Doctoral Dissertation

Smoke Effect Generation and Design with Sketch Control using Diffusion Models

Hengyuan Chang

Supervisor Haoran Xie

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March 2026

Abstract

2D smoke effects are ubiquitous visual elements in graphic design, illustration-oriented creation and animation. In practice, conventional 2D smoke design is generated using procedural physical simulations with various parameters, which requires substantial expert knowledge of fluid dynamics. Non-expert users are required to transfer the creative intentions into modification of simulation parameters, which causes a semantic gap between design intent and the resulting smoke effect. In contrast, hand-drawn sketches provide a an intuitive and abstract representation of visual intent. For smoke effects, a few strokes can convey both salient structures and coarse motion cues, making abstract intent more concrete.

Fluid synthesis has been reformulated as a data-driven generation task due to advancements in conditional generative models. The generative models learn an end-to-end mapping from sketches to target outputs by leveraging paired training data. Still, previous sketch-guided flow field generation methods adopt a one-stage conditional generative adversarial network flow field generation framework using sketch as direct input. The training process may be unstable due to the adversarial generation process. One-stage mapping also offers insufficient control constraints during the generation process. Inferring dense flow fields from sparse sketches is ill-posed, one-stage generation process amplifies the uncertainty inherent in mapping sketches to flow fields. Two-stage flow field generation study invokes intermediate representation extracted from flow field to provide explicit constraint during the generation process. The prior studies rely on regional constraints including mask or filter, which lacks explicit consistency constraints on the motion information contained from flow field.

To address the aforementioned challenges, this dissertation connects the sketch and the smoke effect using diffusion model, and constrains the generation process by extracting underlying physical information from the flow field. The dissertation structure is organized as follows:

The first stage introduces sketch-guided flow field generation using diffusion models. The velocity field used for training are obtained via smoke simulation with simplified streamlines extracted as corresponding sketch representations. Diffusion models circumvent adversarial training between generator and discriminator by adopting a denoising framework for data distribution estimation, ensuring a stable training process. The proposed framework enables users to directly synthesize velocity fields from hand-

drawn sketches.

The second stage introduces two-stage sketch-guided smoke illustration generation using Lagrangian coherent structure (LCS). The process of generating a velocity field from a sketch is decomposed into two simplified subprocesses. The first process generates LCS from input sketches. The second process generates velocity field under the guidance of LCS. The LCS serves as an intermediate representation between sketch and velocity field. The LCS enhances regional control ability of the diffusion model. The generated velocity field can be used as a guiding force, which can be directly integrated into existing fluid simulation tools for smoke effect design.

The third stage introduces two-stage sketch-guided smoke illustration generation using stream function. The stream function carries the rotational information of the flow field and the geometry of the fluid. The stream function is used as an intermediate representation between the sketch and the velocity field, incorporating motion priors into velocity field generation. The generated velocity field is used as guidance force for smoke effect generation. The artistic stylization of smoke is achieved for 2D smoke illustration synthesis through a stylized pre-trained model.

The final stage introduces two-stage sketch-guided smoke video generation using stream function. In the first process, the sketch is used as input condition for generating the smoke video first frame and stream function using diffusion model. In the second process, the generated first frame, visualized stream function and text prompts are combined to control the smoke video generation through video diffusion model. A motion control module is incorporated to leverage the stream function as motion guidance for smoke video generation. In addition, a pre-trained stylization model is optionally provided for stylized smoke video synthesis.

This dissertation provides a sketch-guided approach to create smoke effects, offers a new perspective on balancing user creative intent with maintaining the consistency of generated physical contents. First, this dissertation utilizes diffusion model instead of generative adversarial network as training framework, improves the generation stability by circumventing the adversarial process. Second, this dissertation introduces two-stage generation strategy to decrease the velocity field generation ambiguity from sketch inputs. Third, this dissertation enhances the coherence of generated velocity fields by incorporating explicit motion guidance, moving beyond the reliance on regional constraints. Finally, this dissertation synthesizes 2D smoke animations via video diffusion model conditioned on motion guidance derived from sketch inputs, thereby obviating the requirements of simulation tools.

Keywords: Fluid Simulation, Diffusion Model, Sketch-Guided Generation,

Lagrangian Coherent Structure, Stream Function, Video Generation.

Acknowledgment

First and foremost, I would like to express my sincere gratitude to my primary supervisor, professor Haoran Xie. His dedication to research has been a constant source of support throughout my doctoral studies. His deep understanding of the subject matter helped me clarify my research goals and stay on the right track. Over the studies of my doctoral period, he has not only guided my research, but also taught me the attitude a researcher should have. Thanks to his guidance and his patience, I have been able to take my first steps toward becoming a qualified and responsible researcher.

I also thank the committee members in my doctoral defense, professor Naoya Inoue, professor Shinobu Hasegawa, professor Shogo Okada, and professor Matoko Fujisawa. Thanks for their advices and guidance. Their responsible attitude and generous support have been of immense help to my research and studies.

I would also thank my co-supervisor, professor Masashi Unoki. His rigorous and responsible attitude towards research have been of great help in writing my doctoral dissertation.

Furthermore, I sincerely thank my collaborating professor Syuhei Sato from Hosei University. His professional knowledge has helped advance my research. His different perspectives on the research have helped me examine my research from different angles and opened up new research approaches for me.

In addition, I would like to express my gratitude to Mr.Tianyu Zhang, Mr.Xiaoxuan Xie, Mr.Ryuichi Miyauchi, Dr.Yichen Peng, professor Tsukasa Fukusato and professor Kazunori Miyata, who participated in the projects covered in this dissertation. I am grateful for their conscientious and responsible attitude, which has been of great help to me in writing my papers.

At last, I would like to thank my family, friends, and the rest members of my lab. Their love and friendship have been my source of strength and support throughout my doctoral period.

List of Figures

List of Tables

Contents

Chapter 1

Smoke Effect Generation Introduction

This chapter first introduces the background and challenges of fluid simulation and generative approaches in fluid effect synthesis. By introducing sketches, this dissertation focuses on connecting sketches and fluids through generative models, and introduces the methodology and research framework of this dissertation. Finally, this chapter briefly introduces the subsequent chapters to clarify the overall content of this dissertation.

1.1 Background

Smoke is a distinctive gaseous fluid characterized by complex turbulent structures and buoyant dynamics continuum medium and commonly appears in the forms of liquids, gases, and plasmas. In nature, the rising plumes and the intricate dispersion of clouds are representative manifestations of smoke phenomena. Understanding the fluid dynamics behind the smoke phenomena is crucial for environmental studies and fire safety engineering, driven by the formulation of the Navier–Stokes equations and continuous progress in theoretical analysis, experimental measurement, and numerical computation. With the rapid advancement of computing hardware and the maturation of numerical methods for solving partial differential equations, smoke simulation has evolved from a scientific pursuit into a vital tool for the creative industries. In computer graphics, the physically based fluid solvers have been able to be integrated into production pipelines, supporting the synthesis of visually high-fidelity smoke effects from single smoke plume to atmospheric billowy smoke.

Smoke simulation approximates the continuous Navier–Stokes equations by formulating a discrete numerical scheme on a spatiotemporal domain, enabling the synthesis of physically plausible smoke motion and appearance. Smoke simulation has evolved into a key component of modern visual content creation, generate complex, dynamic motion while maintaining consistency

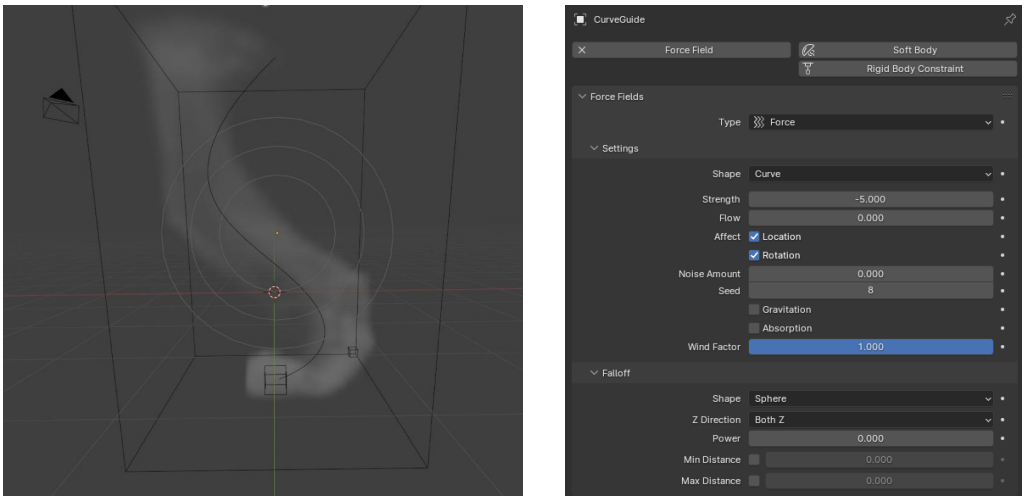

Figure 1.1: The example of fluid simulation tool implemented in Blender with parameters modification interface.

with physical principles. 3D smoke simulation is indispensable in industrial design and realistic effects creation, including aerodynamic design workflows and cinema-grade visual effects. where physical fidelity is critical but remains computationally expensive at high resolutions. In contrast, 2D smoke simulation offers a computationally efficient environment for smoke structure and dynamics analysis in a high resolution. The computational efficiency contributes to hypothesis verification in fluid dynamics research. In geophysical fluid dynamics, 2D smoke simulation is foundational to earth atmosphere analysis. Furthermore, the utility of 2D smoke simulation extends into non-photorealistic rendering, where for synthesizing stylized and artistically-driven smoke effects. Despite the widespread use of both 2D and 3D smoke solvers, smoke effect creation is still a challenging task.

Users design smoke effects by constructing simulation scenes within fluid simulation tools provided by commercial software such as Maya, Houdini, and Blender, or within game engines such as Unreal Engine and Unity3D. Users have to specify a variety of parameter controls while using the simulation tools, including emission sources, initial velocity, external forces, and boundary conditions. The smoke motion is then obtained by numerically evolving the underlying flow fields through the solution of discretized partial differential equations (PDEs) over time. Users are required to translate the conceptions of smoke structure and dynamics into intricate modeling procedures and parameter constraints for smoke simulation. An example of parameter-based control is shown in Figure 1.1. The intricate parameter

controlling limits the accessibility of integrating smoke effects into the visual creation by users, especially for those without a background in fluid dynamics and design expertise. Consequently, reducing creation process complexity and faithfully translating the user intent while preserving the consistency of fluid motion has been a demanded requirement.

1.2 Motivation

Replacing traditional parameter driven fluid effect design workflows with predefined visual components has emerged as an approach to bridge the gap between user intent and fluid design. Schpok et al. [1] defines different smoke motion features including vortex, advection, source and sink into elliptical components. Users are allowed to edit the components for simulation scene editing. Zhu et al. [2] enable users to compose and edit scenes by assembling components. These components serve distinct purposes, such as adding fluid sources, adjusting boundary conditions, and controlling fluid motion. Users are allowed to freely to modify the drawn components for fluid scenario editing. Xing et al. [3] define three brush components that encode distinct fluid motion behaviors, allowing users to produce fluid-like motion effects by painting on the canvas. The motion effects of different brushes can be superimposed to synthesize complex motion patterns. However, component-based fluid design systems are sensitive to user inputs. Users who are unfamiliar with these systems provide non-standard or imprecise inputs to the system, which may lead to unpredictable interactions or conflicts among components. The collisions between components cause the designing process fail or yield unsatisfactory results.

In traditional design process, hand-drawn sketch provides a compact semantic-level representation of visual concept. In fluid effect design, a sketch can describe both the salient structure and the motion direction guidance with only a few strokes and thereby concretize the abstract user intent. Sketch input offers an accessible means of specifying artistic intent compared to parameter tuning or scripting-based controls, particularly for non-expert users.

Recent progress in generative models further redefines fluid creation as a conditional generation task. Generative models [4] learns an end-to-end mapping from paired data and synthesize plausible outputs that are consistent with the control condition. This capability provides a novel approach for simplifying the fluid creation pipeline: user intent can be provided in a semantic level input such as sketch, while the model generates fluid data such as velocity field that can be directly integrated into numerical

simulators, or directly output the target fluid effect illustration and videos.

Nevertheless, current fluid generation studies largely rely on one-stage conditional generative adversarial network. The adversarial training process can be unstable, especially under limited data. The generated results may suffer from artifacts or inconsistent motion. More fundamentally, the mapping from sparse sketches to dense physically constrained flow fields is inherently ill-posed, which is challenging to simultaneously achieve plausible fluid data synthesis. These limitations lead to the key challenges discussed in the next section.

1.3 Technical Challenges

Existing learning-based sketch-guided fluid generation [5–7] still face several challenges that limit training stability, training interpretability, and physical consistency.

Training instability Prior sketch guided velocity field synthesis studies adopt conditional generative adversarial networks (cGANs) to learn a direct mapping from sketches to target flow representations. In a cGAN, two training process alternates between updating discriminator to improve discrimination ability and updating generator to synthesize real-like flow fields from given inputs. This adversarial process is challenging to optimize in practice even effectiveness has bee proved in image-to-image translation tasks. When the discriminator becomes overly strong in early training or the data are limited, gradients received by the generator can become weak or uninformative, leading to unstable convergence. Moreover, adversarial training is known to be susceptible to mode collapse [8], where the generator covers only a small subset of the target distribution and produces limited diversity. In sketch-guided velocity field generation, these issues typically manifest as noisy artifacts, inconsistent motion patterns, which collectively reduce reliability for practical authoring.

Limited interpretability Direct one-stage mapping from sparse sketches to dense flow fields is inherently ill-posed. This ill-posedness causes uncertain mapping between sketch and flow fields, which increases the risk of unstable or inconsistent outputs, especially when the sketch is sparse or deviates from the training distribution. Furthermore, one-stage end-to-end generation offers limited interpretability: Single-stage end-to-end generation has limited interpretability. A sketch provides sparse information about the underlying dynamics and coarse geometric structure. The mapping between sketch and

flow field exhibits excessive functional complexity. Specifically, this mapping must simultaneously achieve structural alignment (consistency between the sketch geometry and the flow topology), complete missing motion information (velocity magnitude, local directional variations, and rotational details), and satisfy physical constraints (incompressibility, boundary conditions), all within a single coupled procedure. Consequently, the monolithic mapping substantially increases the learning difficulty.

Physical guidance insufficiency Two-stage generation frameworks introduce an explicit intermediate condition to decompose sketch-guided flow field generation, which can improve interpretability and reduce the complexity of each sub generation task. However, existing intermediate conditions are derived and represented primarily as visual-based information such as ridge-like structures or scalar maps extracted from the flow. These information are not explicitly encode motion-level information such as velocity magnitude or rotational information. The visual-based condition may provide insufficient control over the generated dynamics. In addition, ensuring physical consistency remains challenging. Due to diffusion-based generators typically operate in a latent space, adopting hard physical constraints directly on latent features is non-trivial. Without explicit motion condition constraint, the generated flow motion may violate physical constraints, leading to artifacts such as mismatched transport behavior. Addressing these issues requires intermediate representations that provide motion-related information.

1.4 Proposed Framework

This dissertation adopts hand-drawn sketch as input reference of latent diffusion model, generates incompressible smoke flow field that consistent with sketch structure and motion direction. In addition, the implicit geometric structure and motion information extracted from the smoke flow field is used as control condition in training process, improve the consistency of generated flow field.

1.4.1 Research Stages

The proposed smoke generation system is divided into the following stages as figure1.2 shows. The first three stages primarily focus on generating the smoke velocity field. Each subsequent stage builds upon the previous stage, continuously improving the accuracy of the generated velocity field. The generated velocity field is invoked into existing smoke simulators to drive the

smoke movement, thus generate the target smoke effect. In the final stage, the first three stages are integrated to construct a video diffusion model for smoke video generation from hand-drawn sketches. While maintaining the consistency of the flow field motion, smoke motion control in different visual styles is successfully achieved.

