coherence compared to prior diffusion-based approaches, it still relies on implicit motion representations and lacks strong structural priors for articulated bodies. This leads to ambiguity in joint behavior and reduced control-

lability of the complex actions shown in Figure 7.4. However, these methods are trained on specific datasets and thus struggle to generalize to novel articulated meshes and diverse text prompts. In contrast to these baselines, Sketch2Motion operates directly on explicit skeletal parameters and 3D meshes. By extending diffusion guidance to full articulated skeletons and integrating forward kinematics with physics-inspired regularization, the proposed framework ensures physically plausible articulation. This explicit parameterization provides precise control over complex motion semantics, including secondary effects, while maintaining geometric consistency and stable deformation across all frames and viewpoints.

Quantitative comparison

Table 7.3 presents a quantitative comparison of the proposed method with AnimateAnyMesh [206] and BiMotion [189] across three complementary metrics: text-to-video alignment, mesh Laplacian distortion, and multi-view feature consistency. These metrics collectively evaluate semantic alignment, geometric stability, temporal smoothness, and 3D coherence. Lower values indicate better performance for mesh distortion, while higher values are preferred for text alignment and multi-view consistency. AnimateAnyMesh [206] improves over prior diffusion-based approaches by leveraging deformation priors, resulting in moderate improvements in geometric stability and smoother motion. However, due to the absence of explicit skeletal constraints, it exhibits higher mesh distortion and lower multi-view feature consistency. BiMotion [189] struggles to generate dynamic animations, often producing small movements. Due to the limited motion, Laplacian distortion is very low; nevertheless, its implicit motion representation limits global 3D consistency, resulting in comparatively lower text-to-video alignment. In contrast, the Sketch2Motion method consistently outperforms both baselines across all metrics. Sketch2Motion achieves the lowest mesh Laplacian distortion, indicating stable, smooth surface deformation. Additionally, the Sketch2Motion approach attains higher text-to-video alignment, reflecting stronger semantic consistency between the generated animation and the input prompt. Finally, the significant improvement in multi-view feature consistency confirms that the proposed framework produces coherent 3D motion representations that remain stable across varying viewpoints.

User evaluation

To further evaluate the perceptual quality of the generated animations, a user study involving 20 participants is conducted. Each participant evaluates 15 generated animation samples produced by Sketch2Motion, AnimateAnyMesh [206], and BiMotion [189]. The evaluation focused on three criteria: text-to-motion alignment (T2MA), motion naturalness (MN), and shape consistency (SC). T2MA measures how well the generated motion corresponds to the text description of

the intended action, while MN evaluates the perceived smoothness, continuity, and naturalness of the resulting motion. SC assesses whether the generated animation maintains a coherent and plausible structure throughout the motion sequence. Participants rated each criterion on a five-point scale, with 5 indicating the highest quality and 1 the lowest. The mean ratings across all participants and samples are reported in Table 7.3. Sketch2Motion achieves higher scores than AnimateAnyMesh [206] and BiMotion [189] across T2MA, MN, and SC, indicating improved semantic alignment, motion naturalness, and structural plausibility. These results further demonstrate the effectiveness of the proposed method in motion generation.

Overall, quantitative and qualitative results demonstrate that Sketch2Motion achieves better semantic motion alignment, lower mesh distortion, and multi-view feature consistency compared to all baselines. These improvements highlight the advantages of combining diffusion-based semantic guidance with explicit skeletal modeling and physics-aware regularization.

7.3.5 Ablation study

To validate the effectiveness of the proposed technique, an ablation study is conducted by systematically removing key components of the pipeline. Figure 7.5 presents the qualitative results, while Table 7.4 reports the quantitative analysis under different settings. The impact of motion initialization, physics-based regularization, and secondary motion on the generated animations is evaluated from both qualitative and quantitative perspectives.

Without motion initialization

To validate the necessity of the proposed procedural motion initialization module (Section 7.2.3), a baseline (*w/o Motion Init*) is evaluated where all skeletal parameters, joint rotations, root translations, and local offsets are zero-initialized to a static rest pose across all frames. In this setting, Video MoSDS synthesizes the full kinematic trajectory from scratch, which often leads to poor local minima due to the lack of inherent 3D structural awareness in text-to-video diffusion models. Consequently, the optimization exhibits artifacts such as floating locomotion, excessive reliance on root translation, foot-skating, and physically implausible joint configurations. Additionally, direct optimization in the high-dimensional space introduces significant temporal jitter, degrading motion smoothness. The limitations of the zero-initialized baseline are clearly visible in Figure 7.5 and Table 7.4. In contrast, the full pipeline utilizes a low-dimensional NURBS projection of a procedural, biomechanically informed base motion, initializing the parameters within a physically plausible kinematic manifold. As a result, enabling video MoSDS to act as a semantic refiner, adaptively modify-

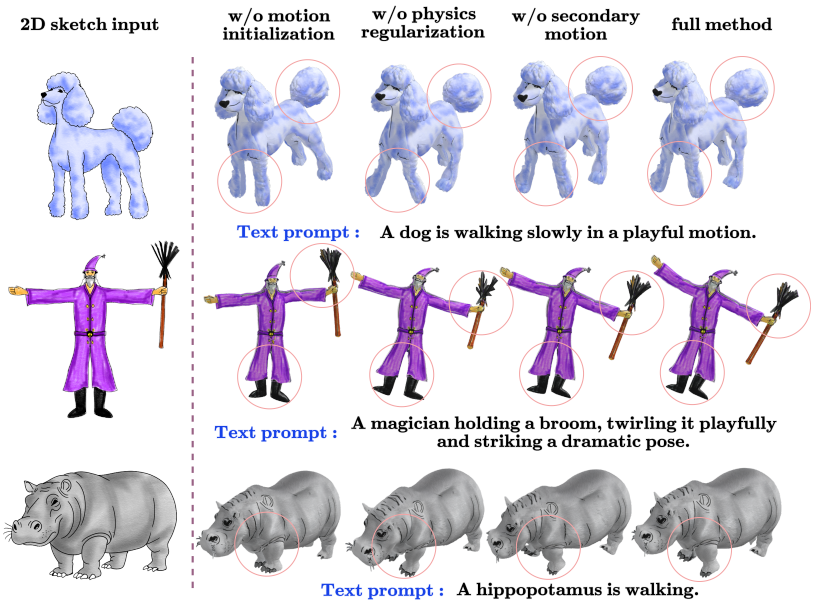


Figure 7.5: Visual results of ablation study on different settings such as w/o motion initialization, w/o physics regularization, w/o secondary motion, and the full method.

ing the base gait to align with the text prompt while preserving critical phase dynamics. This leads to improved consistency of joint motion and significantly faster convergence, demonstrating that a strong morphological prior effectively regularizes the highly non-convex MoSDS optimization landscape.

Without physics-based regularization

In this setting, the skeletal representation is used without the physics-based regularization terms from the objective function. The weights for smoothness, topological constraints, ground contact, and symmetry are set to zero, leaving the MoSDS loss as the sole optimization objective. Removing physics-based priors results in motion that is visually plausible in static frames but physically incoherent over time. Several distinct failure modes are observed: Without the temporal smoothness terms ($\mathcal{L}_{\text{vel}}, \mathcal{L}_{\text{acc}}$), the animation suffers from high-frequency flickering and erratic pose changes, shown in Figure 7.5. Without foot contact loss ($\mathcal{L}_{\text{contact}}$) and ground penetration penalty ($\mathcal{L}_{\text{ground}}$), the character exhibits motion artifacts. Feet slide across the base during the stance phase instead of remaining planted, and limbs frequently clip through the ground plane, breaking the illusion of physical interaction. Without joint limit constraints ($\mathcal{L}_{\text{rom}}$), the optimization exploits the full range of rotation to satisfy the visual prompt, resulting in broken joints (e.g., backward bending of the knees) and biomechanically impossible poses. Further, removing the symmetry regulariza-

tion ($\mathcal{L}_{\text{sym}}$) for bilateral characters leads to uncoordinated locomotion as shown in Figure 7.5, with significant differences in limb motion magnitudes, resulting in a limping or lopsided appearance. Table 7.4 provides a quantitative analysis of this configuration in terms of text-to-video alignment, mesh Laplacian distortion, and multi-view feature consistency. These results confirm that while SDS provides necessary semantic guidance, physics-based regularization is essential for grounding the generated motion in biomechanical reality.

Table 7.4: Quantitative ablation of the proposed approach, evaluating the impact w/o motion initialization, w/o physics-based regularization, and w/o secondary motion, compare to the full method.

Methods \Metrics	**T2VA** (↑)	**MLD** (↓)	**MVFC** (↑)
w\o motion initialization	0.2059	0.4241	0.8930
w\o physics-based regularization	0.2071	0.3745	0.8920
w\o secondary motion	0.2089	0.3195	0.8927
full method (Sketch2Motion)	0.2120	0.3523	0.8933

Without secondary motion

In this configuration, the secondary dynamics module is removed, leaving the mesh to rely strictly on the primary skeletal deformation via LBS. Vertices classified as dynamic soft bodies (e.g., hair or cloth) are treated as rigid extensions of the bone hierarchy. While the primary joints successfully execute the text-guided action, the resulting animation appears stiff and artificial, resembling a rigid figure rather than a compliant, organic entity. A lack of follow-through and overlapping action, two fundamental principles of natural movement, is observed, as illustrated in Figure 7.5. For instance, a dog's tail and the magician's cap fail to exhibit the natural phase-delayed lag and spring-mass recoil dictated by momentum and gravity. By skipping these secondary spring-mass dynamics, the system loses the subtle, passive physical reactions that are essential for bridging the gap between mathematically correct kinematics and perceptually realistic 3D animation. Table 7.4 reports the quantitative results for this setting, showing strong text alignment with a minor increase in distortion introduced by secondary motion effects.

7.3.6 Limitation and future work

Despite its effectiveness, the proposed diffusion-guided, physics-aware framework has certain limitations that suggest promising directions for future research. First, Sketch2Motion relies on pretrained text-to-video diffusion models as semantic motion priors. Figure 7.6 shows the results generated by the T2V model, along with the motion aligned to the 3D model based on T2V embeddings.

While this enables motion synthesis, the quality and diversity of generated motion are inherently bounded by the capabilities and biases of the underlying text-to-video prior model. Complex or highly specialized motions that are underrepresented in the training data may not be accurately captured, leading to suboptimal or ambiguous motion guidance. Additionally, the diffusion-guided optimization process is computationally expensive, requiring repeated rendering and gradient-based updates across long animation sequences, limiting scalability to high-resolution meshes or real-time applications. Second, although physics-inspired constraints are incorporated to stabilize the optimization, the proposed formulation does not explicitly model the full physical dynamics, such as forces, mass, or contact interactions. As a result, certain motion artifacts such as subtle ground sliding or inaccurate contact timing may still occur in challenging scenarios. Furthermore, the skeleton extraction and rigging process depends on the quality of the reconstructed 3D mesh; errors in sketch-to-3D reconstruction or skeleton inference can propagate into the animation stage. Finally, the current framework focuses on single-character animation and does not explicitly handle interactions between multiple animated entities or dynamic environments.

Future work can explore several directions to address these limitations. Recent closed-source text-to-video (T2V) diffusion models can further improve motion-generation capabilities. Furthermore, incorporating lightweight physics simulators or learned dynamics models could improve contact handling and physical realism while maintaining differentiability. Motion efficiency may be enhanced by distilling diffusion guidance into motion priors or adopting latent-space optimization strategies to reduce computational overhead. Extending the framework to multi-character interactions and environment-aware motion generation represents another promising direction. Finally, integrating richer conditioning signals, such as user-provided keyframes or sketch-based motion hints, could further enhance controllability and expand the applicability of the proposed approach.

7.4 Conclusion

A diffusion-guided, physics-aware framework is introduced for text-driven skeleton-based 3D animation. Sketch2Motion works directly on explicit skeletal motion parameters, including joint rotations and root translations, and integrates forward kinematics, linear blend skinning, and differentiable rendering into a unified optimization framework. By integrating motion-aware score distillation guidance from a text-to-video diffusion model, the proposed approach injects high-level semantic motion cues while preserving the interpretability and controllability of skeleton-based representations. To ensure stable optimization and physically plausible animation, a set of physics-inspired constraints is incorporated that enforce temporal smoothness, joint motion limits, structural consistency, and

contact-aware motion. The SpringMass simulator is integrated to model secondary motion, enhancing the realism of the generated animations. This combination enables the proposed framework to generate realistic, temporally coherent motion while avoiding common artifacts, such as jitter, distortion, and kinematic violations. Importantly, this approach does not require paired motion data or task-specific retraining and generalizes across diverse articulated structures, including bipeds, quadrupeds, and non-living objects. Together, these contributions establish a flexible and principled framework for controllable, semantically meaningful motion transfer driven by diffusion priors.

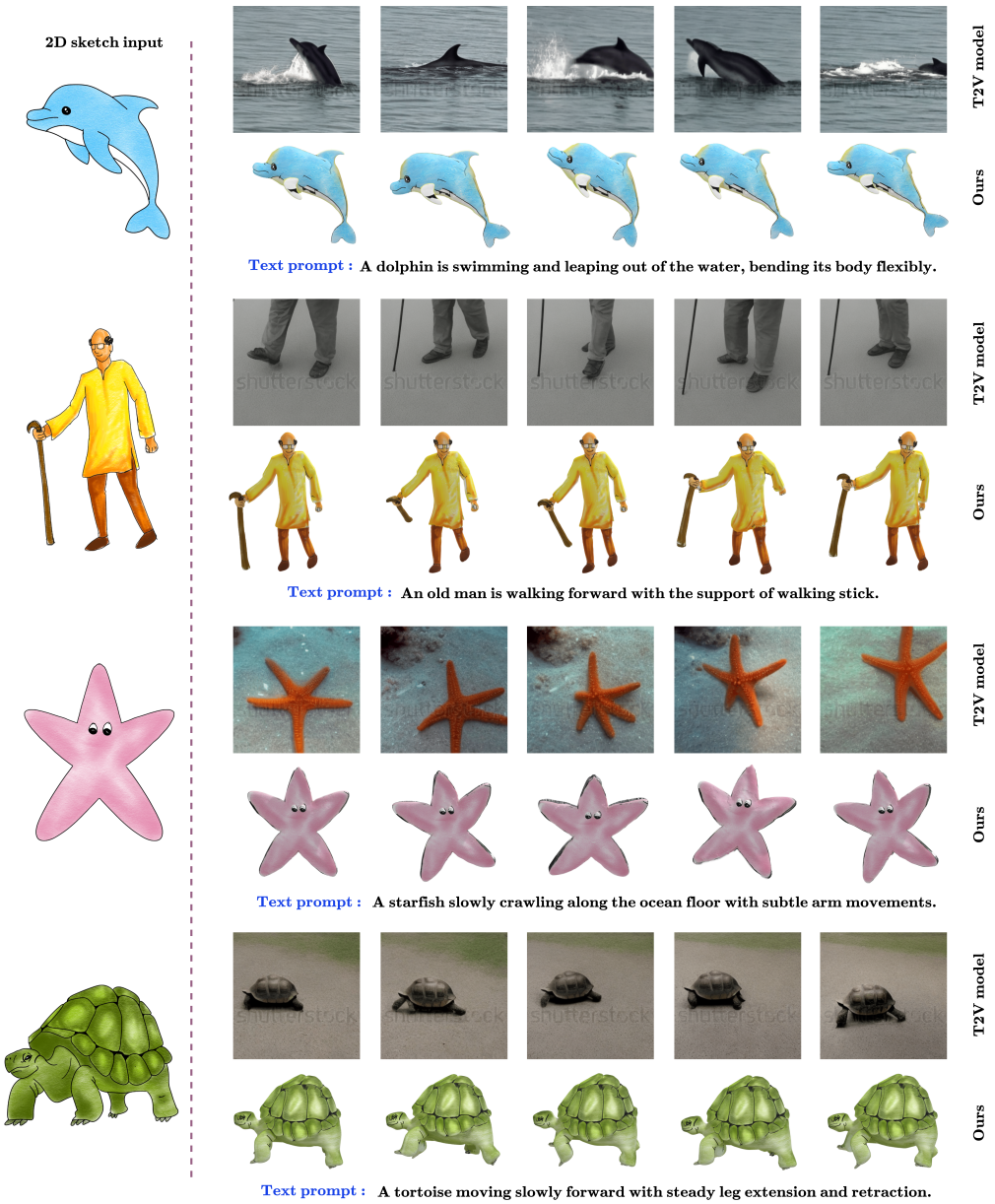


Figure 7.6: Visual comparison of T2V diffusion model results generated by ModelScope [188] and the proposed method.

Chapter 8

Discussion

This chapter provides an overview of the primary contributions of the thesis, limitations, and discusses potential directions for future research. This thesis advances image and sketch animation by exploring diverse motion representations and motion modeling strategies to generate animation with minimal user input. The proposed frameworks demonstrate effective motion transfer from videos, text, and skeletal cues while preserving the visual structure and stylistic consistency of visual input. These techniques have direct applications in content creation, digital art, and animation production, where rapid and controllable animation is essential. Overall, these works provide practical solutions for next-generation animation systems in creative and immersive media applications.

8.1 Contribution of the thesis

This thesis contributes a single, transferable design principle: for sparse visual input, controllable animation is achieved by strengthening the motion representation rather than enlarging the generative model. The main contributions of this thesis are as follows:

1. **Representation-centric design principle.** Sketches and photos are animated reliably by describing the motion structure more explicitly rather than enlarging the generative model.

2. **Progressively explicit motion representations.** The thesis introduces a series of frameworks whose intermediate representation becomes increasingly explicit. From pixel warping to a 2D skeleton and mesh, then to stroke curves, then to a rigged 3D skeleton.

3. **Structural and physical constraints.** Each representation is regularized with constraints suited to it, including keypoint regularization, depth-ordered occlusion reasoning, length-area and as-rigid-as-possible preservation, and physics-inspired constraints.

4. **Bounded role of the generative prior.** The generative model provides only the motion semantics, while identity, topology, proportions, and physical plausibility are guaranteed by the explicit representation it is optimized against.

5. **General and accessible animation.** The frameworks animate a single photograph or hand-drawn sketch from either a reference video or a text prompt. They generalize across bipeds, quadrupeds, and inanimate objects without paired motion data, category-specific retraining, or animation expertise from the user.