1. A one-stage sketch-guided velocity field generation framework based on latent diffusion model is proposed in the first stage. This framework enables unprofessional users to generate velocity fields consistent with hand-drawn sketch shapes. The stable training ability and generalization ability of latent diffusion model improve the alignment accuracy between semantic sketch and velocity field.
2. Second, based on the one-stage model, a two-stage sketch-guided smoke illustration generation framework is introduced. The Lagrangian Coherent Structure (LCS) region is introduced as intermediate representation between sketch and velocity field. The mapping between sketch and velocity is decomposed into simpler sub mappings to decrease the training complexity. The LCS region extracted from velocity field represent fluid region. The LCS provides a strong region control ability for velocity field generation, which enhances the geometrical consistency of the synthesized flow fields. The generated velocity field can be assembled into existing smoke simulators for achieving a semantic-driven smoke illustration synthesis.
3. Third, based on the two-stage model, the intermediate representation is replaced from LCS into stream function. Although the LCS region is a spatially-constrained information extracted from the velocity field, the characterization appears as a mask-like regional control condition, exhibiting a degree of ambiguity. The stream function extracted from the vorticity field explicitly indicates the rotational motion information of the velocity field. The stream function inherently possesses divergence-free property, satisfied with the incompressibility of the generated velocity field.
4. In the final stage, a sketch-guided smoke video generation framework is proposed on the basis of previous stages. Text prompt is given for specifying the generated video as a smoke pattern. A complete sketch is drawn for generating the stream function as smoke motion constraint. The sketch can be locally erased to generate rendered density field at different time step as the first frame condition of video generation. In addition, for the first frame smoke images that differ from the style of the dataset, motion features stably guide smoke motion in video generation. The style-transferred smoke images are generated using a

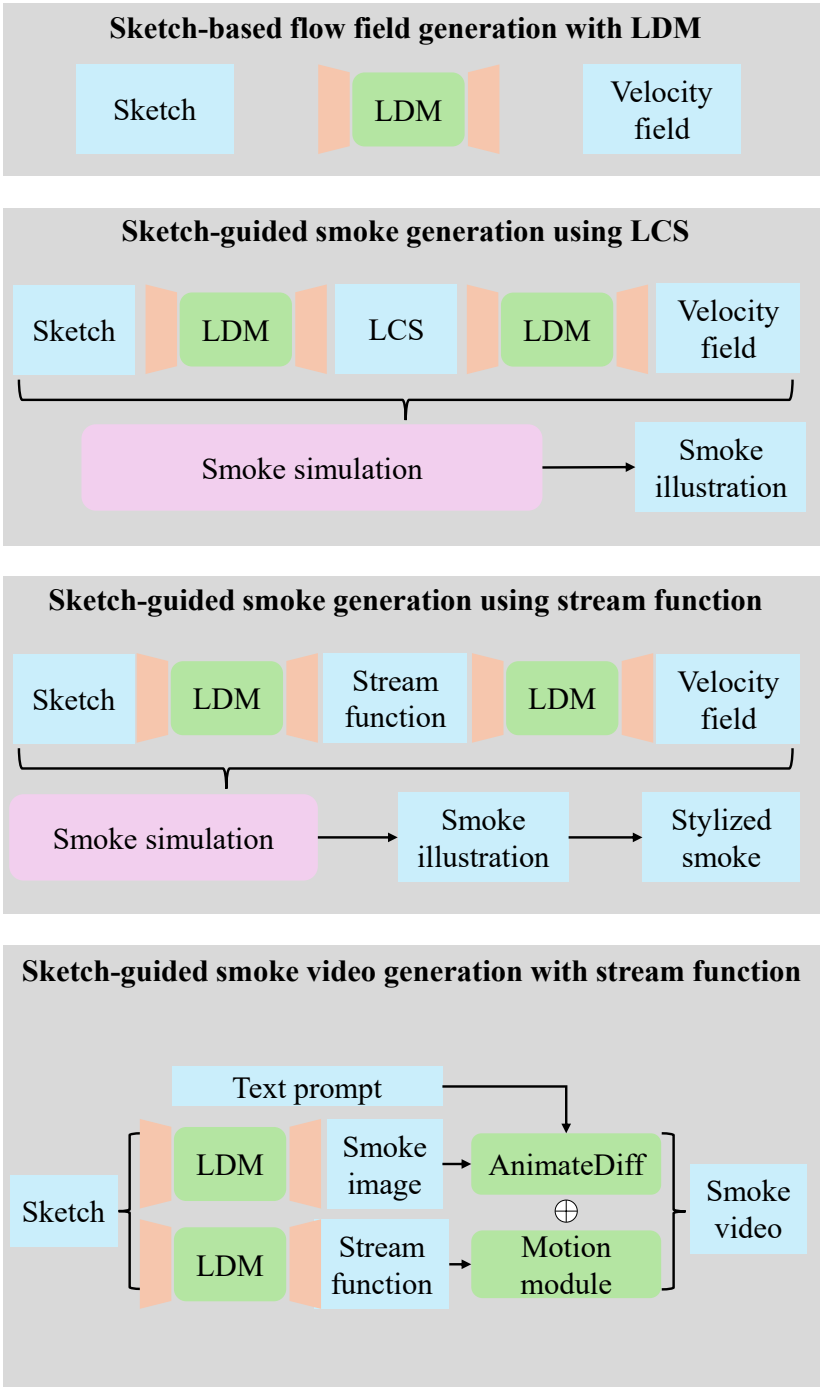

Figure 1.2: The overview of this dissertation. The whole dissertation has totally 4 stages.

pre-trained I2I large model. The proposed smoke video generation model can directly generate stylized smoke motion videos without integrating with simulators and renderers.

This dissertation contributes a physics-aware, sketch-driven generative pipeline for smoke effect creation. The one-stage LDM-based sketch-to-velocity framework proves that improves training stability and generalization relative to cGAN baselines. To address the ill-posedness and limited interpretability of monolithic mappings, a two-stage formulation that decomposes the generation process via explicit intermediate physical variables is furtherly introduced. In particular, this dissertation studies LCS- and stream-function-based intermediate representations: LCS provides strong region-level geometric control, while the stream function offers motion-aware guidance by explicitly encoding rotational structures and supporting divergence-free flows. Building on these components, a sketch-guided latent video diffusion model is proposed that combines text conditioning, first-frame constraints, and stream-function-based motion control to generate stylized smoke videos directly. The proposed framework provides a practical foundation for intent-driven smoke design, with potential impact on stylized content creation, interactive graphics tools, and physically informed video generation.

1.4.2 Core idea

These four research stages are integrated around a core idea: bridge the semantic gap between hand-drawn sketches and flow fields with diffusion model, thereby simplifying the creation process of smoke effects. These four stages form a progressively layered research program.

1. The first stage employs a diffusion model to learn the mapping from streamline sketches to velocity fields, thereby validating the capacity of diffusion model to stably learning characteristics of flow fields. This establishes a foundation for introducing intermediate physical representations into the mapping.
2. Building on the first stage, the second stage further decomposes the mapping from sketch to velocity field by introducing region features extracted from velocity fields as intermediate condition. By explicitly incorporating the spatial geometry characteristics of the velocity field as control conditions, the interpretability and controllability of diffusion model is proved.
3. Building on the second stage, the third stage introduces the motion information extracted from velocity field as intermediate condition of two-stage generation model. The extracted motion information

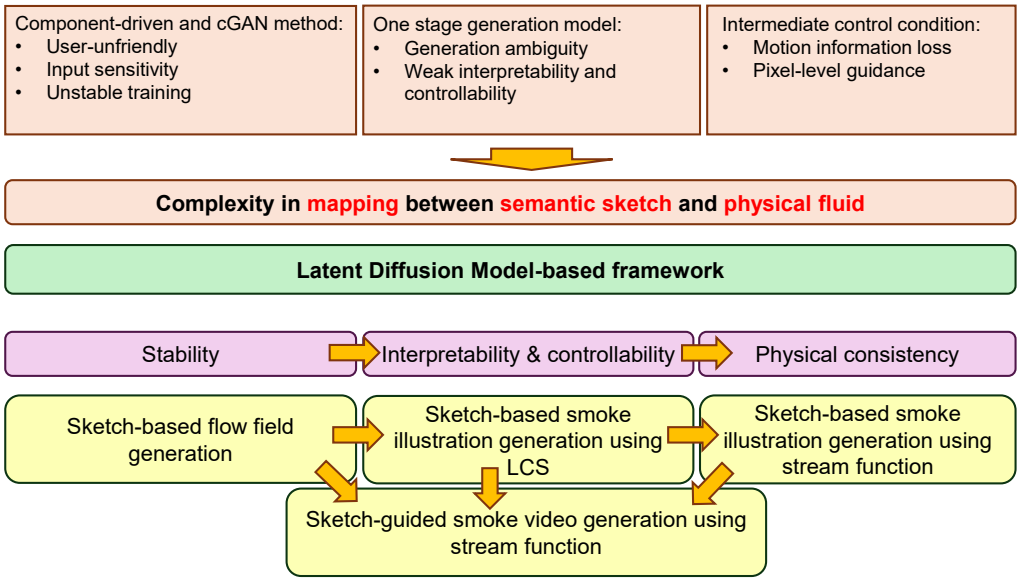

Figure 1.3: The interrelationship between stages in this dissertation.

furtherly aligns the geometrical and physical consistency of synthesized velocity field.

4. The fourth stage extends the generation target from single velocity field to smoke density field sequences building on previous stages. In this stage, the stream function serves as an intermediate representation between sketches and smoke density field sequences, and simultaneously controls the smoke motion information during video generation process. This validates the capability of the diffusion model to learn the spatiotemporal evolution of flow fields.

1.5 Dissertation Organization

Chapter 1 first explores the development and interrelationships of smoke effect creation, sketch-based semantic expression, and generative models. Addressing three challenges in current smoke effect creation: the semantic gap in generation control, the sparse-to-complex mapping complexity, and the physical guidance insufficiency. This dissertation proposes a sketch-guided smoke illustration and video generation system based on diffusion model. Physics-informed flow field motion control conditions are integrated into the generation process, a balance is achieved between semantic hand-

drawn sketch input and the physical consistency of the output flow field.

Chapter 2 reviews research on fluid simulation, fluid control, generative models, and sketch-based generation. Section 1 briefly introduces the development and classification of fluid simulation. Section 2 introduces related research on fluid motion control based on reference shapes, semantic sketches, and guiding fields. Section 3 introduces the development of generative models and their related research on fluid data synthesis. Section 4 introduces related research on sketch-based target image and video generation.

Chapter 3 introduces the foundational knowledge underpinning this dissertation. Firstly, key concepts in grid-based fluid simulation and physical knowledge include finite time Lyapunov exponent, Lagrangian coherent structure, Helmholtz-Hodge decomposition and stream function is introduced. Then the essentials of the auto-encoder and latent diffusion model is introduced, providing the theoretical basis for the subsequent chapters.

Chapter 4 introduces a one-stage sketch-guided velocity field generation framework using latent diffusion model. This framework adopts hand-drawn sketches as the input to synthesize the corresponding velocity field, thereby simplifying the connection gap between semantic sketch input and physics-based target velocity field.

Chapter 5 introduces a two-stage sketch guided smoke illustration generation framework based on LDM. The mapping between sketch and velocity field is decomposed into two sub-mappings for simplifying the generation process. The LCS is extracted from velocity field as an intermediate representation. The proposed two-stage framework enhances interpretability of velocity field generation process and achieves accurate flow region constraint. The generated velocity fields can be seamlessly integrated into existing simulation tools, reducing the complexity of fluid creation and improving creative flexibility.

Chapter 6 extends the two-stage approach by replacing LCS with the stream function as the intermediate representation. Stream function integrates a more precised geometric flow structure and rotational motion features than LCS region, yielding velocity fields with improved rotational structure and higher physical consistency. In addition, an I2I-based generation workflow is introduced for smoke illustration style transfer.

Chapter 7 intorduces a sketch-guided smoke video generation framework based on latent video diffusion model using motion control guidance from stream function. Users specify the first frame via sketches and define the video generation target as smoke through text prompt. The stream function generates from sketch provides motion control features for guiding the smoke motion throughout the generated video. Compared with prior video generation approaches, the proposed framework offers stronger motion coherence

and controllability. Furthermore, leveraging the expressive capacity of large generative models, style transfer can be applied to convert smoke into various artistic styles while preserving motion control guidance.

The research content in Chapters 4, 5, and 6 has been published [9–11].

Chapter 8 summarizes the primary contributions of this dissertation, discuss its limitations, and outline main directions for future research.

Chapter 2

Related works

This chapter introduces prior research relevant to this dissertation, which covers fluid simulation, fluid control, generative models, and sketch-based control. Furthermore, this chapter presents research on the integration of fluid control with sketching, fluid control with generative models, and sketching with generative models.

2.1 Fluid Simulation

Numerical simulation of fluid is one of the earliest and most mature research areas in computer graphics. The fluid simulation theoretical foundation is based on the Navier–Stokes equations. By discretely solving the three processes of convection, diffusion, and pressure projection, a numerical approximation of fluid motion is achieved. Beginning with early research on stabilizing physical solutions [12], the computer graphics community first addressed the fundamental issue of visually plausible fluid generation. Fedkiw et al. proposed a graphics-oriented smoke simulation framework [13]. Based on the incompressible Navier–Stokes equations, the proposed simulation process employed semi-Lagrangian convection and pressure projection to ensure numerical stability. Buoyancy and vorticity constraints were introduced to recover small-scale structure on coarse meshes, resulting in a realistic smoke appearance at a low cost. This framework had a profound impact on subsequent turbulence detail modeling and volume rendering pipelines. Figure2.1 shows the example of visualized 2D smoke and rendered 3D smoke using Mitsuba 3 [14].

From the viewpoint of reference frames and discretized kinematics, numerical methods for fluid simulation can be broadly classified into three categories as figure 2.2 shows: grid-based, particle-based, and hybrid schemes. Grid-based methods adopt an Eulerian perspective, describing the spatiotemporal evolution of field quantities on a fixed mesh; physical variables may be stored at cell centers, cell faces, or cell edges. Advection is typically treated with semi-Lagrangian schemes, while diffusion and pressure

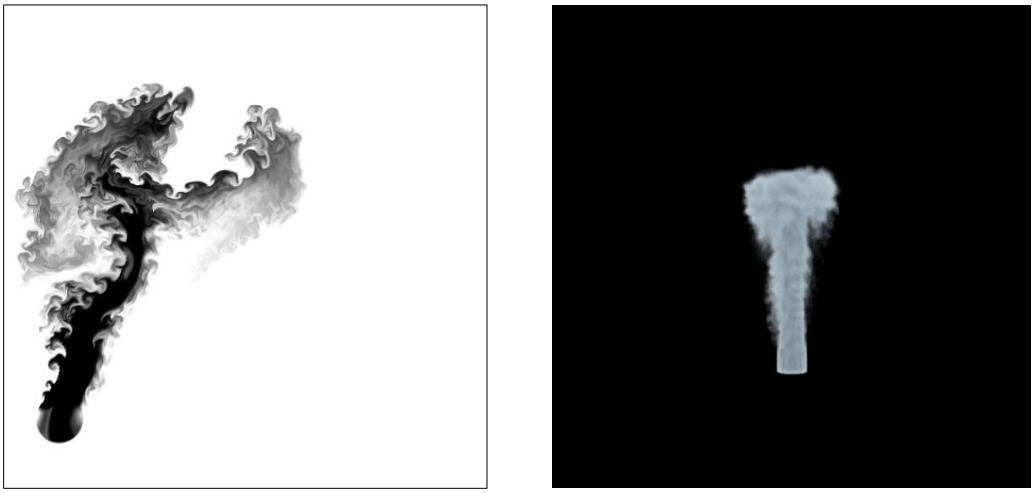

Figure 2.1: The visualization of smoke simulation. The left image is the rendered 2D smoke density field driven by perlin noise in 512×512 resolution. The density is mapped into a grey-scale image. The right image is rendered 3D smoke density field in $256 \times 256 \times 256$.

projection are solved on the grid via Poisson equations. These methods are robust and handle volume preservation and boundary conditions well, but the numerical dissipation and are less flexible when dealing with free surfaces and large deformations are still the significant challenges exists in grid-based simulation. Particle-based methods [15–17], in contrast, follow a Lagrangian perspective, representing the fluid as discrete particles that carry physical quantities and move with the flow. Particle-based methods naturally accommodate free surfaces, fracture, and topological changes, and can preserve fine-scale details sharply. However, particle-based methods

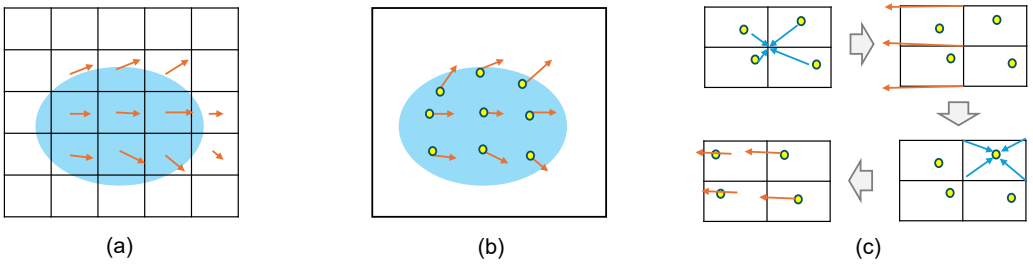

Figure 2.2: The introduction of smoke simulation schemes. Image (a) represents grid-based scheme; image (b) represents particle-based scheme; image (c) represents hybrid scheme.

incur higher computational costs in pressure solving, stability control, and neighborhood search. Hybrid methods [18–20] combine the strengths of both views: particles are used to carry momentum, mass, and local spatio-temporal information of fluid elements, while an auxiliary grid is employed to solve pressure projection and body forces. Fluid quantities are exchanged between particles and grid, leveraging the complementary advantages of particle-based and grid-based formulations.

Since grid-based fluid simulations store physical quantities on regular grids, the extracted fields can be directly represented as dense tensors that align well with the data formats commonly used to train neural networks. The grid-based simulation is primarily used in this dissertation.