Both image and sketch animation systems offer automation, higher visual fidelity, and temporal coherence than traditional animation techniques. Image animation benefits from appearance-aware generative frameworks that preserve identity and reduce motion-induced artifacts, while sketch animation has shifted towards minimally supervised, generalized, and constraint-guided generation that maintains both motion dynamics and structural information.

8.2 Limitations

The limitations outlined are intended to clearly define the scope of the work.

Chapter 4 does not generalize across domains. LiveImage performs within a single visual domain. Its failure on sketches is reported as a finding rather than a defect; however, it remains a limitation on the applicability of the method.

Chapter 5 depends on the quality of boundary extraction. Correspondence is established from the segmented outlines of the sketch. Sketches with open contours, heavy interior detail, or ambiguous figure-ground separation can degrade the shape matching on which the subsequent stages depend. When the driving skeleton contains branches with no geometric counterpart in the sketch, only the subset with valid correspondences contributes useful motion.

Chapter 6 is an extension rather than a standalone system. It inherits its architecture and objective from LiveSketch and is constrained by the quality of the text-to-video prior. The geometric objectives are applied to the output of the T2V diffusion model with a score-distillation stage without gradient optimization, so they correct that stage's output rather than shape its trajectory.

Chapter 7 inherits the limitations of its front end. The sketch-to-3D generation model bounds the quality of the reconstructed mesh, while the automatic

rigging stage bounds the quality of the rig; neither component is a contribution of this thesis. The quality of motion semantics is similarly bounded by the video prior. Motion quality is assessed primarily through distributional and alignment metrics computed on rendered frames. Quantitative evaluation metrics used in Chapter 7 are not specifically designed to measure physics-based constraints.

8.3 Future direction

Despite advances in sketch animation using optimization and learning-based techniques, creating high-quality sketch animation remains a challenging problem. Handling topology changes, self-occlusions, and partial drawings is challenging, and maintaining temporal coherence while preserving the artist's objective and sketch quality (thickness and jitter) is nontrivial, especially under non-rigid deformations and articulated motion. The following are prominent challenges and possible directions of future work in sketch animation.

Functional relationship in multi-sketch animation In the future, several important research directions can further advance sketch and image animation. A key challenge lies in extending current methods to multi-object, multi-entity scenarios, where multiple characters or entities interact with one another in a coherent, physically plausible manner. This requires modeling not only individual motion but also inter-object relationships such as contact, collision, and coordinated behavior. Capturing such interactions is particularly important for complex scenes involving crowds, articulated objects, or human–object interactions, where maintaining temporal consistency and structural integrity across entities remains an open problem.

Real-time and interactive generation Another promising direction is the development of real-time and interactive animation systems. Enabling low-latency inference would allow users to receive fast visual feedback, making animation tools more intuitive and accessible. This can be further enhanced through multimodal control mechanisms, allowing users to guide animations with a combination of inputs, such as text, sketches, audio, or gestures. For example, audio signals could influence rhythm and timing, while richer semantic prompts could provide high-level control over motion style, intent, or emotion. Integrating physics-aware reasoning into such systems would also improve realism by ensuring that generated motions adhere to constraints such as gravity, balance, and contact dynamics.

Creation of large-scale datasets In addition, there is a strong need for larger, more diverse datasets that capture the variability of sketches, images,

and motion patterns. Current datasets are often limited in scale, style diversity, and motion complexity, which restricts the generalization capability of learned models. Developing high-quality paired datasets linking sketches, images, text descriptions, and motion sequences would enable more robust training and evaluation. Such datasets should contain a wide range of artistic styles, levels of abstraction, and structural variations, as well as diverse motion types, including subtle deformations and complex articulated movements.

Motion coherence Finally, improving generalization and robustness remains a critical objective. Future generative AI-based animation models should be able to handle unseen object categories, diverse sketch styles, and extreme motion variations without degrading quality. This may involve advances in representation learning, cross-domain adaptation, and the integration of explicit structural priors.

In summary, this thesis demonstrates progress in sketch animation by systematically addressing key challenges, including motion transfer, structural preservation, occlusion handling, and automation. Through a sequence of frameworks spanning video-driven, text-driven, and 3D animation settings, it demonstrates how combining robust motion representations with diffusion-based generative models enables more accurate, coherent, and visually plausible animations. Overall, this framework bridges the gap between traditional sketch-based techniques and modern AI-driven approaches, enabling more flexible, scalable, and accessible animation pipelines.

Chapter 9

Conclusion

This thesis aims to transfer or synthesize motion for visual content with different input modalities, while maintaining temporal consistency, structural fidelity, and physical plausibility. This chapter summarizes the thesis, discusses the objectives achieved, and concludes with a remark.

9.1 Summary of the thesis

Animation remains challenging, particularly when compared with the ease of creating the visual content to be animated. Creating a drawing, photograph, or rough sketch is achievable to most users, but bringing such content to life typically requires knowledge of timing, keyframing, rigging, and inbetweening, as well as specialized tools. This thesis addresses this gap by exploring ways to make animation more accessible through motion transfer and motion synthesis, using videos and text prompts as intuitive sources of motion rather than requiring users to author animations frame by frame.

This research demonstrates that the choice of visual inputs strongly influences how motion can be transferred and synthesized. Appearance-based methods are effective when rich visual information is available, but become less reliable for sparse representations such as sketches. This motivates the use of explicit geometric and skeletal representations, which provide stronger structural constraints and enable motion to be transferred while preserving the topology and visual characteristics of sketches. The research further shows that replacing reference videos with text prompts can provide greater user control and eliminate the need for suitable motion examples. However, 2D representations remain limited when motion involves significant depth changes, viewpoint variation, and self-occlusion.

Extending the representation to 3D addresses the limitations associated with 2D sketch animation by providing explicit depth and articulated structure. Combining generative motion priors with geometric and physics-based constraints further improves the structural and physical plausibility of generated anima-

tions. Overall, this thesis establishes a progression from appearance-based motion transfer to geometry-aware animation, from reference-driven to semantically controlled motion, and from 2D to physically grounded 3D animation. These results demonstrate how increasingly explicit representations and constraints can reduce the technical effort required to create expressive and controllable animations.

9.2 Objectives revisited

The objectives (O1, O2, and O3) are revisited to assess what the thesis set out to achieve, what was accomplished, and what gaps remain. This also highlights how each objective evolved from limitations identified in the preceding work, progressing from appearance-driven image animation to structure-aware sketch animation and from 2D sketch to 3D animation.

O1: Video-driven image animation. Develop a robust framework for transferring motion from a driving video to a static image while handling large pose variations and preserving structural and appearance consistency.

O2: Controllable sketch animation. Design structure-preserving frameworks that animate hand-drawn sketches using either an exemplar video or a text prompt, while maintaining sketch topology, artistic style, temporal coherence, and robustness to self-occlusion.

O3: Sketch-to-3D animation. Generate rigged and physically plausible 3D animations from 2D sketches while addressing the depth, viewpoint, and occlusion limitations of 2D animation.

All three objectives were achieved through the proposed systems, which were evaluated against relevant baselines. The dependencies among the objectives were also maintained, with each objective motivated by a limitation identified in the preceding work rather than defined independently. Two limitations should be noted. First, the progression from image to sketch animation reduced the strength of the available evidence for evaluation. O1 could be evaluated using ground-truth frames and reconstruction error. In contrast, ground-truth animations are generally unavailable for individual sketches, so O2 and O3 rely on reference-free and cross-modal metrics. The agreement between these metrics and human evaluation has not been established for all frameworks. Second, each system reduces the need for animation expertise but introduces a different practical requirement. O1 requires a suitable reference video, O2 requires a clean closed contour and accurate tracking, and O3 requires several minutes of computation for each result. These requirements represent useful trade-offs rather than the complete removal of the obstacles to animation.

9.3 Concluding remarks

In summary, the thesis highlights an important point: addressing sparse inputs does not necessarily require larger generative models but rather more explicit representations. When the input contains rich appearance information, a generative model can infer missing structure from the available visual cues. However, when such information is absent, a larger generative model cannot reliably recover the missing structure. It must instead be represented explicitly, using skeletal representations and depth ordering, rigidity and length constraints for parametric strokes, or joint limits, contact, and balance constraints in 3D. At each stage, the generative prior contributes semantic plausibility, while the explicit representation provides what the prior cannot ensure: geometric and physical consistency.

The main goal of the thesis is to make animation more accessible to novice users and enable its automation with a simple form of visual input. Video-driven image animation and sketch animation are not simply variants of the same problem. Video-driven animation benefits from dense appearance cues, making motion estimation relatively straightforward, while the main challenges lie in extracting and transferring complex motion and synthesizing disoccluded appearance. In video-driven sketch animation, there is less appearance information to synthesize, but establishing reliable correspondence is more difficult. Consequently, methods do not transfer motion directly between these two settings. Generative priors will continue to improve, and more capable models may eventually replace several components of the systems presented in this thesis. However, the need to make explicit what a generative prior cannot guarantee is likely to persist, as this requirement depends not only on the quality of the prior but also on the structure and constraints of the problem itself.

Bibliography

[1] A. Agarwala, A. Hertzmann, D. H. Salesin, and S. M. Seitz, Keyframe-based tracking for rotoscoping and animation, ACM Transactions on Graphics (ToG), 23 (2004), pp. 584–591.

[2] S. Agrawal, S. Shen, and M. Van de Panne, Diverse motion variations for physics-based character animation, in Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2013, pp. 37–44.

[3] S. Bahmani, X. Liu, W. Yifan, I. Skorokhodov, V. Rong, Z. Liu, X. Liu, J. J. Park, S. Tulyakov, G. Wetzstein, et al., Tc4d: Trajectory-conditioned text-to-4d generation, in European Conference on Computer Vision, Springer, 2024, pp. 53–72.

[4] H. Bandyopadhyay and Y.-Z. Song, Flipsketch: Flipping static drawings to text-guided sketch animations, in Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 28394–28404.

[5] M. Banks and E. Cohen, Real time spline curves from interactively sketched data, ACM SIGGRAPH Computer Graphics, 24 (1990), pp. 99–107.

[6] I. Baran and J. Popović, Automatic rigging and animation of 3d characters, ACM Transactions on graphics (TOG), 26 (2007), pp. 72–es.

[7] C. Barnes, D. E. Jacobs, J. Sanders, D. B. Goldman, S. Rusinkiewicz, A. Finkelstein, and M. Agrawala, Video puppetry: a performative interface for cutout animation, in ACM SIGGRAPH Asia 2008 papers, 2008, pp. 1–9.

[8] N. Barroso, A. Fondevilla, and D. Vanderhaeghe, Automatic inbetweening for stroke-based painterly animation, in Computer Graphics Forum, vol. 44, Wiley Online Library, 2025, p. e15201.

[9] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, Parametric correspondence and chamfer matching: Two new techniques for image matching, in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1977.

[10] N. Ben-Zvi, J. Bento, M. Mahler, J. Hodgins, and A. Shamir, Line-drawing video stylization, in Computer Graphics Forum, vol. 35, Wiley Online Library, 2016, pp. 18–32.

[11] M. Bessmeltsev, N. Vining, and A. Sheffer, Gesture3d: posing 3d characters via gesture drawings, ACM Transactions on Graphics (TOG), 35 (2016), pp. 1–13.

[12] A. Blattmann, T. Dockhorn, S. Kulal, D. Mendelevitch, M. Kilian, D. Lorenz, Y. Levi, Z. English, V. Voleti, A. Letts, et al., Stable video diffusion: Scaling latent video diffusion models to large datasets, arXiv preprint arXiv:2311.15127, (2023).

[13] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, Keep it smpl: Automatic estimation of 3d human pose and shape from a single image, in European conference on computer vision, Springer, 2016, pp. 561–578.

[14] C. Bregler, L. Loeb, E. Chuang, and H. Deshpande, Turning to the masters: Motion capturing cartoons, ACM Transactions on Graphics (TOG), 21 (2002), pp. 399–407.

[15] K. Brodt and M. Bessmeltsev, Sketch2pose: Estimating a 3d character pose from a bitmap sketch, ACM Transactions on Graphics (TOG), 41 (2022), pp. 1–15.

[16] ——, Skeleton-driven inbetweening of bitmap character drawings, ACM Transactions on Graphics (TOG), 43 (2024), pp. 1–19.

[17] C. Cao, Q. Hou, and K. Zhou, Displaced dynamic expression regression for real-time facial tracking and animation, ACM Transactions on graphics (TOG), 33 (2014), pp. 1–10.

[18] L. Carvalho, R. Marroquim, and E. V. Brazil, Dilight: Digital light table–inbetweening for 2d animations using guidelines, Computers & Graphics, 65 (2017), pp. 31–44.

[19] E. Catmull, The problems of computer-assisted animation, ACM Siggraph Computer Graphics, 12 (1978), pp. 348–353.

[20] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros, Everybody dance now, in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 5933–5942.

[21] S. Cheema and J. LaViola, Physicsbook: a sketch-based interface for animating physics diagrams, in Proceedings of the 2012 ACM international conference on Intelligent User Interfaces, 2012, pp. 51–60.

[22] H. Chen, M. Xia, Y. He, Y. Zhang, X. Cun, S. Yang, J. Xing, Y. Liu, Q. Chen, X. Wang, et al., Videocrafter1: Open diffusion models for high-quality video generation, arXiv preprint arXiv:2310.19512, (2023).

[23] J. Chen, X. Zhu, M. Even, J. Basset, P. Bénard, and P. Barla, Efficient interpolation of rough line drawings, in Computer Graphics Forum, vol. 42, Wiley Online Library, 2023, p. e14946.

[24] R. Chen, Y. Chen, N. Jiao, and K. Jia, Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 22246–22256.

[25] X. Chen, Z. Liu, M. Chen, Y. Feng, Y. Liu, Y. Shen, and H. Zhao, Livephoto: Real image animation with text-guided motion control, arXiv preprint arXiv:2312.02928, (2023).

[26] B. Choi, R. B. i Ribera, J. P. Lewis, Y. Seol, S. Hong, H. Eom, S. Jung, and J. Noh, Sketchimo: sketch-based motion editing for articulated characters, ACM Transactions on Graphics (ToG), 35 (2016), pp. 1–12.

[27] Y.-Y. Chuang, D. B. Goldman, K. C. Zheng, B. Curless, D. H. Salesin, and R. Szeliski, Animating pictures with stochastic motion textures, in ACM SIGGRAPH 2005 Papers, 2005, pp. 853–860.

[28] L. Ciccone, M. Guay, M. Nitti, and R. W. Sumner, Authoring motion cycles, in Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2017, pp. 1–9.

[29] F. Corda, J.-M. Thiery, M. Livesu, E. Puppo, T. Boubekeur, and R. Scateni, Real-time deformation with coupled cages and skeletons, in Computer Graphics Forum, vol. 39, Wiley Online Library, 2020, pp. 19–32.

[30] M. G. Cox, The numerical evaluation of b-splines, IMA Journal of Applied Mathematics, (1972).

[31] O. Dadfar and N. Pollard, 3a2a: A character animation pipeline for 3d-assisted 2d-animation, in International Conference on Image and Graphics, Springer, 2021, pp. 557–568.

[32] B. Dalstein, R. Ronfard, and M. Van De Panne, Vector graphics animation with time-varying topology, ACM Transactions on Graphics (TOG), 34 (2015), pp. 1–12.

[33] J. Davis, M. Agrawala, E. Chuang, Z. Popović, and D. Salesin, A sketching interface for articulated figure animation, in Acm siggraph 2006 courses, 2006, pp. 15–es.

[34] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and

A. Farhadi, Objaverse: A universe of annotated 3d objects, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 13142–13153.

[35] B. Delaunay, Sur la sphère vide, Bulletin de l'Académie des Sciences de l'URSS: Classe des Sciences Mathématiques et Naturelles, 6 (1934), pp. 793–800.

[36] H. Deng, X. Dai, J. Hu, and Y. Qi, Animatesketches: Animate sketches with instance-aware mask, in ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2025, pp. 1–5.

[37] M. C. Doukas, S. Zafeiriou, and V. Sharmanska, Headgan: One-shot neural head synthesis and editing, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 14398–14407.

[38] M. Dvorožnák, W. Li, V. G. Kim, and D. Sỳkora, Toonsynth: example-based synthesis of hand-colored cartoon animations, ACM Transactions on Graphics (TOG), 37 (2018), pp. 1–11.

[39] M. Dvorožňák, D. Sỳkora, C. Curtis, B. Curless, O. Sorkine-Hornung, and D. Salesin, Monster mash: a single-view approach to casual 3d modeling and animation, ACM Transactions on Graphics (ToG), 39 (2020), pp. 1–12.

[40] M. Eitz, J. Hays, and M. Alexa, How do humans sketch objects?, ACM Transactions on graphics (TOG), 31 (2012), pp. 1–10.

[41] S. Eroglu, S. Gebhardt, P. Schmitz, D. Rausch, and T. W. Kuhlen, Fluid sketching—immersive sketching based on fluid flow, in 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), IEEE, 2018, pp. 475–482.

[42] M. Even, P. Bénard, and P. Barla, Non-linear rough 2d animation using transient embeddings, in Computer Graphics Forum, vol. 42, Wiley Online Library, 2023, pp. 411–425.

[43] ——, Inbetweening with occlusions for non-linear rough 2d animation, Computers & Graphics, (2025), p. 104223.

[44] G. Farin, Curves and Surfaces for Computer-Aided Geometric Design, Academic Press, 1990.

[45] L. Feng, X. Yang, and S. Xiao, Magictoon: A 2d-to-3d creative cartoon modeling system with mobile ar, in 2017 IEEE Virtual Reality (VR), IEEE, 2017, pp. 195–204.

[46] M. S. Floater, Mean value coordinates, Computer aided geometric design, 20 (2003), pp. 19–27.

[47] A. Frezzato, A. Tangri, and S. Andrews, Synthesizing get-up motions for physics-based characters, in Computer Graphics Forum, vol. 41, Wiley Online Library, 2022, pp. 207–218.

[48] H. Fu, S. Zhou, L. Liu, and N. J. Mitra, Animated construction of line drawings, in Proceedings of the 2011 SIGGRAPH Asia Conference, 2011, pp. 1–10.

[49] X. Fu, X. Wang, J. Liu, S. Wang, J. Dai, and J. Han, Cop: Chain-of-pose for image animation in large pose changes, in Proceedings of the 31st ACM International Conference on Multimedia, 2023, pp. 6969–6977.