2.2 Fluid Control

The evolution of a fluid is completely determined automatically by the system of equations, which is difficult for users to manipulate the fluid shape or trajectory without violating physical consistency. As computer graphics gradually shifts from "physical simulation" to "visual creation," achieving controllable fluids while maintaining physical plausibility has become a core research topic.

One line of research in fluid control focuses on driving smoke toward a target shape by applying appropriately designed external forces. Given a prescribed target configuration, the simulation is guided via optimization or control forces—to deform the input smoke so that its motion evolves toward the desired shape. Treuille et al. [21] proposed a introduced keyframe-driven control in animation scenarios, optimizing external forces with continuous quasi-Newton optimization to guide fluids toward a specified keyframe or shape in time. Mcnamara et al. proposed a keyframe-guided adjoint fluid animation method [22]. The adjoint method is subsequently utilized to improve the efficiency and accuracy of gradient solutions, making high resolution smoke control feasible. In addition, the control is extended onto level-set liquid simulation from smoke simulation. Fattal et al. proposed a target driven 2D smoke animation method [23]. The external force is regarded as an explicit function instead of optimization process, which decreases the simulation computational consumption. Shi et al. proposed a shape driven smoke animation method [24]. The smoke area and object are represented as implicit function. A shape transformation is conducted from the initial smoke area to the target object. An shape matching optimization process is adopted for constraining the velocity field in simulation process. The smoke area is movable following the referenced object motion. Zhang

et al. proposed a skeleton guided fluid simulation method [25]. The fluid shape is initialized as the shape of given skeleton. The skeleton motion drives the fluid motion via an optimization that enforces velocity constraint. Following a solid-like liquid motion strategy, the fluid tracks the skeleton at a coarse scale, while fine-scale fluid motion behaviors are generated by a standard liquid solver. Yang et al. controls the smoke animation towards the given target shape through the signed distance field [26]. Three control forces are provided for deforming the smoke into target shape, including path control force, boundary control force, and shape control force. Path control constrains the motion follows the given path, boundary control constrains the smoke movement region, shape control drives the deformation. Chen et al. proposed a Lagrangian-eigenfunction-driven method [27] to deform an object toward the target image in the form of fluid-like motion. The control process is accelerated via analytical gradient expressions. The proposed method supports real-time fluid simulation and editing.

With the growing demand for interactive design, some research has begun to focus on artificially control the flow movement through path constraints. To address these challenges, recent efforts have explored integrating user input with semantically oriented fluid control signals, such as using hand-drawn paths, pen stroke directions, or symbolic markers as flow guidance information. This offers a new direction for human-machine collaborative fluid design. However, existing work is largely limited to geometric or velocity-level control, lacking a unified framework that transitions from high-level semantics such as hand-drawn sketch to physical flow field. Kim et al. proposed a path-based smoke control method [28]. This method allows animators to directly provide a 3D curve, from which the system generates a target velocity field (with the ability to overlay small-scale vortex details). Then, a linear closed-loop feedback is used to gradually pull the actual simulated velocity field back to the target velocity field, thus allowing the smoke to move along a specified path while maintaining a natural turbulent appearance.

In addition to being driven by lines, the flow field can also be edited by integrating components with different fluid motion modes to achieve fluid control. Angelidis et al. proposed a vortex-based smoke simulation and control method [29] for control the smoke field. In 2016, the energy brushes [3] was proposed. This research presented a user interface for achieving fluid motion effects through brushes. The study defined different brush types to achieve three motion effects: vortex, smoke, and waves. However, this research neglected the external forces of the global field, the possibility of mutual cancellation between components, and did not consider the composability between components.

To improve interactivity, subsequent research has proposed the concept of a guiding field. This involves incorporating a reference velocity field or flow direction into the solution process, forcing the simulation results to follow the reference field pattern. Bridson et al. calculates the stream function in 2D fluid simulation and vector potential in 3D fluid simulation for fluid control [30]. Noise and potential functions are added, the detailed motion and boundary conditions of the velocity field can be controlled, which simplifies the calculation while satisfying the divergence-free condition. Nielsen et al. proposed a multi-grid-guided smoke simulation control method [31]. Subsequently, research expanded to reference fields and shape-guided simulation. Nielsen et al. also proposed a variational-based fluid guidance method [32]. By introducing a low-resolution reference field into the energy functional, high-resolution smoke is enabled to follow the target flow while preserving detail. Further, Nielsen et al. further proposed the Guide Shapes framework [33], which directly maps geometric shapes into guiding conditions, allowing the fluid to visually conform to the target contour without disrupting the internal fluid dynamic structure. Yuan et al. controls the fluid deformation with Lagrangian coherent structure region [34] that extracted from finite time Lyapunov exponent. Sato et al. proposed adding local turbulent details to the smoke by globally constraining the smoke field with a low-resolution field and then performing the constraint on a high-resolution field [35]. Sato et al. proposed a vector potential deformation method for driving the smoke deformation [36, 37]. Sato et al. proposed a method [38] to control the deformation of the smoke by performing vision-based deformation operations on the flow field in the stream function domain and then remapping the stream function back to the velocity field. Processing in the stream function domain naturally satisfies the condition of flow field incompressibility, making the process more robust than processing in the velocity domain. Forootaninia et al. proposed a fluid guidance method [39] with the constraints from frequency domain, which achieves efficient and stable target tracking by constraining flow patterns in frequency space. Predefined guidance field is widely used in fluid control, relies on manually designed or pre-calculated guiding field, which is user-unfriendly to unprofessional users. These research are also sensitive to the user input, an abnormal input causes the generation process unstable.

2.3 Generative Model

For a long time, machine learning research has been dominated by discriminative models, whose main task is to learn conditional probability distributions

or decision boundaries, aiming to perform classification based on given inputs. These methods typically do not directly provide new samples sampled from the model. However, learning discrimination is insufficient to support many critical tasks. In scenarios such as engineering simulation, content creation and scientific design, synthesize new samples is also required. Meanwhile, traditional generative frameworks such as Gaussian Mixture Models (GMMs) [40] exist, but limited by sampling inference costs and expressive power, traditional frameworks struggle to process high-dimensional, complex data such as images end-to-end. The challenges mentioned above extend the research paradigm from classifying data to generating data.

Autoencoder (AE) [41] has been systematically used for end-to-end reconstruction and representation learning: the encoder compresses high-dimensional observations into a low-dimensional latent representation, while the decoder recovers the input from the latent representation. In 2013–2014, Kingma et al. proposed variational auto-encoder (VAE) [42] for learning latent features of data. The VAE maps input data to a probability distribution and learns the probability distribution characteristics of the input data. This method can not only perform feature learning but also sample latent variables from a Gaussian distribution to generate data samples. Based on this, new data can be obtained by changing the Gaussian distribution. In recent years, with the development of GAN and diffusion model, autoencoders have mainly played a role in compressing and decompressing data during the generation process, mapping high-resolution images to the latent space for learning, ensuring generation quality while reducing computational consumption.

A brief introduction of generative models is given in figure2.3.

2.3.1 Generative Adversarial Network

In 2014, Goodfellow et al. [43] introduced Generative Adversarial Networks (GANs), establishing the adversarial generative paradigm through a zero-sum game between a generator and a discriminator. By iteratively improving the generator under the pressure of the discriminator, GANs learn to produce samples that are indistinguishable from real data, marking the beginning of the deep generative modeling era. Generative models can be broadly categorized along two axes. From a control perspective, generative models are divided into unconditional and conditional models. From a representation perspective, generative models are divided into models that learn directly in the data space and those that operate in an implicit latent feature space. Conditional GANs (cGANs) [44] extend GANs by introducing auxiliary control conditions so that the model learns the data distribution conditioned on given inputs, thereby enabling controllable generation. Early cGANs

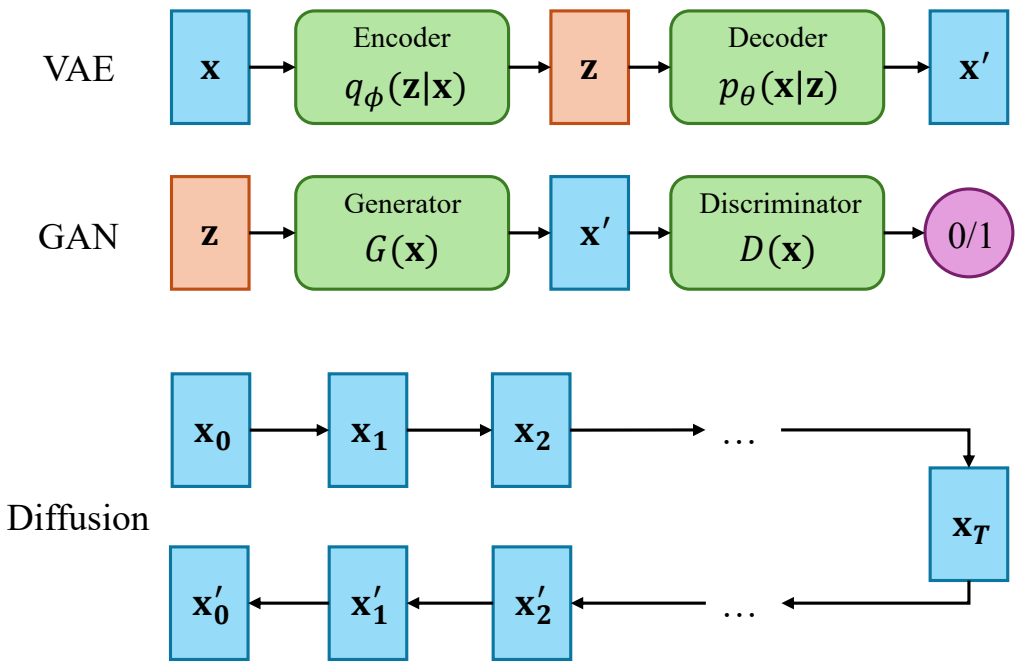

Figure 2.3: An overview of typical kinds of generative model. The first row is the VAE generation process introduction. The second row is the GAN generation process introduction. The third row is the diffusion model generation process introduction. x and x_0 represents the original data, x' and $x_0{}'$ represents the reconstructed data,x_T represents the features add noise for T times, z represents the latent feature of x and x_0.

injected conditions in two main ways: (i) concatenating the condition with the input at the network entrance, or (ii) feeding condition features into intermediate layers of the generator to modulate the feature maps. Compared with the original GAN, cGANs offer improved consistency and controllability of the generated outputs. In 2017, Isola et al. [4] proposed an image-to-image generation framework, marking another milestone for GAN networks. Based on cGAN, P2p uses uNet as the generator and patchGAN as the discriminator, jointly optimizing the loss and adversarial loss to improve the geometric consistency and texture details of the reconstruction.

Inspired by the end-to-end generation strategy, data-driven fluid field analysis and synthesis methods have been widely proposed. Xie et al. [45] applies conditional GANs to fluid super-resolution, achieving, for the first time, the generation of temporally consistent four-dimensional (3D volume $\times$ time) flow field details using a network. The core approach involves adding

a temporal discriminator in addition to a conventional spatial discriminator, adversarially constraining temporal coherence using adjacent frames. The generator can conditionally input low-resolution density, velocity/vorticity, etc., to synthesize high-resolution turbulent textures. Physically aware data augmentation is also introduced during training, allowing inference to produce continuous details in real-time with only a single step of low-resolution input, balancing efficiency and controllability. Hu et al. [5] proposed a cGAN-based velocity field generation framework for fluid motion control and design by assembling with fluid simulation tools. A small-scale sketch-velocity dataset is built, where the sketch data is drawn artificially. Wei et al. [46] uses cGAN to address temporal inconsistencies between adjacent frames after smoke super-resolution: First, a generator with residual blocks is used to recover two frames of high-resolution volume data from down-sampled data; then, these two frames and their corresponding velocity fields are fed into U-Net, and velocity-based warping and linear fusion are performed to generate multiple intermediate frames; and a slow-fusion temporal discriminator is designed to gradually constrain sequence consistency. Yan et al. [47] proposed a cGAN-based 3D fluid splash generation using 3D sketch drawn with VR tools. The system uses the trajectory and velocity of the curve as conditions to first generate an initial splash model with 3D flow information using a cGAN that trained with physically simulated data. Then the proposed system performs fine-tuning at small scales and provides VR interaction for rapid editing, resulting in realistic liquid splashes mesh within minutes. Chu et al. [48] uses a conditional GAN to directly infer the incompressible velocity field from the single-frame density field, and integrated physical parameters such as boundaries, buoyancy, kinetic energy and vorticity as control conditions into latent space to achieve a sensitive response to user modifications. A density-velocity reciprocal adversarial mechanism is designed to mitigate mode collapse and improve the generation of small-scale structures; a curl layer was used at the output to ensure dispersion.

2.3.2 Diffusion Model

The core concept of the diffusion model is a progressive generation process including a noise-adding and a noise-decreasing process. During the training process, groundtruth samples are gradually noisy until the noise is approximately isotropic; during the generation process, the network learns to denoise the data back to original data. Compared to adversarial methods, diffusion models offer more stable training and more comprehensive distribution coverage. Since 2020, diffusion models have rapidly become a mainstream paradigm for image and video generation. Denoising diffusion probabilistic

models (DDPM) [49] proposes a trainable and stable bidirectional markov-chain-based diffusion-denoising process: the forward process adds the Gaussian noise to the data; the reverse process, in which a parameterized trainable network predicts and removes the noise, gradually restoring the sample. The training objective is to ensure that the network produces high-quality denoised predictions at every noise level.

While retaining the DDPM training paradigm, denoising diffusion implicit models (DDIM) [50] redesigns the temporal and noise dependencies of the reverse process, transforming the original stochastic Markov chain into a deterministic or low-noise non-Markov trajectory. This significantly reduces the number of sampling steps while maintaining consistency with the DDPM training distribution.

Latent Diffusion Model (LDM) [51] encodes data to reduce its dimensionality to a more clearly structured, lower-dimensional latent space using an autoencoder, and then perform diffusion modeling and sampling within this latent space. This significantly reduces the computational cost at each step, addressing the high sampling cost issue of typical diffusion models. Furthermore, the latent space itself is more suitable for conditional fusion. Through the Cross-Attention [52] mechanism, input information from various modalities, along with control conditions such as text, images, masks, segmentation maps, and vector fields, can be injected into the denoising process, thereby achieving interpretable and adjustable control between global semantics, local geometry, and appearance details. In practice, LDM supports image-to-image generation, local image inpainting and redrawing, and efficient local model fine-tuning such as low-rank adaptation strategy(LoRA) [53]. The inference process can be combined with fast samplers such as DDIM and knowledge distillation for further acceleration.

ControlNet [54] is a conditional framework that introduces programmable structural constraints to the generation process without compromising the capabilities of the main diffusion model. The controlNet copys a parallel control branch from each downsampling and upsampling block of a pre-trained U-Net. This branch is trained using zero-initialized convolutions and locked weights. This allows the trainable control branch to learn to transform external conditions such as edge maps, depth maps or semantic segmentation mask into guiding features at the same scale as the backbone network. These features are then added to the backbone features at the corresponding layer with adjustable weights. Because the initialization is zero, the early stages of training are essentially hands-off. As training progresses, the control branch gradually learns to apply constraints where needed. This approach preserves the original style and detail of the base model while accurately incorporating geometric and structural priors.

Diffusion models are not only rapidly developing in image generation, but also gradually becoming a popular research method in data-driven physical data generation. Shu et al. proposed a physical informed diffusion model (PIDM) [55] for high-fidelity flow field reconstruction. During training, only high-resolution computational fluid dynamics (CFD) data is used; during inference, the inputs are low-resolution or sparse measurements, guiding sampling by incorporating known physical conditions of the partial differential equation (PDE) in denoising, thereby recovering fine-scale structures and improving robustness. The method is validated on two-dimensional turbulence, adapting to different input formats without retraining, and achieving reconstruction accuracy superior to traditional learning methods. Lienen et al. changed the 3D turbulence model from autoregressive time integration to generative modeling [56]. The manifold of the three-dimensional turbulent state is directly learned using a diffusion model, without the requirement for an initial field, thus avoiding error accumulation and small step size constraints.

Stable Video Diffusion (SVD) [57] is a video generation model based on a diffusion model. SVD extends traditional static image generation models to the realm of video generation, utilizing the diffusion process in latent space to generate each frame while ensuring smooth transitions between frames, thus avoiding the jumpy and disjointed effects common in traditional video generation methods. The core advantage of SVD is the ability to effectively combine image generation with temporal consistency, generating video content that conforms to textual descriptions and possesses dynamic performance. AnimateDiff [58] is a video generation method based on latent diffusion models. AnimateDiff introduces motion modules to enable the model to generate smooth dynamic content using motion field as guidance. This method extends the framework of static image generation to video generation, controlling motion information to generate consistent motion effects, resulting in coherent and natural movements between multiple generated frames. Cao et al. proposes to infuse video diffusion models with knowledge of underlying physical phenomena [59]. First, pre-training the model on physical phenomenon data to obtain visual embeddings carrying physical laws. Then the CLIP is invoked for aligning visual embeddings with pseudo-textual information in a quaternion latent space. This method enables fine-tuning existing video diffusion models efficiently. The generated video has a high-level physically consistency in both numerical simulations and real observations.