[50] C. Furusawa, T. Fukusato, N. Okada, T. Hirai, and S. Morishima, Quasi 3d rotation for hand-drawn characters, in ACM SIGGRAPH 2014 Posters, 2014, pp. 1–1.

[51] O. Gafni, L. Wolf, and Y. Taigman, Vid2game: Controllable characters extracted from real-world videos, arXiv preprint arXiv:1904.08379, (2019).

[52] R. Gal, Y. Vinker, Y. Alaluf, A. Bermano, D. Cohen-Or, A. Shamir, and G. Chechik, Breathing life into sketches using text-to-video priors, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 4325–4336.

[53] R. Gal, Y. Vinker, Y. Alaluf, A. H. Bermano, D. Cohen-Or, A. Shamir, and G. Chechik, Breathing life into sketches using text-to-video priors, arXiv preprint arXiv:2311.13608, (2023).

[54] Z. Geng, C. Cao, and S. Tulyakov, 3d guided fine-grained face manipulation, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 9821–9830.

[55] M. Gleicher, Retargetting motion to new characters, in Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98), ACM, 1998, pp. 33–42.

[56] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Pearson, 2008.

[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial networks, Communications of the ACM, 63 (2020), pp. 139–144.

[58] J. Guajardo, O. Bursalioglu, and D. B. Goldman, Generative ai for 2d character animation, in ACM SIGGRAPH 2024 Posters, 2024, pp. 1–2.

[59] M. Guay, M.-P. Cani, and R. Ronfard, The line of action: an intuitive interface for expressive character posing, ACM Transactions on Graphics (TOG), 32 (2013), pp. 1–8.

[60] M. Guay, R. Ronfard, M. Gleicher, and M.-P. Cani, Adding dynamics to sketch-based character animations, in Sketch-based interfaces and modeling (SBIM) 2015, Eurographics Association, 2015, pp. 27–34.

[61] ——, Space-time sketching of character animation, ACM Transactions on Graphics (ToG), 34 (2015), pp. 1–10.

[62] A. Gunturu, Y. Wen, N. Zhang, J. Thundathil, R. H. Kazi, and R. Suzuki, Augmented physics: Creating interactive and embedded physics simulations from static textbook diagrams, in Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, 2024, pp. 1–12.

[63] Y. Guo, C. Yang, A. Rao, Y. Wang, Y. Qiao, D. Lin, and B. Dai, Animatediff: Animate your personalized text-to-image diffusion models without specific tuning, arXiv preprint arXiv:2307.04725, (2023).

[64] Z. Guo, O. Zhang, J. Xiang, A. Zhao, W. Zhou, and H. Li, Make-it-poseable: Feed-forward latent posing model for 3d humanoid character animation, arXiv preprint arXiv:2512.16767, (2025).

[65] H. Gupta and P. Chaudhuri, Sheetanim-from model sheets to 2d hand-drawn character animation-., in VISIGRAPP (1: GRAPP), 2018, pp. 17–27.

[66] D. Ha and D. Eck, A neural representation of sketch drawings, arXiv preprint arXiv:1704.03477, (2017).

[67] S. Ha, M. Kersner, B. Kim, S. Seo, and D. Kim, Marionette: Few-shot face reenactment preserving identity of unseen targets, in Proceedings of the AAAI conference on artificial intelligence, vol. 34, 2020, pp. 10893–10900.

[68] F. Hahn, F. Mutzel, S. Coros, B. Thomaszewski, M. Nitti, M. Gross, and R. W. Sumner, Sketch abstractions for character posing, in Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2015, pp. 185–191.

[69] R. Held, A. Gupta, B. Curless, and M. Agrawala, 3d puppetry: a kinect-based interface for 3d animation., in UIST, vol. 12, Citeseer, 2012, pp. 423–434.

[70] A. Hermans, L. Beyer, and B. Leibe, In defense of the triplet loss for person re-identification, arXiv preprint arXiv:1703.07737, (2017).

[71] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, Gans trained by a two time-scale update rule converge to a local nash equilibrium, Advances in neural information processing systems, 30 (2017).

[72] T. Hinz, M. Fisher, O. Wang, E. Shechtman, and S. Wermter, Charactergan: Few-shot keypoint character animation and reposing, in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 1988–1997.

[73] J. Ho, A. Jain, and P. Abbeel, Denoising diffusion probabilistic models, Advances in neural information processing systems, 33 (2020), pp. 6840–6851.

[74] D. Holden, J. Saito, and T. Komura, A deep learning framework for character motion synthesis and editing, ACM Transactions on Graphics (ToG), 35 (2016), pp. 1–11.

[75] L. Hu, Animate anyone: Consistent and controllable image-to-video synthesis for character animation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 8153–8163.

[76] L. Hu, G. Wang, Z. Shen, X. Gao, D. Meng, L. Zhuo, P. Zhang, B. Zhang, and L. Bo, Animate anyone 2: High-fidelity character image animation with environment affordance, arXiv preprint arXiv:2502.06145, (2025).

[77] Y. Hu, Z. Chen, and C. Luo, Lamd: Latent motion diffusion for video generation, arXiv preprint arXiv:2304.11603, (2023).

[78] Y. Hu, C. Luo, and Z. Chen, Make it move: controllable image-to-video generation with text descriptions, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18219–18228.

[79] Z. Hu, H. Xie, T. Fukusato, T. Sato, and T. Igarashi, Sketch2vf: Sketch-based flow design with conditional generative adversarial network, Computer Animation and Virtual Worlds, 30 (2019), p. e1889.

[80] Z. Huang, H. Feng, Y.-T. Sun, Y.-C. Guo, Y.-P. Cao, and L. Sheng, Animax: Animating the inanimate in 3d with joint video-pose diffusion models, in Proceedings of the SIGGRAPH Asia 2025 Conference Papers, 2025, pp. 1–13.

[81] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, Comparing images using the hausdorff distance, IEEE Transactions on Pattern Analysis and Machine Intelligence, 15 (1993), pp. 850–863.

[82] T. Igarashi, T. Moscovich, and J. F. Hughes, As-rigid-as-possible shape manipulation, ACM transactions on Graphics (TOG), 24 (2005), pp. 1134–1141.

[83] A. Jacobson, I. Baran, J. Popovic, and O. Sorkine, Bounded biharmonic weights for real-time deformation., ACM Trans. Graph., 30 (2011), p. 78.

[84] A. Jain, A. Xie, and P. Abbeel, Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 1911–1920.

[85] E. Jain, Y. Sheikh, M. Mahler, and J. Hodgins, Augmenting hand animation with three-dimensional secondary motion, in Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2010, pp. 93–102.

[86] J. Jiang, H. S. Seah, and H. Z. Liew, Stroke-based drawing and inbetweening with boundary strokes, in Computer Graphics Forum, vol. 41, Wiley Online Library, 2022, pp. 257–269.

[87] O. Johnston and F. Thomas, The illusion of life: Disney animation, (1981).

[88] B. Jones, J. Popovic, J. McCann, W. Li, and A. Bargteil, Dynamic sprites, in Proceedings of Motion on Games, 2013, pp. 39–46.

[89] H. W. Kang, W. He, C. K. Chui, and U. K. Chakraborty, Interactive sketch generation, The Visual Computer, 21 (2005), pp. 821–830.

[90] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht, Cotracker: It is better to track together, arXiv preprint arXiv:2307.07635, (2023).

[91] R. H. Kazi, F. Chevalier, T. Grossman, and G. Fitzmaurice, Kitty: sketching dynamic and interactive illustrations, in Proceedings of the 27th annual ACM symposium on User interface software and technology, 2014, pp. 395–405.

[92] R. H. Kazi, F. Chevalier, T. Grossman, S. Zhao, and G. Fitzmaurice, Draco: bringing life to illustrations with kinetic textures, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2014, pp. 351–360.

[93] R. H. Kazi, K. C. Chua, S. Zhao, R. Davis, and K.-L. Low, Sandcanvas: A multi-touch art medium inspired by sand animation, in

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2011, pp. 1283–1292.

[94] R. H. Kazi, T. Grossman, N. Umetani, and G. Fitzmaurice, Motion amplifiers: sketching dynamic illustrations using the principles of 2d animation, in Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, 2016, pp. 4599–4609.

[95] R. H. Kazi, T. Igarashi, S. Zhao, and R. Davis, Vignette: interactive texture design and manipulation with freeform gestures for pen-and-ink illustration, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2012, pp. 1727–1736.

[96] A. Khandelwal, Flexiclip: Locality-preserving free-form character animation, arXiv preprint arXiv:2501.08676, (2025).

[97] T.-Y. Kim and E. Vendrovsky, Drivenshape: a data-driven approach for shape deformation, in Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2008, pp. 49–55.

[98] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, (2014).

[99] D. P. Kingma and M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114, (2013).

[100] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al., Segment anything, arXiv preprint arXiv:2304.02643, (2023).

[101] M. Kitagawa and B. Windsor, MoCap for artists: workflow and techniques for motion capture, Routledge, 2020.

[102] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks, in Advances in Neural Information Processing Systems, 2012.

[103] J. Lasseter, Principles of traditional animation applied to 3d computer animation, in Proceedings of the 14th annual conference on Computer graphics and interactive techniques, 1987, pp. 35–44.

[104] ——, Principles of traditional animation applied to 3d computer animation, in Seminal graphics: pioneering efforts that shaped the field, 1998, pp. 263–272.

[105] E. M. Li, A. Qi, M. Sousa, and T. Grossman, Enchantedbrush: Animating in mixed reality for storytelling and communication, in Graphics Interface 2023-second deadline.

[106] L. Li, C. Zou, Y. Zheng, Q. Su, H. Fu, and C.-L. Tai, Sketch-r2cnn: An attentive network for vector sketch recognition.

[107] ——, Sketch-r2cnn: An rnn-rasterization-cnn architecture for vector sketch recognition, IEEE transactions on visualization and computer graphics, 27 (2020), pp. 3745–3754.

[108] Q. L. Li, W. D. Geng, T. Yu, X. J. Shen, N. Lau, and G. Yu, Motionmaster: authoring and choreographing kung-fu motions by sketch drawings, in Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation, 2006, pp. 233–241.

[109] W. Li, X. Zhang, Z. Sun, D. Qi, H. Li, W. Cheng, W. Cai, S. Wu, J. Liu, Z. Wang, et al., Step1x-3d: Towards high-fidelity and controllable generation of textured 3d assets, arXiv preprint arXiv:2505.07747, (2025).

[110] X. Li, Q. Ma, T.-Y. Lin, Y. Chen, C. Jiang, M.-Y. Liu, and D. Xiang, Articulated kinematics distillation from video diffusion models, in Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 17571–17581.

[111] Y. Li, M. Gleicher, Y.-Q. Xu, and H.-Y. Shum, Stylizing motion with drawings, in Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, 2003, pp. 309–319.

[112] Y. Li and T. Harada, Non-rigid point cloud registration with neural deformation pyramid, Advances in Neural Information Processing Systems, 35 (2022), pp. 27757–27768.

[113] Y. Li, M. Min, D. Shen, D. Carlson, and L. Carin, Video generation from text, in Proceedings of the AAAI conference on artificial intelligence, vol. 32, 2018.

[114] G. Liang, J. Hu, X. Xing, J. Zhang, and Q. Yu, Multi-object sketch animation with grouping and motion trajectory priors, arXiv preprint arXiv:2508.15535, (2025).

[115] L. Lingens, R. W. Sumner, and S. Magnenat, Towards automatic drawing animation using physics-based evolution, in Proceedings of the 2020 ACM Interaction Design and Children Conference: Extended Abstracts, 2020, pp. 314–319.

[116] C. Liu and M. Bessmeltsev, State-of-the-art report in sketch processing, in Computer Graphics Forum, Wiley Online Library, 2025, p. e70079.

[117] J. Liu, Z. Xin, Y. Fu, R. Zhao, B. Lan, and X. Li, Multi-object sketch animation by scene decomposition and motion planning, arXiv preprint arXiv:2503.19351, (2025).

[118] Y. Liu, M. De Nadai, G. Zen, N. Sebe, and B. Lepri, Gesture-to-gesture translation in the wild via category-independent conditional maps, in Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 1916–1924.

[119] Z. Liu, Y. Meng, H. Ouyang, Y. Yu, B. Zhao, D. Cohen-Or, and H. Qu, Dynamic typography: Bringing text to life via video diffusion prior, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2025, pp. 14787–14797.

[120] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, Video frame interpolation via adaptive separable convolution, in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.

[121] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, Smpl: A skinned multi-person linear model, in Seminal Graphics Papers: Pushing the Boundaries, Volume 2, 2023, pp. 851–866.

[122] D. Lorenz, L. Bereska, T. Milbich, and B. Ommer, Unsupervised part-based disentangling of object shape and appearance, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 10955–10964.

[123] I. Loshchilov and F. Hutter, Decoupled weight decay regularization, arXiv preprint arXiv:1711.05101, (2017).

[124] Y. Ma, G. Xu, X. Sun, M. Yan, J. Zhang, and R. Ji, X-clip: End-to-end multi-grained contrastive learning for video-text retrieval, in Proceedings of the 30th ACM international conference on multimedia, 2022, pp. 638–647.

[125] N. Magnenat-Thalmann, R. Laperrière, and D. Thalmann, Joint-dependent local deformations for hand animation and object grasping, in Proceedings on Graphics interface'88, 1989, pp. 26–33.

[126] S. Maheshwari, R. Narain, and R. Hebbalaguppe, Transfer4d: A framework for frugal motion capture and deformation transfer, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 12836–12846.

[127] H. Mo, C. Gao, and R. Wang, Joint stroke tracing and correspondence for 2d animation, ACM Transactions on Graphics, 43 (2024), pp. 1–17.

[128] R. Mokady, R. Tzaban, S. Benaim, A. Bermano, and D. Cohen-Or, Jokr: Joint keypoint representation for unsupervised video retargeting, in Computer Graphics Forum, vol. 41, Wiley Online Library, 2022, pp. 245–257.

[129] T. Moscovich and J. F. Hughes, Animation sketching: An approach to accessible animation, Unpublished Master's Thesis, CS Department, Brown University, 5 (2001).

[130] Y. Mu, Z. Zhang, Y. Shi, M. Matsumoto, K. Imamura, G. Tevet, C. Guo, M. Taylor, C. Shu, P. Xi, et al., Smp: Reusable score-matching motion priors for physics-based character control, arXiv preprint arXiv:2512.03028, (2025).

[131] A. Nagrani, J. S. Chung, and A. Zisserman, Voxceleb: a large-scale speaker identification dataset, arXiv preprint arXiv:1706.08612, (2017).

[132] R. Narita, K. Hirakawa, and K. Aizawa, Optical flow based line drawing frame interpolation using distance transform to support inbetweenings, in 2019 IEEE International Conference on Image Processing (ICIP), IEEE, 2019, pp. 4200–4204.

[133] B. Ni, H. Peng, M. Chen, S. Zhang, G. Meng, J. Fu, S. Xiang, and H. Ling, Expanding language-image pretrained models for general video recognition, in European Conference on Computer Vision, Springer, 2022, pp. 1–18.

[134] H. Ni, C. Shi, K. Li, S. X. Huang, and M. R. Min, Conditional image-to-video generation with latent flow diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 18444–18455.

[135] Y. Nirkin, Y. Keller, and T. Hassner, Fsgan: Subject agnostic face swapping and reenactment, in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 7184–7193.

[136] M. Okabe, K. Anjyo, T. Igarashi, and H.-P. Seidel, Animating pictures of fluid using video examples, in Computer Graphics Forum, vol. 28, Wiley Online Library, 2009, pp. 677–686.

[137] J. Pan and J. J. Zhang, Sketch-based skeleton-driven 2D animation and motion capture, Springer, 2011.

[138] P. Patel, H. Gupta, and P. Chaudhuri, Tracemove: A data-assisted interface for sketching 2d character animation, in Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications: Volume 1: GRAPP, 2016, pp. 191–199.

[139] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, DreamFusion: Text-to-3D using 2D diffusion, arXiv, (2022).

[140] Y. Qi and Z.-H. Tan, Sketchsegnet+: An end-to-end learning of rnn for multi-class sketch semantic segmentation, Ieee Access, 7 (2019), pp. 102717–102726.

[141] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in International conference on machine learning, PMLR, 2021, pp. 8748–8763.

[142] G. Rai, S. Gupta, and O. Sharma, Sketchanim: Real-time sketch animation transfer from videos, in Computer Graphics Forum, Wiley Online Library, 2024, p. e15176.

[143] G. Rai and O. Sharma, Enhancing sketch animation: Text-to-video diffusion models with temporal consistency and rigidity constraints, arXiv preprint arXiv:2411.19381, (2024).

[144] K. Ranjan, P. Gyory, S. Howell, T. Stewart, M. L. Rivera, and E. Y.-L. Do, Cartoonimator: A paper-based tangible kit for keyframe animation, in Proceedings of the Nineteenth International Conference on Tangible, Embedded, and Embodied Interaction, 2025, pp. 1–16.

[145] B. Reinert, T. Ritschel, and H.-P. Seidel, Animated 3d creatures from single-view video by skeletal sketching., in Graphics Interface, 2016, pp. 133–141.

[146] Y. Ren, X. Yu, J. Chen, T. H. Li, and G. Li, Deep image spatial transformation for person image generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 7690–7699.

[147] K. T. Rosenberg, R. H. Kazi, L.-Y. Wei, H. Xia, and K. Perlin, Drawtalking: Building interactive worlds by sketching and speaking, in Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, 2024, pp. 1–25.

[148] P. Sangkloy, N. Burnell, C. Ham, and J. Hays, The sketchy database: learning to retrieve badly drawn bunnies, ACM Transactions on Graphics (TOG), 35 (2016), pp. 1–12.

[149] S. Santosa, F. Chevalier, R. Balakrishnan, and K. Singh, Direct space-time trajectory control for visual media editing, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2013, pp. 1149–1158.

[150] R. K. Sarvadevabhatla, S. Suresh, and R. V. Babu, Object category understanding via eye fixations on freehand sketches, IEEE Transactions on Image Processing, 26 (2017), pp. 2508–2518.

[151] R. Schwartz, M. Mullery, J. Dingliana, and R. McDonnell, Computational topology for hand-drawn animation technology: A survey, Computers & Graphics, (2025), p. 104225.