2.4 Sketch-Based Content Generation and Design

In computer graphics, sketching target objects is a classic research direction. Sketch-based generation has evolved from early methods of drawing simple geometric models and industrial design drawings primarily using straight lines, to the automatic recognition of human hand-drawn sketches and the automatic generation of complex target images. In early research, Zeleznik et al. proposed a method that generated scenes from hand-drawn sketches [60], the geometric structures are defined with different types of sketches. Igarashi et al. proposed a sketch-based system to generate and edit 3D objects [61]. Black contours generates 3D objects; red contours add and erase parts on the objects, and blue strokes deform the objects. Karpenko et al. extends the input to complex visible contours containing cusps and T-junctions, enabling robust inference of 3D freeform surface shapes from complex line drawings [62]. There are also 3D curve-guided 3D surface optimization and bidirectional editing methods [63, 64] for art and design.

Similarly, sketches are not only used to depict the appearance and shape of an object, but also to represent the trajectory or trend of motion. Thorne et al. is the first proposed the idea of sketch-driven motion control [65]. Users draws strokes such as lines, arcs, and loops on a pre-drawn character. The system interprets these strokes as parametric actions (walking, jumping, waving, etc.) and maps the spatial-temporal distribution of the strokes to the amplitude and rhythm of the action.

In recent years, with the development of generative models, linking sketches representing semantic information with various visual information through this end-to-end generation method has gradually become a new paradigm for generation. Chen et al. introduced a GAN-based network [66], aligning sketches and target images in the pixel domain to form data pairs. Sketchycoco [67] generated scene image from freehand scene sketches. The scene information and input sketches are aligned by mapping sketches to the edge map attribute space. Liu et al. proposed a GAN-based network that synthesizing stylized artistic images from sketches [68]. The sketches are separated into content and style mappings, achieving style transfer on the basis of generation. He et al. generated 3D mesh normal maps from single-view sketch inputs using GAN [69]. Du et al. proposed a GAN-based network for 3D point cloud generation from the input sketches [70]. Sketch2Cloth [71] generated 3D clothes mesh from sketches under the guidance of unsigned distance field (UDF). DualShape [72] generated 3D car and tire signed distance field (SDF) mesh from 2D sketches. The input sketches matches the

pattern stored in a given car and tire shape database. Peng et al. proposed a free-hand sketch-guided global-local human face generation framework using latent diffusion model [73]. Human facial organs are extracted and labeled from human face image as training data. Users draw faces by setting specific facial parts to gain a more precised local detail generation result.

Zhang et al. proposed a framework for scene generation and editing based on sketches [74] , which allows for strong control over the position of the generated result by specifying positional relationships for different objects in the sketches. Jin et al. proposed a method for motion control of complex landscape videos such as rivers and oceans [75], which uses scene sketches and streamlines to control the flow direction. These studies on sketch generation analyze the integration of different data modalities with sketches, adjusting the generative model structure to fit the sketch input.

Chapter 3

Preliminaries

In this chapter, the fundamental theories supporting this dissertation is introduced. First, the grid-based 2D inviscid fluid simulation method used in this dissertation is introduced. Second, this chapter introduces the intermediate representations used to connect the sketch input and the target field output, including the LCS and stream function, along with the prerequisite theoretical knowledge for both: finite time Lyapunov exponent and Helmholtz-Hodge decomposition. Finally, the LDM is introduced, including autoencoders as a prior knowledge.

3.1 2D Grid-Based Fluid Simulation

This work tackles the time-evolving flow within a two-dimensional spatial region. At each location and time, there is a two-dimensional velocity vector (with horizontal and vertical components), a scalar density field and a scalar pressure value. For simplicity and compatibility with computer graphics applications, the fluid is assumed as incompressible, with a constant density greater than zero, and negligible viscosity (in other words, treating the fluid as an inviscid fluid).

Let $\Omega \subset \mathbb{R}^2$ be the spatial domain and $t \in [0, T]$ the time interval. Denote the velocity and pressure by

$$\mathbf{u}(\boldsymbol{x}, t) = \big(u_x, u_y\big)^\top, \qquad p(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \Omega. \tag{3.1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla p + \boldsymbol{f}_e \tag{3.2}$$

where $\boldsymbol{f}_e$ denotes external body forces (e.g., buoyancy or numerically motivated forces).

Under the assumptions of constant density $\rho > 0$, and zero viscosity ($\nu = 0$), the flow follows the incompressible inviscid Navier-Stokes equations, which is also named as Euler equation. The velocity update is determined by three components: advection, the pressure gradient, and external forces.

These external forces can be gravity, buoyancy, vorticity confinement force or given user control. The core idea can be more intuitively expressed using the concept of material derivatives: along the trajectory of a fluid particle moving with the fluid, its acceleration is equal to the thrust due to the pressure plus the external forces.

Divergence free condition Use the material derivative $D/Dt = \partial_t + \boldsymbol{u}\cdot\nabla$, (3.2) is equivalently

$$\frac{D\boldsymbol{u}}{Dt} = -\frac{1}{\rho}\nabla p + \boldsymbol{f}_e, \qquad \nabla\cdot\boldsymbol{u} = 0. \tag{3.3}$$

Furthermore, incompressibility requires that the velocity field always satisfy the zero-divergence constraint, there is no local volume expansion or volume compression in the simulation field. The above definitions and constraints together constitute a continuous model of two-dimensional incompressible fluid.

Boundary condition A rectangular obstacle domain with solid walls is considered as $\partial\Omega$. The boundary condition is given as:

$$\boldsymbol{u}\cdot\boldsymbol{n} = 0 \quad \text{on } \partial\Omega, \tag{3.4}$$

where $\boldsymbol{n}$ is the outward normal. The pressure boundary condition follows naturally from the projection step. In a projection method, this corresponds to a Neumann boundary condition for the pressure that enforces $(\boldsymbol{u}^{n+1}\cdot\boldsymbol{n}) = 0$ on solid walls. The tangential component typically adopts a free-slip condition in the inviscid case, $\partial(\boldsymbol{u}\cdot\boldsymbol{t})/\partial n = 0$, where $\boldsymbol{t}$ represents unit tangent vector.

Advection The most common advection approach is based on semi-Lagrangian scheme by following [12, 76]. Given divergence-free $\boldsymbol{u}^n$ at time layer n, the semi-Lagrangian advection is performed:

$$\tilde{\boldsymbol{u}}(\boldsymbol{x}) \approx \boldsymbol{u}^n\big(\boldsymbol{x} - \Delta t\,\boldsymbol{u}^n(\boldsymbol{x})\big), \tag{3.5}$$

$$\boldsymbol{u}^* = \tilde{\boldsymbol{u}} + \Delta t\,\boldsymbol{f}_e^n. \tag{3.6}$$

The back-traced departure point can be integrated with Second-Order Runge-Kutta (RK2) and Fourth-Order Runge-Kutta method (RK4) [77] [78] for higher accuracy; the interpolation is typically bilinear. The RK4 is a

common approach for particle position updating. The equations are defined as follows:

$$\begin{aligned} x_{n+1} &= x_n + \frac{\Delta t}{6}(k_{1x} + 2k_{2x} + 2k_{3x} + k_{4x}) \\ y_{n+1} &= y_n + \frac{\Delta t}{6}(k_{1y} + 2k_{2y} + 2k_{3y} + k_{4y}) \end{aligned} \tag{3.7}$$

$$\begin{cases} k_{4x}, k_{4y} = f(x_n + \Delta t k_{3x}, y_n + \Delta t k_{3y}) \\ k_{3x}, k_{3y} = f(x_n + \frac{\Delta t}{2} k_{2x}, y_n + \frac{\Delta t}{2} k_{2y}) \\ k_{2x}, k_{2y} = f(x_n + \frac{\Delta t}{2} k_{1x}, y_n + \frac{\Delta t}{2} k_{1y}) \\ k_{1x}, k_{1y} = f(x_n, y_n) \end{cases} \tag{3.8}$$

where x_n, y_n are given as the positions of particles at current status n, x_{n+1}, y_{n+1} are the positions of particle at next status $n+1$. The time step is donated as Δt, with the slopes at the start point represented by k_{1x}, k_{1y}. Similarly, $k_{2x}, k_{2y}, k_{3x}, k_{3y}$ represent the slopes at the midpoints, while k_{4x}, k_{4y} correspond to the slopes at the endpoint. These four sets of slopes are calculated by the slope estimation function f. The function f specifically refers to the mapping from a given position to its corresponding velocity. If the position lies outside the boundaries of the velocity field, the calculation will be omitted. The calculation is determined by the given position x_n, y_n, and velocity information corresponding to the given position. Each grid in the 2D vector field requires calculation cost in a time complexity of $O(n^2)$.

Although semi-Lagrangian advection is formally unconditionally stable, accuracy and interpolation error motivate the set of Δt must not exceed a certain upper limit. Excessive forces or time steps can cause large back-tracing distances, amplify projection error, or induce nonphysical diffusion, manifesting as unnatural expansion or distortion of smoke.

Pressure projection The intermediate velocity $\boldsymbol{u}^*$ is generally not divergence-free. The irrotational component is removed via a pressure update:

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \frac{\Delta t}{\rho}\nabla p^{n+1}, \qquad \nabla \cdot \boldsymbol{u}^{n+1} = 0. \tag{3.9}$$

Taking divergence of (3.9) yields the pressure Poisson equation

$$\nabla^2 p^{n+1} = \frac{\rho}{\Delta t}\,\nabla \cdot \boldsymbol{u}^* \quad \text{in } \Omega, \tag{3.10}$$

with the Neumann-type boundary condition induced by impermeability,

$$\frac{\partial p^{n+1}}{\partial n} = \frac{\rho}{\Delta t}\left(\boldsymbol{u}^* - \boldsymbol{u}_b\right) \cdot \boldsymbol{n} \quad \text{on } \partial\Omega. \tag{3.11}$$

Solving (3.10)–(3.11) for p^{n+1} and substituting into (3.9) enforces incompressibility. Discrete divergence and gradient usually use four-point differences, yielding an approximately orthogonal projection at the discrete level. The Poisson equation (3.10) leads to a sparse symmetric positive definite linear system, efficiently solved by conjugate gradient method (CG) or preconditioned conjugate gradient method (PCG) [79] [80].

External Forces Typical external force $\boldsymbol{f}_e$ include buoyancy, $\boldsymbol{f}_b = \alpha(\rho_a - \rho_s)\,\boldsymbol{g}$ vorticity confinement force$\boldsymbol{f}_\omega = \epsilon\,(\boldsymbol{N} \times \boldsymbol{\omega})$ and user control forces.

Vorticity confinement [13] is a visually driven force used in grid-based fluid solvers to counteract the numerical dissipation introduced by stable advection schemes: small-scale rotational energy is injected near vortex cores without significantly altering the large-scale flow. In 2D flow with velocity $\mathbf{u} = (u, v)$, the (out-of-plane) scalar vorticity is

$$\omega \;=\; \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}, \tag{3.12}$$

and the magnitude is donated by $\Omega = |\omega|$. The unit vector pointing toward regions of stronger vorticity is defined as

$$\mathbf{N} \;=\; \frac{\nabla\Omega}{\|\nabla\Omega\| + \varepsilon_N},$$

where ε_N is a small regularization parameter to avoid division by zero. In 2D incompressible flow, the vorticity confinement force can be written directly in terms of the scalar vorticity ω and the normalized gradient of its magnitude. Let ω denote the out-of-plane vorticity and $\mathbf{N} = (N_x, N_y)$ be the unit vector in the direction of $\nabla|\omega|$. The confinement force is then given by

$$\mathbf{f}_{\text{vc}} \;=\; \varepsilon_{\text{vc}}\, h\,\big(N_y\,\omega,\; -N_x\,\omega\big), \tag{3.13}$$

where $h = \min(\Delta x, \Delta y)$ provides resolution scaling and ε_{vc} controls the confinement strength. The velocity updates follows:

$$\Delta\mathbf{u}_{\text{vc}} = \frac{\Delta t}{\rho}\,\mathbf{f}_{\text{vc}}. \tag{3.14}$$

The vorticity confinement force acts tangentially along vortex cores and perpendicular to the vorticity gradient, enhancing and sustaining visually prominent small-scale vortex structures, while the final divergence-free state is still enforced by the subsequent pressure projection.

3.2 Finite Time Lyapunov Exponent and Lagrangian Coherent Structure

Finite Time Lyapunov Exponent The finite-time Lyapunov exponent (FTLE) [81] [82] quantifies the maximal exponential rate at which nearby trajectories of a flow separate over a finite time interval. FTLE is an objective quantity that invariant to time-dependent motion, but sensitive to the length of the time interval and the starting time. Different settings yield different structure patterns. FTLE measures the maximum exponential separation rate of adjacent trajectories within a given time interval, used to characterize regions where material trajectories exhibit the strongest finite-time stretching in a flow field. For a velocity field $\mathbf{u}(\mathbf{x}, t)$ with flow trajectory map $\mathbf{F}_{t_0}^{t_0+T}$: $\mathbf{x} \mapsto \mathbf{x}(t_0 + T; t_0, \mathbf{x})$, let the deformation gradient be $\nabla \mathbf{F}_{t_0}^{t_0+T}(\mathbf{x})$, the FTLE field σ at $(t_0, \mathbf{x})$ over duration T is defined by

$$\sigma_{t_0}^{T}(\mathbf{x}) = \frac{1}{|T|} \ln\left(\sqrt{\lambda_{\max}(\Delta)}\right), \Delta = (\nabla \mathbf{F}_{\mathbf{t_0}}^{\mathbf{t_0}+\mathbf{T}}(\mathbf{x}))^{\top} \nabla \mathbf{F}_{\mathbf{t_0}}^{\mathbf{t_0}+\mathbf{T}}(\mathbf{x}) \tag{3.15}$$

where the Δ is also named as Cauchy-Green deformation tensor in a finite-time version, the flow trajectory map $\mathbf{F}_{t_0}^{t_0+T}$ represents where a fluid material point$\mathbf{x}$ at timet_0 moves after time T. The above equation gives the logarithmic stretch of the longest principal axis of the advected point. Forward ($T > 0$) and backward ($T < 0$) FTLE highlight repelling and attracting material structures, respectively. The ridge of a forward FTLE ($T > 0$) approximately indicates a repelling matter boundary, while the ridge of a backward FTLE ($T < 0$) approximately indicates an attracting matter boundary.

In the discretization process, a specified time step δt is given, and then use the Runge-Kutta method to track the positions of points in a given velocity field within a specified time interval. The updated position information is then saved. By discretizing the field using a uniform grid, the initial positions of points within the field are specified at the intersections of the grid lines. In computer data structures, this can be viewed as the indices of each value in an array equivalent to the field resolution.

Lagrangian Coherent Structure Intuitively, the Lagrangian Coherent Structure (LCS) are time-dependent material curves (in 2D) or surfaces (in 3D) that organize point inside the flow field transport. The LCS acts as separatrices that partition the flow into regions with qualitatively different trajectories. At time t, the LCS is defined as the ridges of the FTLE field $\sigma_t^T(\mathbf{x})$, i.e., smooth curves that follow FTLE gradient lines and realize a strict

transverse maximum. Forward-time ridges ($T > 0$) approximate repelling finite-time manifolds, while backward-time ridges ($T < 0$) approximate attracting ones. Because FTLE is computed from particle trajectories via the flow-map Jacobian Δ, integrated transport information is captured for each discretized point over $[t,\, t+T]$. Therefore the σ is computed from $\nabla\phi$ and extract ridges accordingly, and the resulting structures serve as the LCS.

At time t_0, a Lagrangian coherent structure (LCS) is defined as a ridge of the scalar field $\sigma_{t_0}^T$. More precisely, a smooth curve Γ is an FTLE ridge if for each point $\mathbf{x} \in \Gamma$ with unit normal $\mathbf{n}(\mathbf{x})$:

$$\begin{aligned} \mathbf{n}^\top \nabla \sigma_{t_0}^T(\mathbf{x}) &= 0, \\ \mathbf{n}^\top \left(\nabla^2 \sigma_{t_0}^T(\mathbf{x})\right) \mathbf{n} &< 0 \end{aligned} \tag{3.16}$$

The first equation represents stationary in the normal direction, the second equation represents strict transverse maximum. Equivalently, the ridge tangent is aligned with the local gradient lines of $\sigma_{t_0}^T$, and $\sigma_{t_0}^T$ attains a local maximum in the direction normal to Γ.