[152] J. Scott and R. Davis, Physink: sketching physical behavior, in Adjunct Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology, 2013, pp. 9–10.

[153] A. Selle, M. Lentine, and R. Fedkiw, A mass spring model for hair simulation, in ACM SIGGRAPH 2008 papers, 2008, pp. 1–11.

[154] J. Shen, K. Hu, W. Bao, C. W. Chen, and Z. Wang, Bridging the gap: Sketch-aware interpolation network for high-quality animation sketch inbetweening, in Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 10287–10295.

[155] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe, Animating arbitrary objects via deep motion transfer, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2377–2386.

[156] ——, First order motion model for image animation, Advances in Neural Information Processing Systems, 32 (2019).

[157] A. Siarohin, W. Menapace, I. Skorokhodov, K. Olszewski, J. Ren, H.-Y. Lee, M. Chai, and S. Tulyakov, Unsupervised volumetric animation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 4658–4669.

[158] A. Siarohin, E. Sangineto, S. Lathuiliere, and N. Sebe, Deformable gans for pose-based human image generation, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3408–3416.

[159] A. Siarohin, O. J. Woodford, J. Ren, M. Chai, and S. Tulyakov, Motion representations for articulated animation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13653–13662.

[160] E. Sifakis, T. Shinar, G. Irving, and R. Fedkiw, Hybrid simulation of deformable solids, in Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, 2007, pp. 81–90.

[161] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, in International Conference on Learning Representations, 2015.

[162] L. Siyao, T. Gu, W. Xiao, H. Ding, Z. Liu, and C. C. Loy, Deep geometrized cartoon line inbetweening, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 7291–7300.

[163] H. J. Smith, N. He, and Y. Ye, Animating childlike drawings with 2.5 d character rigs, arXiv preprint arXiv:2502.17866, (2025).

[164] H. J. Smith, Q. Zheng, Y. Li, S. Jain, and J. K. Hodgins, A method for animating children's drawings of the human figure, ACM Transactions on Graphics, 42 (2023), pp. 1–15.

[165] E. Sohn and Y.-C. Choy, Sketch-n-stretch: Sketching animations using cutouts, IEEE Computer Graphics and Applications, 32 (2012), pp. 59–69.

[166] C. Song, X. Li, F. Yang, Z. Xu, J. Wei, F. Liu, J. Feng, G. Lin, and J. Zhang, Puppeteer: Rig and animate your 3d models, Advances in Neural Information Processing Systems, (2025).

[167] J. Song, C. Meng, and S. Ermon, Denoising diffusion implicit models, arXiv preprint arXiv:2010.02502, (2020).

[168] Y.-P. Song, Y.-T. Liu, X. Wu, Q. He, Z. Yuan, and A. Luo, Magiccartoon: 3d pose and shape estimation for bipedal cartoon characters, in Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 8219–8227.

[169] S. Starke, I. Mason, and T. Komura, Deepphase: Periodic autoencoders for learning motion phase manifolds, ACM Transactions on Graphics (ToG), 41 (2022), pp. 1–13.

[170] Q. Su, X. Bai, H. Fu, C.-L. Tai, and J. Wang, Live sketch: Video-driven dynamic deformation of static drawings, in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, 2018, pp. 1–12.

[171] Q. Sun, Y. Ni, T. Yuan, J. Zhang, F. Yang, Z. Yao, and H. Mi, Spiritus: An ai-assisted tool for creating 2d characters and animations, arXiv preprint arXiv:2503.09127, (2025).

[172] Q. Sun, C. Wang, J. Shang, W. Feng, and J. Liao, Animus3d: Text-driven 3d animation via motion score distillation, in Proceedings of the SIGGRAPH Asia 2025 Conference Papers, 2025, pp. 1–11.

[173] S. Suwajanakorn, N. Snavely, J. J. Tompson, and M. Norouzi, Discovery of latent 3d keypoints via end-to-end geometric reasoning, Advances in neural information processing systems, 31 (2018).

[174] R. Suzuki, R. H. Kazi, L.-Y. Wei, S. DiVerdi, W. Li, and D. Leithinger, Realitysketch: Embedding responsive graphics and visualizations in ar through dynamic sketching, in Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology, 2020, pp. 166–181.

[175] H. Tang, W. Wang, D. Xu, Y. Yan, and N. Sebe, Gesturegan for hand gesture-to-gesture translation in the wild, in Proceedings of the 26th ACM international conference on Multimedia, 2018, pp. 774–782.

[176] Z. Tang, Z. Yang, C. Zhu, M. Zeng, and M. Bansal, Any-to-any generation via composable diffusion, Advances in Neural Information Processing Systems, 36 (2024).

[177] J. Tao, B. Wang, T. Ge, Y. Jiang, W. Li, and L. Duan, Motion transformer for unsupervised image animation, in Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XVI, Springer, 2022, pp. 702–719.

[178] J. Tao, B. Wang, B. Xu, T. Ge, Y. Jiang, W. Li, and L. Duan, Structure-aware motion transfer with deformable anchor model, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 3637–3646.

[179] G. Tevet, S. Raab, B. Gordon, Y. Shafir, D. Cohen-Or, and A. H. Bermano, Human motion diffusion model, arXiv preprint arXiv:2209.14916, (2022).

[180] A. Thiault, T. Philippe, A. D. Parakkat, E. Eisemann, R. Muthuganapathy, and T. Igarashi, Spineloft: Interactive spine-based 2d-to-3d modeling, in CHI'25: Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems, 2025.

[181] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Niessner, Face2face: Real-time face capture and reenactment of rgb videos, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2387–2395.

[182] Y. Tian, J. Ren, M. Chai, K. Olszewski, X. Peng, D. N. Metaxas, and S. Tulyakov, A good image generator is what you need for high-resolution video synthesis, arXiv preprint arXiv:2104.15069, (2021).

[183] B. Tversky, J. B. Morrison, and M. Betrancourt, Animation: can it facilitate?, International journal of human-computer studies, 57 (2002), pp. 247–262.

[184] T. Unterthiner, S. van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly, Fvd: A new metric for video generation, (2019).

[185] M. Unuma, K. Anjyo, and R. Takeuchi, Fourier principles for emotion-based human figure animation, in Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, 1995, pp. 91–96.

[186] C. Wang, C. Xu, and D. Tao, Self-supervised pose adaptation for cross-domain image animation, IEEE Transactions on Artificial Intelligence, 1 (2020), pp. 34–46.

[187] F. Wang, S. Lin, H. Wu, H. Li, R. Wang, X. Luo, and X. He, Spfusionnet: Sketch segmentation using multi-modal data fusion, in 2019 IEEE International Conference on Multimedia and Expo (ICME), IEEE, 2019, pp. 1654–1659.

[188] J. Wang, H. Yuan, D. Chen, Y. Zhang, X. Wang, and S. Zhang, Modelscope text-to-video technical report, arXiv preprint arXiv:2308.06571, (2023).

[189] M. Wang, Q. Yan, Z. Cao, Y. Li, O. Mac Aodha, J. J. Corso, and A. Vaxman, Bimotion: B-spline motion for text-guided dynamic 3d character generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2026.

[190] W. Wang, X. Alameda-Pineda, D. Xu, P. Fua, E. Ricci, and N. Sebe, Every smile is unique: Landmark-guided diverse smile generation, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7083–7092.

[191] Y. Wang, D. Yang, F. Bremond, and A. Dantcheva, Latent image animator: Learning to animate images via latent space navigation, arXiv preprint arXiv:2203.09043, (2022).

[192] Y. Wang, D. Yang, F. Bremond, and A. Dantcheva, Lia: Latent image animator, IEEE Transactions on Pattern Analysis and Machine Intelligence, (2024), pp. 1–16.

[193] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, Image quality assessment: From error visibility to structural similarity, IEEE Transactions on Image Processing, 13 (2004), pp. 600–612.

[194] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, Convolutional pose machines, in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2016, pp. 4724–4732.

[195] C.-Y. Weng, B. Curless, and I. Kemelmacher-Shlizerman, Photo wake-up: 3d character animation from a single photo, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 5908–5917.

[196] B. Whited, G. Noris, M. Simmons, R. W. Sumner, M. Gross, and J. Rossignac, Betweenit: An interactive tool for tight inbetweening, in Computer Graphics Forum, vol. 29, Wiley Online Library, 2010, pp. 605–614.

[197] O. Wiles, A. Koepke, and A. Zisserman, X2face: A network for controlling face generation using images, audio, and pose codes, in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 670–686.

[198] N. S. Willett, R. H. Kazi, M. Chen, G. Fitzmaurice, A. Finkelstein, and T. Grossman, A mixed-initiative interface for animating static pictures, in Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology, pp. 649–661.

[199] N. S. Willett, R. H. Kazi, M. Chen, G. Fitzmaurice, A. Finkelstein, and T. Grossman, A mixed-initiative interface for animating static pictures, in Proceedings of the 31st annual ACM symposium on user interface software and technology, 2018, pp. 649–661.

[200] N. S. Willett, W. Li, J. Popovic, F. Berthouzoz, and A. Finkelstein, Secondary motion for performed 2d animation, in Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, 2017, pp. 97–108.

[201] N. S. Willett, W. Li, J. Popovic, and A. Finkelstein, Triggering artwork swaps for live animation, in Proceedings of the 30th annual ACM symposium on User interface software and technology, 2017, pp. 85–95.

[202] N. S. Willett, H. V. Shin, Z. Jin, W. Li, and A. Finkelstein, Pose2pose: Pose selection and transfer for 2d character animation, in Proceedings of the 25th International Conference on Intelligent User Interfaces, 2020, pp. 88–99.

[203] A. Witkin and M. Kass, Spacetime constraints, ACM Siggraph Computer Graphics, 22 (1988), pp. 159–168.

[204] C. Wu, L. Huang, Q. Zhang, B. Li, L. Ji, F. Yang, G. Sapiro, and N. Duan, Godiva: Generating open-domain videos from natural descriptions, arXiv preprint arXiv:2104.14806, (2021).

[205] R. Wu, W. Su, K. Ma, and J. Liao, Aniclipart: Clipart animation with text-to-video priors, International Journal of Computer Vision, 133 (2025), pp. 3149–3165.

[206] Z. Wu, C. Yu, F. Wang, and X. Bai, Animateanymesh: A feed-forward 4d foundation model for text-driven universal mesh animation, arXiv preprint arXiv:2506.09982, (2025).

[207] Z. Xia, K. Monteiro, K. Van, and R. Suzuki, Realitycanvas: Augmented reality sketching for embedded and responsive scribble animation effects, in Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, 2023, pp. 1–14.

[208] L. Xie, Z. Zhou, K. Yu, Y. Wang, H. Qu, and S. Chen, Wakey-wakey: Animate text by mimicking characters in a gif, in Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, 2023, pp. 1–14.

[209] T. Xie, Y. Chen, Y. Guo, Y. Yang, B. Zhou, D. Terzopoulos, Y. Jiang, and C. Jiang, Animamimic: Imitating 3d animation from video priors, arXiv preprint arXiv:2512.14133, (2025).

[210] Z. Xie, H. Mo, and C. Gao, Video-driven sketch animation via cyclic reconstruction mechanism, in 2024 IEEE International Conference on Multimedia and Expo (ICME), IEEE, 2024, pp. 1–6.

[211] J. Xing, R. H. Kazi, T. Grossman, L.-Y. Wei, J. Stam, and G. Fitzmaurice, Energy-brushes: Interactive tools for illustrating stylized elemental dynamics, in Proceedings of the 29th Annual Symposium on User Interface Software and Technology, 2016, pp. 755–766.

[212] J. Xing, L.-Y. Wei, T. Shiratori, and K. Yatani, Autocomplete hand-drawn animations, ACM Transactions on Graphics (TOG), 34 (2015), pp. 1–11.

[213] J. Xing, M. Xia, Y. Zhang, H. Chen, X. Wang, T.-T. Wong, and Y. Shan, Dynamicrafter: Animating open-domain images with video diffusion priors, arXiv preprint arXiv:2310.12190, (2023).

[214] X. Xing, C. Wang, H. Zhou, J. Zhang, Q. Yu, and D. Xu, Diffsketcher: Text guided vector sketch synthesis through latent diffusion models, Advances in Neural Information Processing Systems, 36 (2024).

[215] B. Xu, B. Wang, J. Deng, J. Tao, T. Ge, Y. Jiang, W. Li, and L. Duan, Motion and appearance adaptation for cross-domain motion transfer, in Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XVI, Springer, 2022, pp. 529–545.

[216] Z. Xu, J. Zhang, J. H. Liew, H. Yan, J.-W. Liu, C. Zhang, J. Feng, and M. Z. Shou, Magicanimate: Temporally consistent human image animation using diffusion model, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 1481–1490.

[217] W. Yan, Y. Zhang, P. Abbeel, and A. Srinivas, Videogpt: Video generation using vq-vae and transformers, arXiv preprint arXiv:2104.10157, (2021).

[218] R. Yang, D. Li, H. Zhang, and Y.-Z. Song, Sketchanimator: Animate sketch via motion customization of text-to-video diffusion models, in 2024 IEEE International Conference on Visual Communications and Image Processing (VCIP), IEEE, 2024, pp. 1–5.

[219] W. Yang, Context-aware computer aided inbetweening, IEEE transactions on visualization and computer graphics, 24 (2017), pp. 1049–1062.

[220] W. Yang, J. Feng, and X. Wang, Structure preserving manipulation and interpolation for multi-element 2d shapes, in Computer Graphics Forum, vol. 31, Wiley Online Library, 2012, pp. 2249–2258.

[221] S. Yoon, G. Koo, Y. Lee, J. W. Hong, and C. D. Yoo, Occlusion-robust stylization for drawing-based 3d animation, arXiv preprint arXiv:2508.00398, (2025).

[222] J. Yu, D. Liu, D. Tao, and H. S. Seah, Complex object correspondence construction in two-dimensional animation, IEEE Transactions on Image Processing, 20 (2011), pp. 3257–3269.

[223] Y. Yuan, J. Song, U. Iqbal, A. Vahdat, and J. Kautz, Physdiff: Physics-guided human motion diffusion model, in Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 16010–16021.

[224] P. Zablotskaia, A. Siarohin, B. Zhao, and L. Sigal, Dwnet: Dense warp-based network for pose-guided human video generation, arXiv preprint arXiv:1910.09139, (2019).

[225] B. Zeng, X. Liu, S. Gao, B. Liu, H. Li, J. Liu, and B. Zhang, Face animation with an attribute-guided diffusion model, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 628–637.

[226] C. Zhang, L. Yang, N. Chen, N. Vining, A. Sheffer, F. C. Lau, G. Wang, and W. Wang, Creatureshop: Interactive 3d character modeling and texturing from a single color drawing, IEEE Transactions on Visualization and Computer Graphics, 29 (2022), pp. 4874–4890.

[227] M. Zhang, Z. Cai, L. Pan, F. Hong, X. Guo, L. Yang, and Z. Liu, Motiondiffuse: Text-driven human motion generation with diffusion model, IEEE transactions on pattern analysis and machine intelligence, 46 (2024), pp. 4115–4128.

[228] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 586–595.

[229] J. Zhao and H. Zhang, Thin-plate spline motion model for image animation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 3657–3666.

[230] Q. Zhao, P. Li, W. Yifan, S.-H. Olga, and G. Wetzstein, Pose-to-motion: Cross-domain motion retargeting with pose prior, in Computer Graphics Forum, vol. 43, Wiley Online Library, 2024, p. e15170.

[231] Z. Zhao, Z. Lai, Q. Lin, Y. Zhao, H. Liu, S. Yang, Y. Feng, M. Yang, S. Zhang, X. Yang, et al., Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, arXiv preprint arXiv:2501.12202, (2025).

[232] J. Zheng and X. Cun, Fairygen: Storied cartoon video from a single child-drawn character, arXiv preprint arXiv:2506.21272, (2025).

[233] L. Zhong, C. Guo, Y. Xie, J. Wang, and C. Li, Sketch2anim: Towards transferring sketch storyboards into 3d animation, ACM Transaction on Graphics (TOG), 44 (2025), pp. 1–15.

[234] D. Zhou, W. Wang, H. Yan, W. Lv, Y. Zhu, and J. Feng, Magicvideo: Efficient video generation with latent diffusion models, arXiv preprint arXiv:2211.11018, (2022).

[235] J. Zhou, L. Qu, M.-L. Lam, and H. Fu, From rigging to waving: 3d-guided diffusion for natural animation of hand-drawn characters, ACM Transactions on Graphics (TOG), 44 (2025), pp. 1–11.

[236] J. Zhou, C. Xiao, M.-L. Lam, and H. Fu, Drawingspinup: 3d animation from single character drawings, in SIGGRAPH Asia 2024 Conference Papers, 2024, pp. 1–10.

[237] Q. Zhou, D. Ledo, G. Fitzmaurice, and F. Anderson, Timetunnel: Integrating spatial and temporal motion editing for character animation in virtual reality, in Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems, 2024, pp. 1–17.

[238] B. Zhu, M. Iwata, R. Haraguchi, T. Ashihara, N. Umetani, T. Igarashi, and K. Nakazawa, Sketch-based dynamic illustration of fluid systems, in Proceedings of the 2011 SIGGRAPH Asia Conference, 2011, pp. 1–8.

[239] J. Zhu, H. Ma, J. Chen, and J. Yuan, Motionvideogan: A novel video generator based on the motion space learned from image pairs, IEEE Transactions on Multimedia, (2023).

[240] T. Zhu, W. Shang, and D. Ren, Thin-plate spline-based interpolation for animation line inbetweening, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 39, 2025, pp. 10995–11003.

[241] X. Zhu, X. Yang, S. Zheng, Z. Zhang, F. Gao, J. Huang, and J. Chen, Vector sketch animation generation with differentialable motion trajectories, arXiv preprint arXiv:2509.25857, (2025).

[242] Z. Zhu, T. Huang, B. Shi, M. Yu, B. Wang, and X. Bai, Progressive pose attention transfer for person image generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2347–2356.

[243] C. Zou, Q. Yu, R. Du, H. Mo, Y.-Z. Song, T. Xiang, C. Gao, B. Chen, and H. Zhang, Sketchyscene: Richly-annotated scene sketches, in Proceedings of the european conference on computer vision (ECCV), 2018, pp. 421–436.