3.3 Helmholtz Hodge Decomposition

The Helmholtz decomposition states that for a vector field which is sufficiently "well-behaved" $\mathbf{f}$ (decaying fast enough in the whole space, or satisfying suitable boundary conditions on a bounded domain), the vector field can be represented as the sum of a gradient field and a divergence-free field, possibly plus a harmonic component determined by the domain topology:

$$\mathbf{f} = \nabla\phi + \nabla \times \mathbf{A} + \mathbf{h}. \tag{3.17}$$

Here $\nabla\phi$ is the irrotational part that carries divergence information such as source or sink; $\nabla \times \mathbf{A}$ is the solenoidal (which means divergence-free) part that carries vorticity information; and $\mathbf{h}$, which is called as harmonic field that carries velocity potential information, is both divergence-free and curl-free equivalently:

$$\Delta\mathbf{h} = \mathbf{0}. \tag{3.18}$$

A standard numerical construction proceeds via Poisson equations: first solve

$$\Delta\phi = \nabla \cdot \mathbf{f}, \tag{3.19}$$

to obtain ϕ and hence the irrotational component $\nabla\phi$. Then set $\mathbf{f}_{\text{sol}} = \mathbf{f} - \nabla\phi$ to get the divergence-free part. If the vector potential is needed, under the Coulomb gauge one further solves

$$-\Delta\mathbf{A} = \nabla \times \mathbf{f}. \tag{3.20}$$

In two dimension situation, the construction is particularly convenient: the divergence-free part can be written via a scalar stream function ψ as the rotated gradient $\nabla^{\perp}\psi = (\partial_y\psi, -\partial_x\psi)$, which automatically has zero divergence, and

$$\mathbf{f}_{\text{sol}} = \nabla \times \psi, \tag{3.21}$$

$$\Delta\psi = \omega = \partial_x v - \partial_y u, \tag{3.22}$$

hence

$$\mathbf{f} = \nabla\phi + \nabla \times \psi \ + \mathbf{h}. \tag{3.23}$$

3.4 Stream Function

In two-dimensional incompressible flow ($\nabla \cdot \mathbf{u} = 0$), the stream function is a single scalar field $\psi(x, y, t)$ that represents the entire velocity field, defined by

$$u = \frac{\partial\psi}{\partial y}, \qquad v = -\frac{\partial\psi}{\partial x}. \tag{3.24}$$

This definition guarantees that any velocity field field $\mathbf{u}$ reconstructed from ψ is divergence-free, the incompressibility is enforced without an additional pressure projection.

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = \frac{\partial^2\psi}{\partial x\partial y} - \frac{\partial^2\psi}{\partial y\partial x} = 0. \tag{3.25}$$

The equation for calculating stream function from vorticity field ω in an incompressible 2D flow is given as follows:

$$\omega = \nabla \times \mathbf{f} = -\nabla^2\psi, \tag{3.26}$$

The ψ is obtained by solving the Poisson equation. The velocity is recovered by

$$\mathbf{u} = \big(\psi_y, \ -\psi_x\big). \tag{3.27}$$

In a typical centered-grid discretization, given cell-centered vorticity, one solves the five-point Laplacian for ψ with a linear solver such as CG scheme while pinning ψ to boundary constants, then obtains velocities by centered differences,

$$u_{i,j} \approx \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2h}, \qquad v_{i,j} \approx -\frac{\psi_{i+1,j} - \psi_{i-1,j}}{2h}, \tag{3.28}$$

achieving (discretely) divergence-free flow. Compared with pressure projection, the stream function method uses single scalar field, keep the incompressibility for 2D problems. For smoke and interactive control in graphics,

the ψ-based approach is especially convenient in 2D, smooth, divergence-free velocity can be directly generated to advect density. Geometrically, the level sets of ψ are streamlines, the stream function aligns with the streamline sketch pattern.

3.5 Auto-Encoder

Auto-encoder (AE) is a self-supervised representation learning model that learns useful latent data feature by training a network to reconstruct the given inputs: an encoder f_θ compresses an input $\mathbf{x}$ into a latent vector $\mathbf{z}$, and a decoder g_ϕ reconstructs $\hat{\mathbf{x}}$ from $\mathbf{z}$; training minimizes a reconstruction loss such as Mean Absolute Error loss (MAE loss) [83] or Mean Squared Error loss (MSE loss) [84] with optional regularization (to prevent mere memorization.

Let $\mathbf{x} \in \mathbb{R}^d$ be an input and $\mathbf{z} \in \mathbb{R}^k$ a latent code with $k < d$.

Encoder

$$\mathbf{z} = f_\phi(\mathbf{x}) = \sigma\big(W_e\,\mathbf{x} + \mathbf{b}_e\big) \;\in\; \mathbb{R}^k, \tag{3.29}$$

Decoder

$$\widehat{\mathbf{x}} = g_\theta(\mathbf{z}) = \psi\big(W_d\,\mathbf{z} + \mathbf{b}_d\big) \;\in\; \mathbb{R}^d, \tag{3.30}$$

where $\phi = \{W_e, \mathbf{b}_e\}$, $\theta = \{W_d, \mathbf{b}_d\}$, and σ, ψ are nonlinearities such as activation function. For multi-layer AEs, f_ϕ and g_θ denote layer compositions.

The training objective of auto-encoder is given as follows.

$$\min_{\phi,\theta} \; \frac{1}{N} \sum_{n=1}^{N} \ell\Big(\mathbf{x}^{(n)},\; g_\theta\big(f_\phi(\mathbf{x}^{(n)})\big)\Big) \tag{3.31}$$

As for the reconstruction loss of the auto-encoder, the most common choices would be MSE loss, the equation is given as follows:

$$\ell_{\text{MSE}}(\mathbf{x}, \widehat{\mathbf{x}}) = \|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2, \tag{3.32}$$

$$\tag{3.33}$$

3.6 Latent Diffusion Model

A latent diffusion model(LDM) is a generative framework that migrates the diffusion process from high-dimensional pixel space to a compact and perceptually faithful latent space in order to reduce computation while

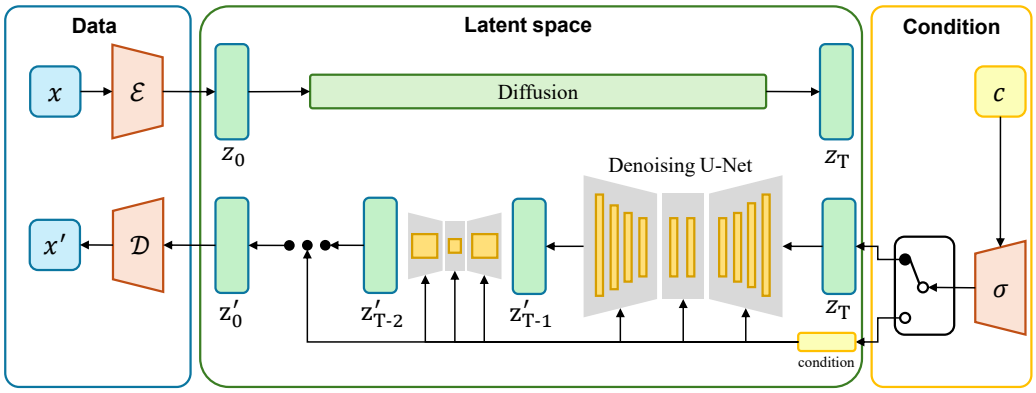

Figure 3.1: The structure of latent diffusion model. Training data are encoded into latent space for training using pretrained auto-encoders. The control condition is encoded into latent space and injected into the U-Net for constraining the denoising process.

preserving visual quality. The image of LDM structure is given as Figure3.1 shows. Concretely, a perceptual auto-encoder, which with encoder E and decoder D trained by a combination of pixel reconstruction and perceptual losses, compresses data x into a high-channel, low-resolution latent $z = E(x)$, after which the forward noising and reverse denoising of diffusion operate solely on z. The equations for a pretrained auto-encoder are given as follows:

$$z = E(x), \qquad \hat{x} = D(z) \tag{3.34}$$

The diffusion core remains standard: with a noise schedule $\{\beta_t\}$ (and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$), the forward process is $q(z_t \mid z_0) = \mathcal{N}\big(\sqrt{\bar{\alpha}_t}\, z_0, (1-\bar{\alpha}_t)I\big)$, while a U-Net parameterization predicts either the noise $\epsilon_\theta(z_t, t, c)$ or the velocity/clean target; training minimizes an MSE over random time steps and Gaussian noise, and inference starts from $z_T \sim \mathcal{N}(0, I)$ and iteratively denoise to z_0 using samplers or few-step samplers such as DDIM before decoding $\hat{x} = D(z_0)$ to pixels, where the $\hat{x}$ represents the reconstructed data decompressed by D.

Conditioning is injected via cross-attention inside multi-scale U-Net blocks, while U-Net features form Queries so that semantic/geometric constraints steer each denoising step. The equations of LDM is given as follows:

$$\epsilon_\theta = \epsilon_\theta(z_t, t, c), \qquad \hat{z}_0(z_t, t, c) = \frac{z_t - \sqrt{1-\bar{\alpha}_t}\,\epsilon_\theta}{\sqrt{\bar{\alpha}_t}} \tag{3.35}$$

$$\mathcal{L}_\epsilon = \mathbb{E}_{z_0,\epsilon,t}\big\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon,\ t,\ c)\big\|_2^2 \tag{3.36}$$

where c represents the control condition that injected into LDM. Compared with pixel-space diffusion, LDM is markedly more computationally efficient and scale more readily to high resolutions.

Chapter 4

Sketch-Based Flow Field Generation

This chapter further discusses about the convergence of flow field generation with deep learning for augmenting the efficiency of flow simulation [9]. Current approaches primarily utilize conditional generative adversarial networks(cGANs) to generate velocity fields guided by sketches. In contrast, the cGAN training process exhibits instability. A 2D velocity-field generation framework based on a latent diffusion model (LDM) is proposed in this study. In this formulation, a user-provided sketch serves as a condition that constrains the denoising process, thereby guiding the target 2D velocity field generation. The proposed method enables the synthesis of velocity fields whose spatial structures are consistent with the geometry implied by the input sketches. This research further evaluate the method against cGAN-based method, the results demonstrate improved robustness of the proposed framework.

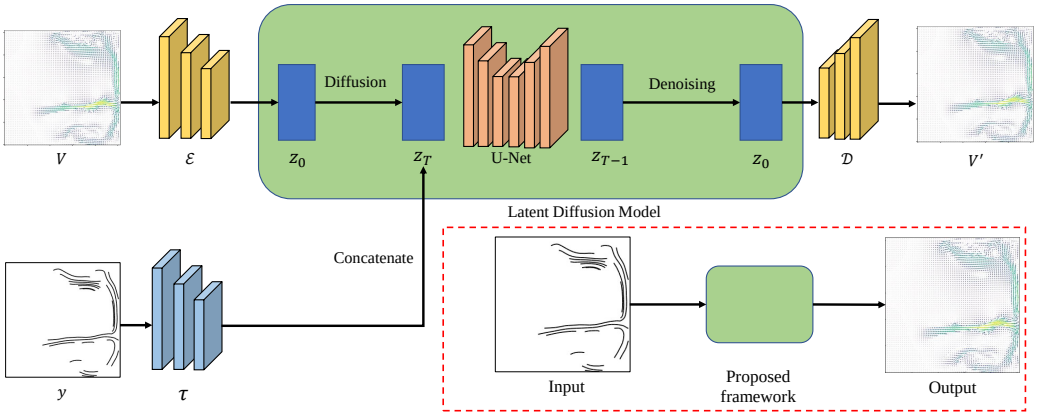

Figure 4.1: The introduction and framework of this study.

This study proposed an LDM-based sketch-guided 2D velocity field design framework. The overview of the fundamental concept is shown in Figure 4.1,

inside the red dotted lines. Inspired by Peng et al. [73], and the sketch data is configured as the input constraint condition, the 2D velocity field is generated within the proposed framework. The LDM effectively condenses intricate data into a more simplified feature map or feature vector, streamlining the analytical process and decreasing the computational demand while preserving the primary features of the initial input. Notably, LDM exhibits adaptability in terms of target generation constraining through diverse input [51]. A 2D velocity field and sketch dataset is constructed since no existing open-source dataset is tailored to this research. Additionally, two auto-encoders is formulated to compress velocity fields and sketches into corresponding feature maps and reconstruct data from given feature maps. The contributions of this research are given as:

- 1. The proposed LDM-based velocity field generation framework is the first attempt to apply LDM for velocity field generation design by inputting sketch strokes.
- 2. A comparison between the proposed framework and cGAN is conducted, which validated the stability of the proposed framework within the context of 2D velocity field generation tasks.

4.1 Proposed Method

In this section, an overview of the proposed framework is provided. The proposed framework is shown in Figure 4.1. This figure shows the complete generation flow from left to right. The top half is the training-generation backbone. Given a paired example sketch y and velocity field V, the velocity field autoencoder encoder $\mathcal{E}$ first compresses V into a latent feature Z_0. In the latent space diffusion model, z_0 is noised per time step to obtain z_T, which is then gradually denoised by a conditioned U-Net to z^0. Latent condition comes from y, the sketch encoder τ compresses the sketch into a latent feature. This is then concatenated with z_t at each step in the channel dimension, constraining the denoising process to the geometric intent of the sketch. After denoising, z^0 is restored to the velocity field V by the decoder $\mathcal{D}$, thus achieving sketch-guided velocity field synthesis in latent space.

Before training the proposed framework, the training data collection process will be introduced first. The training target is generating velocity field from sketch, 2D incompressible inviscid smoke simulation is conducted to extract velocity field data and paired streamlines. The streamline images are regraded as sketches for training. Then the implemented auto-encoder and LDM will be introduced.

4.1.1 Dataset construction

The velocity field data are extracted from 2D smoke simulation scenarios. The particles are moved through advection, influenced by velocity $\mathbf{u}$ and pressure p at time step t. Assuming $\mathbf{u}$ and p are initialized, the velocity field is updated by 2D incompressible inviscid Navier-Stokes equations as follows:

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u} \cdot \nabla \mathbf{u} - \frac{1}{\rho} \nabla p + \mathbf{b} \tag{4.1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{4.2}$$

Where $\mathbf{u}$ is the given velocity, and $\mathbf{b}$ is the buoyancy. The simulation operates inside rectangle obstacle domain with a solid boundary. The boundary condition is given as the following equation:

$$\mathbf{u} \cdot \mathbf{n} = 0 \tag{4.3}$$

where $\mathbf{n}$ is the normal vector of the boundary. The semi-Lagrangian scheme [12] is applied to update the fluid field.

The 2D smoke simulation process is given as Algorithm 4.1 shows.

The sketch data used in this study are composed of streamlines extracted from the underlying 2D velocity fields. A streamline is defined as an integral curve whose tangent direction coincides with the velocity vector in the field. Streamlines provide an intuitive representation of fluid motion and offer a concise yet effective description for characterizing complex velocity-field structures. The extraction operation is given as Algorithm 4.2 shows. To calculate the streamlines, the Fourth-Order Runge-Kutta method [77] is applied for its satisfaction in tracing the trajectories while maintaining low computational complexity with minimal parameters.

Samples of velocity field and sketch data are given as Figure 4.2 shows. The Matplotlib tools is used to visualize the velocity field. The direction of vector arrow corresponds to the velocity direction. The color represents the magnitude of the velocity. The color distributes linearly from 0 to 1, where blue color indicates lower speed, red color indicates higher speed.

4.1.2 Velocity field and Sketch Auto-Encoder

Auto-encoder is an important module in LDM. The encoding operation compresses the data into latent features, enabling the model to learn these features in the latent space. The decoding operation is responsible for restoring the latent features back to the original data. In this study, the velocity field auto-encoder and sketch auto-encoder are implemented. Here

Algorithm 4.1: 2D Incompressible Inviscid Smoke Simulation

Input: grid size (N_x, N_y); time step Δt; number of steps T;
Circle-shape smoke source S locates at given center $\mathbf{c}$ with radius r;
base emission velocity $\mathbf{u}_e$;

Output: 2D velocity field $\{\mathbf{u}^{(t)}\}_{t=0}^{T}$

Initialize velocity field $\mathbf{u} = (u_x, u_y)$ and density field ρ to zero;
Initialize auxiliary fields (previous velocity $\mathbf{u}_{\text{prev}}$, previous density ρ_{prev}, pressure p, divergence field div) to zero;
for $t = 0$ **to** $T - 1$ **do**
    foreach *grid cell* (i, j) *inside the* S **do**
        `// Add smoke source`
        $\rho_{ij} \leftarrow \rho_{ij} + S\,\Delta t$;
    end
    begin
        `// Velocity advection (semi-Lagrangian)`
        $\mathbf{u}_{\text{prev}} \leftarrow \mathbf{u}$;
        $u_x = \text{Advect}(u_x, u_x^{\text{prev}}, \mathbf{u}_{\text{prev}}, \Delta t) + f_x \Delta t$;
        $u_y = \text{Advect}(u_y, u_y^{\text{prev}}, \mathbf{u}_{\text{prev}}, \Delta t) + f_y \Delta t$;
        `// Projection`
        Compute divergence field $\text{div} = \nabla \cdot \mathbf{u}$;
        Solve Poisson equation $\nabla^2 p = \text{div}$ for pressure p;
        $\mathbf{u} \leftarrow \mathbf{u} - \nabla p$;
        Apply boundary conditions to $\mathbf{u}$;
    end
    begin
        `// Density advection`
        $\rho_{\text{prev}} \leftarrow \rho$;
        $\rho \leftarrow \text{Advect}(\rho, \rho_{\text{prev}}, \mathbf{u}, \Delta t)$;
    end
end

Algorithm 4.2: Streamline Extraction from 2D Velocity Field

Input : 2D velocity field $\mathbf{u}(x, y) = (u(x, y), v(x, y))$;
number of high-magnitude samples k;
integration step size Δt.

Output: A set of polylines $\mathcal{L}$ representing sparse streamlines.

For each $\mathbf{u}(x_i, y_j)$ at column position i and row position j, sample the velocity components $u_{ij} = u(x_i, y_j),\ v_{ij} = v(x_i, y_j)$.

Compute the speed magnitude $m_{ij} = \sqrt{u_{ij}^2 + v_{ij}^2}$.

Select high-magnitude samples

Flatten $\{m_{ij}\}$ into a 1D array and find threshold θ such that exactly k samples satisfy $m_{ij} \geq \theta$

Define a binary mask $M_{ij} = \begin{cases} 1, & m_{ij} \geq \theta, \\ 0, & \text{otherwise.} \end{cases}$

Let the set of seed points be $\mathcal{S} = \{(x_i, y_j) \mid M_{ij} = 1\}$.

Generate streamlines by RK4 integration

Initialize an empty set of polylines $\mathcal{L} \leftarrow \emptyset$.

foreach *seed point* $(x_0, y_0) \in \mathcal{S}$ **do**

    `// Forward integration`

    $\ell^+ \leftarrow \mathrm{RK4}(x_0, y_0, +1, \Delta t)$.

    `// Backward integration (optional)`

    $\ell^- \leftarrow \mathrm{RK4}(x_0, y_0, -1, \Delta t)$.

    Concatenate $\ell = \mathrm{reverse}(\ell^-) \cup \ell^+$, removing the duplicated seed point.

    Append ℓ to $\mathcal{L}$.

end

return $\mathcal{L}$

Algorithm 4.3: RK4 (Fourth-Order Runge-Kutta) in One Step

Input : Initial position $\mathbf{x}_n(x_n, y_n)$ at status n;
direction sign $\sigma \in \{+1, -1\}$ (forward/backward);
integration step size Δt;
velocity field $\mathbf{u}(x, y) = (u(x, y), v(x, y))$ with interpolation.

Output: Next position $\mathbf{x}_{n+1}$ at status n+1.

`// Sample local velocity`

$\mathbf{u}_n = \mathbf{u}(x_n, y_n)$ via bilinear interpolation.

`// RK4 update (time-like parameter` t`)`

Let $\Delta\tau = \sigma\Delta t$.

$k_1 = \mathbf{v}(x_n, y_n)$

$k_2 = \mathbf{v}(x_n + \frac{1}{2}\Delta\tau k_{1,x},\ y_n + \frac{1}{2}\Delta\tau k_{1,y})$

$k_3 = \mathbf{v}(x_n + \frac{1}{2}\Delta\tau k_{2,x},\ y_n + \frac{1}{2}\Delta\tau k_{2,y})$

$k_4 = \mathbf{v}(x_n + \Delta\tau k_{3,x},\ y_n + \Delta\tau k_{3,y})$

`// 4. Combine RK4 stages`

$\Delta\mathbf{x} = \frac{\Delta\tau}{6}(k_1 + 2k_2 + 2k_3 + k_4)$.

`// 5. Append next point`

$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta\mathbf{x}$.

Append $\mathbf{x}_{n+1}$ to ℓ.

return ℓ.

the widely used auto-encoder structure [41] is adopted for integration into the proposed framework.

The auto-encoder demonstrates effective representation learning capability and achieves substantial dimensionality reduction of the input data. An autoencoder reduces the spatial resolution of data features by down sampling, while increasing the number of channels in the data features to compensate for the information loss caused by the reduced resolution. The latent features obtained by the autoencoder are spatially aligned with the original data. The latent features do not need to be forced to approximate a Gaussian distribution, thus better preserving the local structure and details of the original data. Furthermore, autoencoders do not need to consider the distribution of learning data, and their network structure and reconstruction loss are simpler and more intuitive. This construction has a high-level control and modification simplicity.

The implemented autoencoders adopt an identical network architecture, since the velocity field and sketch data share the same input resolution. The structures of the sketch auto-encoder and velocity field auto-encoder

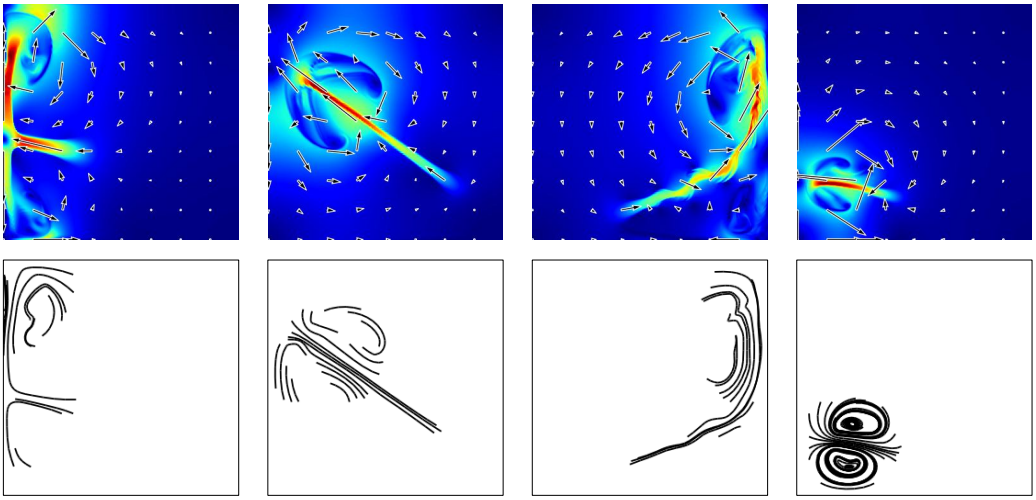

Figure 4.2: The samples of velocity field and extracted sketch. The first row is visualized velocity field data, the second row is correlated extracted sketch data.

are shown in Figure 5.6. The loss function of the auto-encoder is given as follows:

$$\begin{aligned} loss &= ||X_i - \hat{X}_i||^2, \\ \hat{X}_i &= \mathcal{D}(\mathcal{E}(X_i)) \end{aligned} \tag{4.4}$$

where X_i represents the input data, $\hat{X}_i$ represents the output generated data, $\mathcal{E}$ represents the encoding process, $\mathcal{D}$ represents the decoding process.

4.1.3 Latent Diffusion Model

Diffusion models learn the parameters of a denoising process that maps an initial Gaussian prior to the data distribution. Concretely, A fixed forward Markov diffusion is defined for gradually corrupting data with Gaussian noise under a prescribed schedule. And a learnable denoising process is defined for sequentially reconstructing data samples. The forward process is fully specified and requires no learning, whereas the reverse transitions are parameterized by a neural network trained to predict the injected noise at given diffusion time steps. Conventional diffusion models typically operate directly in the high-dimensional data space such as pixel space, which incurs substantial computational and memory costs during both training and sampling process. The latent diffusion model (LDM) is adopted in the proposed framework for mitigating the computational and memory

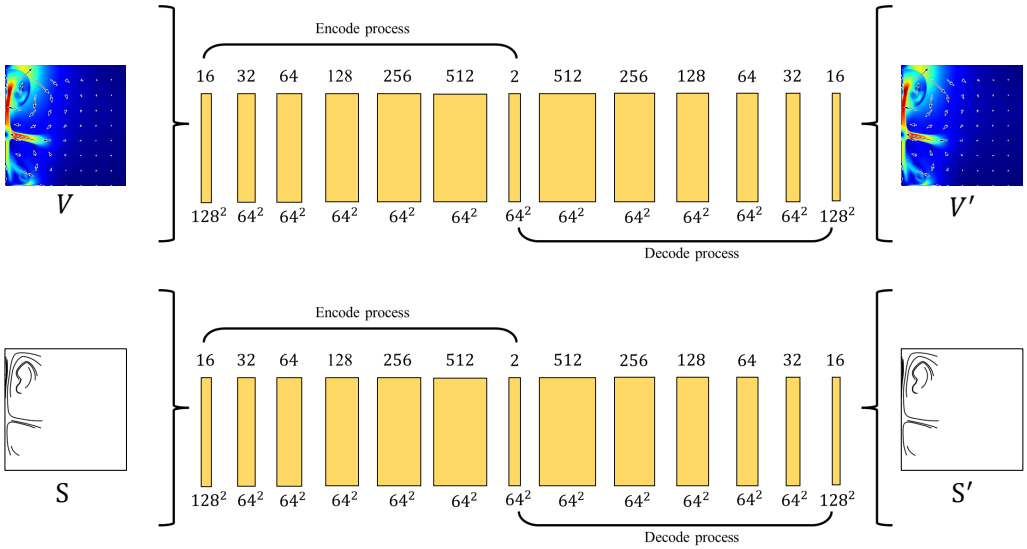

Figure 4.3: The structures of auto-encoders for both velocity fields and sketch inputs.

consumption. The key idea of LDM is to perform the diffusion and denoising process in a lower-dimensional latent space learned by the pretrained auto-encoder. Specifically, the encoder maps the input data into a compact latent representation, which serves as the domain for LDM, while a decoder transforms the denoised latent feature back to the original data space.

The conditional control of LDM in this paper employs a concat operation to concatenate control conditions and noise. The synthesized noise is then fed into a UNet for denoising, ultimately yielding the latent features of the denoised target data. This control method achieves alignment between control conditions and target data at the spatial scale. In the case of this paper, streamline sketches, as a simple representation of the main structure of the velocity field, are naturally suitable for concatenation-based control techniques. Since this paper does not use non-visual modal information such as text for control, a cross-attention mechanism is not used.

The LDM proposed in this research chooses a concatenate strategy to transmit the condition feature into the LDM. As illustrated in Figure4.1, given a 2D velocity field $V \in \mathbb{R}^{C\times H\times W}$, the encoder $\mathcal{E}$ maps V into latent representation $z_0 \in \mathbb{R}^{c\times h\times w}$. A forward diffusion process then progressively perturbs z_0 with Gaussian noise, yielding a noisy latent z_T. In the setting of this research, the sketch y is used as a conditioning input. The condition encoder τ transforms y into an latent representation $\tau(y)$,which is concatenated with the noisy latent to form the conditioned input to the

denoising process. The $\tau(y)$ undergoes the concatenation with z_T to get new z_T where $z_T \in \mathbb{R}^{(C+c)\times h\times w}$. During the denoising process, the U-Net iteratively removes noise from the conditioned latent and produces a reconstructed latent $z_0{}'$. Finally, the decoder $\mathcal{D}$ maps $z_0{}'$ back to the data space to obtain the generated velocity field V'. The loss function of LDM is given as follows:

$$L_{LDM} = \mathbb{E}_{\mathcal{E}(x),y,\epsilon\ \mathcal{N}(0,1),t}[||\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))||_2^2] \tag{4.5}$$

where $\mathcal{E}(x)$ is encoded feature map, y is condition, $\tau_\theta(y)$ is encoded condition, $\epsilon_\theta(..., t)$ is neural backbone that usually is implemented as time conditional U-Net [85].

4.2 Experiment Results

4.2.1 Dataset Generation

The 2D smoke plume simulation scenes was implemented with Phiflow [86], which is a simulation tool designed for the Python environment. The size of the simulation scene is 256 × 256. The area is enclosed by solid boundaries, within which smoke particle clusters are situated. The buoyancy force is applied to induce the upward movement. In the simulation, the parameters are varied including initial buoyancy strength and direction, initial smoke cluster position, simulation time step, and smoke cluster size to generate diverse smoke patterns. A total of 1,000 scene simulation was conducted, each span in 150 time steps, and extract 10,000 velocity fields. The velocity fields are organized in 2 channels with the size of 256 × 256, one represents the x direction of velocity, the other for y direction. The entire simulation time is about 25 hours. Streamline images are created to correspond with exported velocity fields as sketches. Considering the impracticality of trajectory tracing for every individual particle, the particles inside the top 512 grids of velocity field with the highest velocities is traced. This is achieved through the application of a filter mask to block unnecessary grids. The generated sketch data encompasses a single channel with the size of 256 × 256.

4.2.2 Auto-Encoder Implementation

First, the sketch autoencoder was trained for 500 epochs using the Adam optimizer with a batch size of 16. After loading all samples from the data directory, a random shuffle was applied to mitigate potential ordering effects during training. The dataset was split into training and testing

subsets at an 8:2 ratio, comprising 8,000 training samples and 2,000 test samples. The training samples were divided into 9:1 ratio for training and validation. The sketch is processed into a single-channel grayscale image before being written to the GPU. Therefore, the input and output channels of the sketch auto-encoder were both set to 1. Before training process, the z-score standardization [87] was attached for data unit variance scaling, the mean value was set to both 0.5, which ensure fair weight updates and stably gradient propagation. The Mse loss was introduced for evaluating the reconstruction loss. The result is given in Figure 4.4. The sketch images were accurately reconstructed with a loss convergence to 1e-4 accuracy. Although the background color of the encoded and reconstructed sketch data was relatively unstable, the main lines were as sharp as the ground truth.

Next is the velocity field auto-encoder. The training batch size for the velocity field autoencoder is set to 16, with a total of 500 training iterations. The velocity field used for training is represented as a two-channel array or tensor in the data structure; therefore, the number of input and output channels for the velocity field autoencoder is set to 2. The overall network layer construction of the velocity field autoencoder is the same as that of the sketch autoencoder. Prior to the training process, the velocity-field data was normalized to improve numerical stability and facilitate optimization. The resulting outputs are shown in Fig. 4.5. The main flow the velocity field was correctly reconstructed. Yet, some errors existed in the reconstruction of local regions of the velocity field. Weak connectivity was observed in some previously unconnected areas. The smoke motion is turbulent and stochastic, there is a high visual tolerance for minor reconstruction errors, consequently meaning that the errors generated during the decoding process have little impact on the subsequent usage of velocity field.

4.2.3 Comparison Study

The LDM trains for 250 epochs using the Adam optimizer with a batch size of 16. The dataset consists of 10,000 samples, split into 8,000 for training and 2,000 for testing. The training samples were divided into 9:1 ratio for training and validation. This research performs a comparative evaluation against Pix2Pix [4]. Pix2Pix is trained on the same dataset with the same data split ratio, using a batch size of 16 for 300 epochs. Fig. 4.6 presents a visualized comparison between the two frameworks. The velocity fields generated by Pix2Pix exhibit substantial noise, although the overall motion direction is consistent with the input. In contrast, the proposed framework produces cleaner velocity fields that better preserve the geometry implied by the sketch condition. However, the magnitude of flow is not matched due to

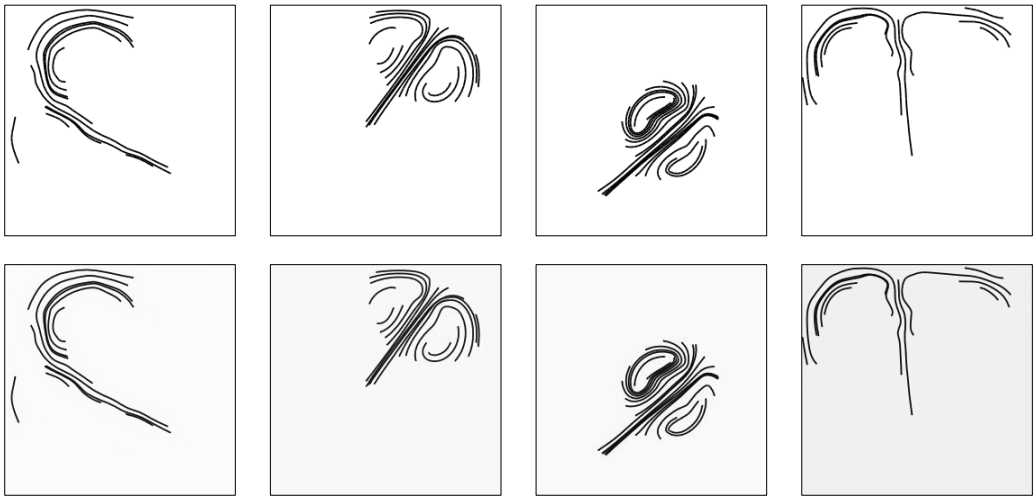

Figure 4.4: The generated results from sketch auto-encoders. The first row is ground truth sketch data, the second row is generated sketch data.

the process of synthesizing a dense velocity field from sparse sketch condition is inherently ill-posed. For quantitative evaluation, this research measure the discrepancy between the generated velocity field and the ground-truth field using the mean squared error (MSE) loss. Specifically, given a prediction V' and the corresponding reference V, the MSE is computed by averaging the squared $\updownarrow_2$ differences over all spatial locations and channels. This metric provides a straightforward assessment of per-element reconstruction accuracy and enables an objective comparison across different methods and experimental settings. The proposed framework has a result of 0.086 while cGAN has 26.372.

In addition, this research conducts experiments by inputting hand-drawn sketches into the proposed framework, the result is shown in Figure 4.7. This demonstrates even with sketch patterns that do not exist in the dataset, the proposed framework successfully generates velocity fields that correspond to the shape of sketches. The training process of cGAN was analyzed. The training of cGAN is essentially an adversarial game between a generator G and a discriminator D under a given condition. The generator takes random noise together with the condition to synthesize samples that aim to match the distribution of real data given the same condition, while the discriminator receives the condition and predicts whether an input sample comes from the real dataset or is generated. Training typically alternates between updating D to improve real–fake discrimination and updating G to produce outputs that are increasingly difficult for D to classify as fake. As this adversarial

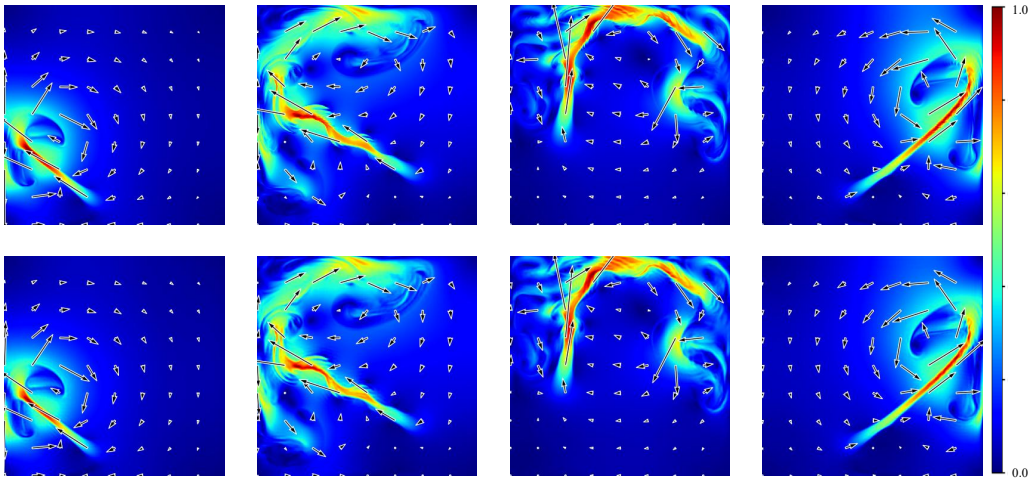

Figure 4.5: The generated results of velocity field auto-encoder. The first row is ground truth velocity field data, the second row is generated velocity field data.

optimization progresses, G learns a condition-to-data mapping and improves generation quality, ideally making the discriminator unable to reliably distinguish generated samples from real ones. On the provided dataset, the cGAN without any dedicated stabilization or tuning becomes overly strong in the early stages of training and can easily distinguish the generator outputs. As a result, the generator receives weak or uninformative gradients and fails to learn the desired sketch-to-velocity mapping. This adversarial training imbalance makes the generator optimization unstable, leading to inconsistent generation quality. Achieving satisfactory performance with GANs typically requires careful tuning of both the network architecture and training hyperparameters. The proposed diffusion-model-based framework learns a stable mapping between the input sketch and velocity field, while cGAN-based framework fails to learn to generate a stable velocity field using the same dataset in the same training epochs.

The generated velocity field is integrated with existing fluid simulator as a guidance force. The guidance force drives the smoke motion follows the movement direction of provided force field. The generated smoke pattern follows the shape of the generated velocity field and input sketch. The example is given as Figure4.8 shows. The smoke source is set as a emitter that adding a initial upward velocity.

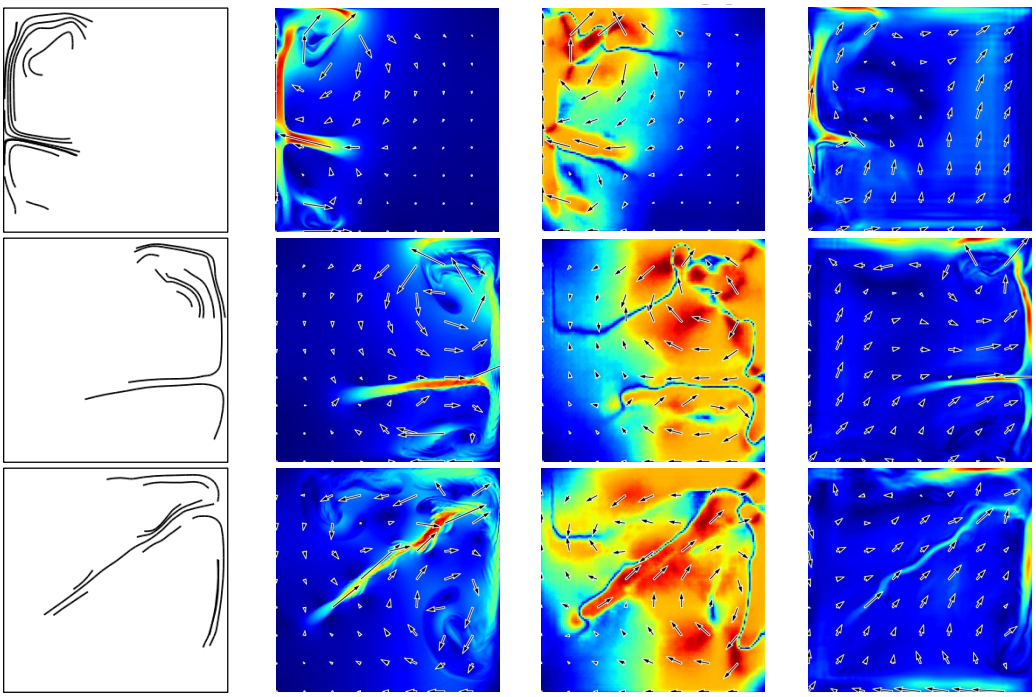

Figure 4.6: Comparison result of velocity field generation. The first column is input sketch, the second column is ground truth velocity field, the third column is generated result of Pix2Pix, the fourth column is generated result of the proposed framework.

4.3 Conclusion

This study propose a sketch-guided 2D velocity-field generation framework using LDM. The proposed method synthesizes velocity fields that are structurally consistent with the geometry implied by the user-provided sketches, while achieving reliable reconstruction fidelity. This research benchmark the proposed framework against cGAN-based method trained on the same dataset under comparable settings. The experiment and comparisons indicate that the proposed LDM-based framework yields substantially improved results, producing cleaner velocity fields with fewer artifacts and stronger adherence to the sketch constraints than the cGAN method.

The proposed framework still has limitations outlined below. The flow types within the provided dataset remain limited. There is potential for improvement by incorporating additional factors like vortex positioning and rotation to exert influence over smoke simulations. Also, the proposed framework is insensitive to small flow patterns with vortexes as shown in

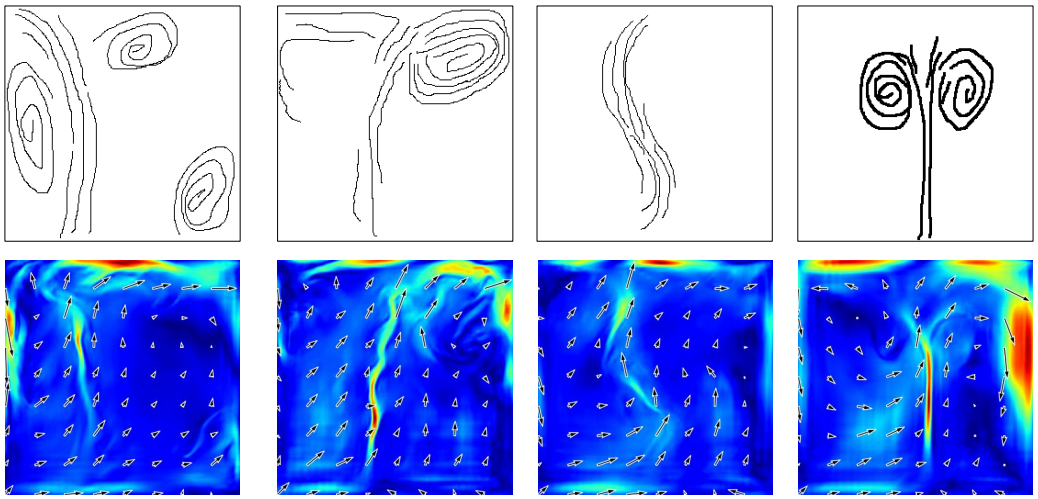

Figure 4.7: The generated velocity field with hand-drawn sketches as input . The first row is input hand-drawn sketch, the second row is generated velocity field.

Figure 4.9. The representation of the vortex needs optimization. The current study constitutes a one-stage model wherein the velocity field is generated directly from a sketch image. However, the process of converting a sketch into velocity field involves multiple sub-stages. Notably, recent research [7] has proved that training sub-stages individually generates better results compared with one stage. For potential future work, reconfiguring the current one-stage structure into a multi-stage framework for enhancing performance will be crucial. Furthermore, strengthening the alignment between sketch data and velocity field is a pressing issue. In next chapter, for the propose of improving velocity field generation accuracy, this dissertation proposes a two-stage framework to connect the sketch and velocity by invoking intermediate representation extracted from velocity field.

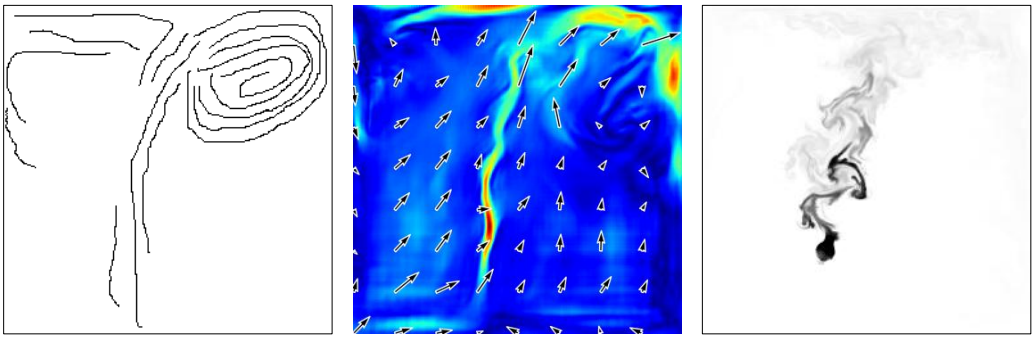

Figure 4.8: The smoke illustration generation guided by the generated velocity field. The first image is the input sketch, the second image is the visualized generated velocity field, the third image is the guided smoke illustration result. The image shows that the generated smoke moves follows the shape of given input sketch.

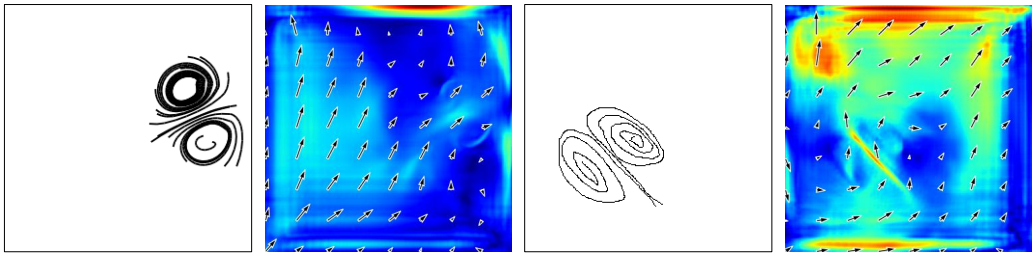

Figure 4.9: Failed generations. The first and third images and are input sketches, the second and fourth images are generated results.

Chapter 5

Sketch-Based Smoke Illustration Design using LCS

This chapter proposes a two-stage LDM-based framework for sketch-guided smoke illustration generation that explicitly incorporates Lagrangian coherent structures (LCS) [10]. Recent advances in generative modeling have substantially expanded the scope of content creation. Nevertheless, generating smoke illustrations remains challenging due to complex fluid dynamics and the need to satisfy strict geometric and motion-related constraints. Prior learning-based fluid design methods often rely on conditional GANs (cGANs) to map sketches to velocity fields, but such approaches can suffer from unstable training and may fail to capture the flow geometric characteristics underlying smoke motion. To address these limitations, a smoke illustration generation method using the two-stage latent diffusion model is proposed. In the first stage, the input sketch conditions an LDM to predict an LCS-related region that reflects flow organization, associated with the hyperbolic finite-time Lyapunov exponent (hyper-FTLE) field. In the second stage, the inferred LCS representation is used as control condition to guide velocity-field synthesis. Experiments demonstrate that proposed method produces velocity fields that better conform to the sketch-implied shapes, and further provides a practical interface for smoke design from hand-drawn inputs, achieving more stable and robust generation than both cGAN baselines and a single-stage diffusion strategy. This chapter builds upon the generative framework in Chapter 4, designing a two-stage generative framework. This two-stage framework also serves the subsequent Chapter 6.

This research proposes a two-stage, sketch-guided 2D velocity-field generation method based on the latent diffusion model (LDM). As illustrated in Figure 7.1, in the first stage, the sketch guides the LCS region generation. The second stage then takes the predicted LCS region as a control signal to guide velocity-field synthesis. The LCS training data are extracted from hyper-FTLE and correspond to the velocity field. Diffusion models [49,51,88] avoid the issue of mode collapse encountered in GAN models and bring about an improvement in the quality of generation. Additionally, to avoid inaccura-

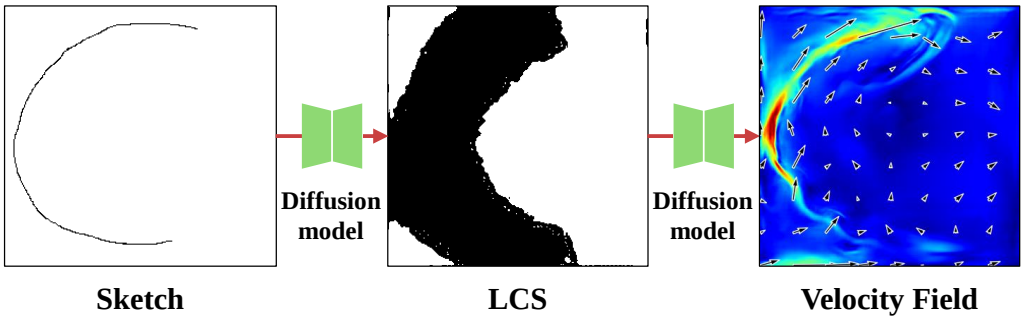

Figure 5.1: Overview of the proposed framework. The user inputs a sketch image into the diffusion model to generate the LCS region. The generated LCS region is then used to generate the velocity field. The velocity field guides smoke simulation and generates smoke-styled illustrations.

cies in data pairing caused by neglecting hidden physical fluid features when directly matching sketch data with velocity fields. This research employ the two-stage method to deconstruct the complex problem of sketch-to-smoke simulation into two sub-problems: LCS generation from given sketch and velocity field generation using LCS constraints. To assess the benefits of the proposed decomposition, The two-stage pipeline with a one-stage baseline is compared to analyze generation stability. This research further conduct comparison experiments to examine the effectiveness of hyper-FTLE as an intermediate representation.

The contributions are given as:

- This research proposes a two-stage LDM-based method for designing velocity field and smoke, applying temporal flow geometric structure to connect the input sketch and output velocity field.
- This research creates datasets for sketch-LCS and LCS-velocity fields, where the LCS data are obtained from the hyper-FTLE field.
- This research provide a fluid design framework that reconstructs the flow field from the given force field calculated using velocity field and LCS data.

5.1 Proposed Framework

The proposed two-stage framework is illustrated as Figure 6.3. The sketch serves as input for the LCS generator to produce the corresponding LCS data, which then guides the velocity field generation in the VF generator.

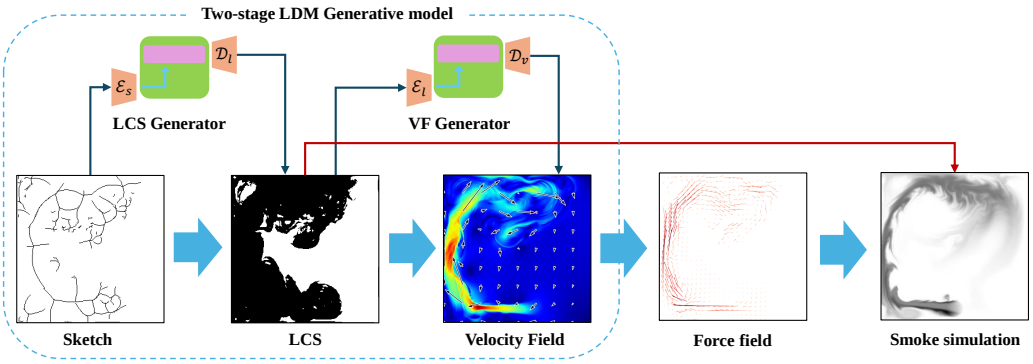

Figure 5.2: The framework of the proposed method. The input sketch is converted into the LCS region by the LCS generator, which subsequently drives the velocity field generation through the velocity field generator (VF generator). Next, the velocity field is used to calculate the force field, providing guidance for smoke design. In the LCS generator, the sketch encoder $\mathcal{E}_s$ encodes the sketch into latent space, and the LCS decoder $\mathcal{D}_l$ decodes the latent feature back to LCS data. In the VF generator, the LCS encoder $\mathcal{E}_l$ encodes the LCS into latent space, and the velocity field decoder $\mathcal{D}_v$ recovers the latent information back to the velocity field.

These two generators are based on LDM. Finally, the force field is derived from the generated velocity field and LCS data for conducting the smoke simulation within the LCS region. The equation of the two-stage model is given as follows.

$$\begin{aligned} V &= f_V(L;\theta_V), \\ L &= f_L(S;\theta_L) \end{aligned} \tag{5.1}$$

The output velocity field is denoted as V, while f_V represents the learnable mapping function used to generate the velocity field from the given LCS data L. Meanwhile, L is generated through the mapping function f_L with the given sketch condition S. The parameters of the VF generator and LCS generator are denoted as θ_V and θ_L. The sketch-to-velocity mapping function is defined by the equation below:

$$V = f'_V(S;\theta'_V) \tag{5.2}$$

Here, f'_V represents a non-linear mapping function from the sketch to the velocity field V, with θ'_V denoting the parameters of the sketch-to-velocity generator. This research decomposes f'_V into two separated mapping functions of f_V and f_L, following the principles of the universal

approximation theorem. This approach allows a complex mapping function to be interpreted as two sub-mapping functions. The LCS data, refining flow field characteristics, is incorporated as a physics-informed condition in the sketch-to-velocity generation process, which explicitly connects f_L and f_V. The f_V and f_L provide detailed information about flow geometry and motion structures. In contrast, f'_V only represents the mapping from the sketch to the velocity field, resulting in limited expressive ability.

Section 5.2 first presents the implementation of the smoke simulation and velocity field acquisition. Next, a detailed explanation of hyper-FTLE and its calculation method is provided. Finally, the computation process for the LCS and sketch data is introduced. Section 5.3 introduces the auto-encoders to encode data into latent features and recover it back to the original data. The LDM model is introduced after auto-encoder, which utilizes auto-encoders to process inputs and control conditions, performing inference and generation within the low-dimensional latent space.

5.2 Data Collection

This research collected sketch, LCS data, and velocity field datasets from smoke simulations to train the LDM-based LCS and VF generator. From the velocity fields that were extracted during the smoke simulation, the hyper-FTLE data is calculated. The LCS data are extracted from hyper-FTLE by the Gaussian Mixture Model (GMM). The sketch data are synthesized from LCS data by heat diffusion.

5.2.1 Smoke Simulation

The velocity field data are extracted from incompressible 2D smoke simulation scenarios. The smoke moves through advection in the simulation, influenced by pressure p and velocity $\mathbf{u}$ at time step t. Assuming p and $\mathbf{u}$ are given as initial conditions, the velocity field is updated with the inviscid Navier-Stokes equations by following [76], also referred to as the Euler equations. There is an upper limit to the time step setting. A large time step may increase the computational errors in each simulation step, potentially leading to unstable simulation results. Similarly, the external force also has an upper limit where a large force may distort the smoke simulation results. The incompressible condition ensures that the velocity field divergence is zero. The boundary condition is set as: the component of the fluid velocity in the normal direction is zero, the fluid can only flow tangentially along the boundary. The boundary condition enforces the confinement of the smoke

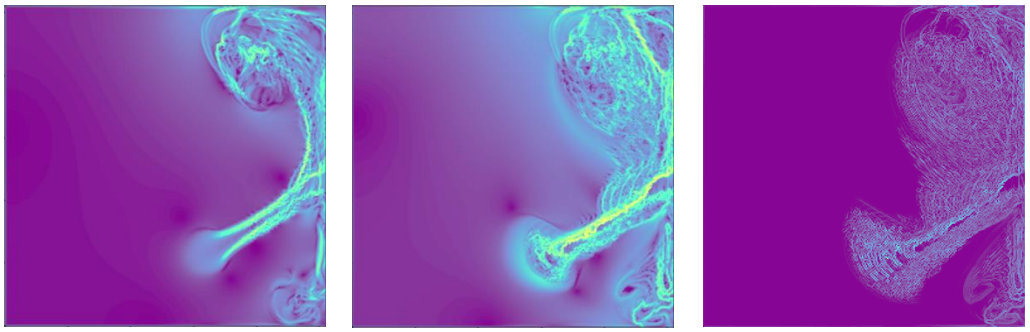

Figure 5.3: FTLE fields and the hyper-FTLE field. The first image is calculated FTLE with integration time 1, the second image is FTLE with integration time 5, and the third image is infused hyper-FTLE.

simulation within the defined boundaries. The semi-Lagrangian scheme [12] updates the velocity field, and the Mac-Cormack method [89] updates the density field.

5.2.2 Hyper-Finite-Time Lyapunov Exponent

This research adopts Hyper-FTLE in the design framework for LCS region generation. The hyper-FTLE, also known as temporal FTLE [90], integrates all FTLE fields over the given integration time interval. The hyper-FTLE collects and compresses all maximum LCS values into one field, which represents more information than a single FTLE field but also characterizes the dynamic properties of the fluid. Before calculating the hyper-FTLE, this research first traces all the particles inside the given fluid area. The initial position of each particle is set to the corresponding grid point.

Suppose the particles at simulation time t for T steps are traced, where T also represents the integration time. The pre-position and post-position of the fluid particle grid are defined as x_t and x_{t+T}. Calculating an FTLE with the specific integration time follows the equation below:

$$\begin{aligned} \sigma_t^T(\mathbf{x}) &= \frac{1}{T}\ln\sqrt{\lambda_{max}(\Delta)}, \\ \Delta = \mathbf{M}^*\mathbf{M}, \mathbf{M} &= \frac{d\phi_t^T(\mathbf{x})}{d\mathbf{x}} \end{aligned} \tag{5.3}$$

Figure 5.4: A sample of LCS data and sketch. From left to right is the hyper-FTLE field, LCS data, and sketch. The extracted LCS region and sketch match the shape of the given hyper-FTLE.

where σ is the FTLE value, $\mathbf{x}$ is a particle at each grid; ϕ_t^{t+T}, which is also equal to $\mathbf{x}(t+T)$, represents the position of each particle when advected for time T from time t. Δ is a 2D matrix, calculated by the gradient of flow map M. As for the calculation of hyper-FTLE, the equation is given as shown:

$$\mathcal{H}_t(\mathbf{x}) = \max_{T_i \in (T_{lb}, T_{ub})} (\sigma_t^{T_i}(\mathbf{x}) \times \mathcal{S}_t^{T_i}(\mathbf{x})) \tag{5.4}$$

where T_i belongs to an integration time interval with lower bound T_{lb} and upper bound T_{ub}. The ridge strength $\mathcal{S}_t^{T_i}(\mathbf{x})$ is extracted from FTLE field $\sigma_t^{T_i}(\mathbf{x})$. The hyper-FTLE elects the maximum value from $\sigma_t^{T_i}(\mathbf{x}) \times \mathcal{S}_t^{T_i}(\mathbf{x})$ defined by a given integration time interval. The ridge strength corresponds to the negative minor eigenvalue of the Hessian matrix [91]. Since the hyper-FTLE is computed by calculating two scalar fields for each grid, the time complexity of hyper-FTLE calculation is $O(n)$. Figure 5.3 shows an example of FTLE and hyper-FTLE fields.

5.2.3 Lagrangian Coherent Structure

The local maximum values in hyper-FTLE indicate significant repelling or attracting fluid behavior at corresponding positions, while also describing the geometric structure of the flow. These values represent the ridge lines of hyper-FTLE, also known as LCS. The LCS provides simplified visual representations of fluid motion patterns and geometric structures over a given time interval. Since the LCS ridge is composed of strokes, which are insufficient for fully capturing the fluid region. This research adopts the

LCS region instead of the LCS ridge as the middle-visualized presentation to connect the sketch and velocity field. Nevertheless, the optimal threshold for filtering the LCS region varies across different hyper-FTLE fields. Hence, the LCS filtering strategy follows the approach of Dualsmoke [7]. In this research, a GMM model is applied to determine the optimal threshold and classify the data into two groups. One group contains high-value data, representing the main hyper-FTLE structure. The other group with lower-value data, corresponds to regions unrelated to fluid areas. The mean values of two extracted hyper-FTLE groups are tested for LCS extraction. Examples are shown in Figure 5.5. When the filter threshold is set as the minimum mean value, the extracted LCS region blurs and loses the local geometric features of hyper-FTLE. Conversely, setting the threshold to the maximum mean value results in fragmented ridge areas, and the shattered LCS data are inadequate to match the velocity field area. Using the average of the mean values of each class as a threshold addresses these issues. Additionally, a Gaussian filter is applied before clustering to reduce the noise present in hyper-FTLE fields. The sample of LCS is given in Figure 5.4.

5.2.4 Sketch Synthesis

After obtaining the LCS data, this research further simplify it to extract line structures as sketches for training. Specifically, the ridge lines of the LCS region are used as sketches to establish the mapping between the sketch and LCS data. The heat diffusion equation [92] is applied to generate a ridge map and extract the height ridge [93]. The heat source is placed at the boundary of the LCS region, and heat diffusion outside the area is blocked. The LCS region is heated until no temperature-free area remains. The equation is shown as follows:

$$\frac{\partial \phi}{\partial t} = \alpha(\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2}), x, y \in \Omega \tag{5.5}$$

The heat diffusion equation describes the heat distribution in a given domain over time, where ϕ represents temperature, Ω is LCS region, and α is thermal diffusivity positive constant. Since the heat diffusion is calculated for each grid, the time complexity is $O(n^2)$. Upon completing heat diffusion, the boundaries of the LCS region exhibit the highest temperature in the heatmap, while the local central areas of the LCS show the lowest temperature. The $-log()$ function is applied to transform the heat map into a distance map, where lower temperatures correspond to higher distance values. Therefore, the local central areas of the LCS region will hold the largest distance values in the distance map. Ridge lines are constructed by filtering out points with

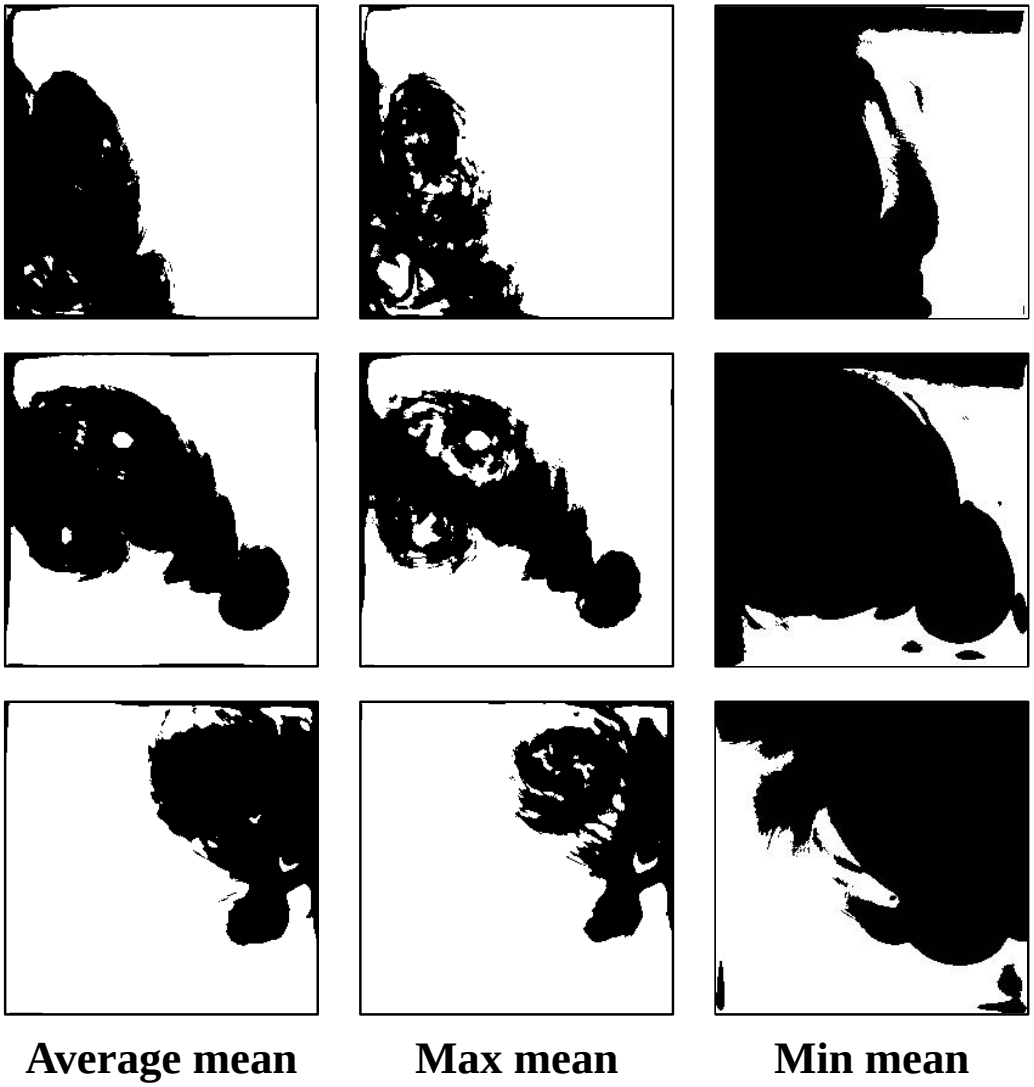

Figure 5.5: The examples of different LCS region data extracted by different strategies. The first column presents results using the average mean value as the threshold; the second column shows results with the maximum mean value as the threshold; the third column shows the result with the minimum mean value as the threshold.

the largest distance values. The ridge extraction is performed by calculating zero-crossing, negative minor eigenvalue, and average direction in parallel. Finally, the length of extracted sketch segments with a threshold is filtered to keep the main structure. The sketch sample is given in Figure 5.4.

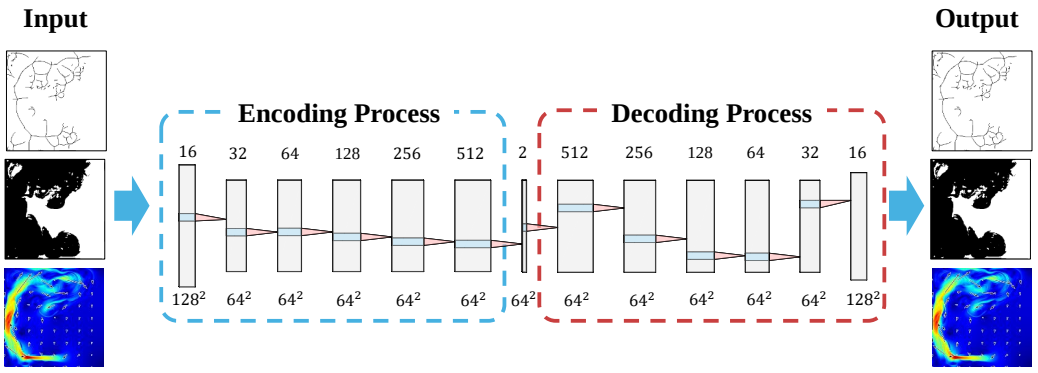

Figure 5.6: The structure of the implemented auto-encoder. The structure inside the blue box represents the encoder, and the structure inside the red box represents the decoder. The numbers above the network structure (16, 32, ...) represent the layer channel number, while the numbers below the network structure (128^2, 64^2, ...) indicate the layer size, calculated as the product of width and height. A sketch auto-encoder, LCS auto-encoder, and velocity field auto-encoder (VF auto-encoder) are trained in this research.

5.3 Network Structure

This study employs two LDM networks to construct the LCS generator and VF generator. The generators learn the relationships between sketch, LCS data, and velocity field. The proposed two-stage generative framework structure generates the global LCS region from the initial input sketch, and the local detailed velocity field is generated from the guided LCS region. The sketch data and LCS data are paired for training the LDM-based LCS generator, and the LCS data and velocity field data are paired for the LDM-based VF generator training.

5.3.1 Auto-encoder

For the two generators, this research employ three autoencoders to map the input data into compact latent representations and to decode the denoised

latent features back to the original data domain. A fundamental auto-encoder structure [41] is implemented as Figure 5.6 shows. This choice is informed by the flexibility and adaptability of the auto-encoder. Additionally, the autoencoder provides effective representation learning and enables substantial dimensionality reduction of the input data. These auto-encoders share an identical structure. The loss function of the auto-encoder is given as follows:

$$\begin{aligned} loss &= ||X_i - \hat{X}_i||^2, \\ \hat{X}_i &= \mathcal{D}(\mathcal{E}(X_i)) \end{aligned} \tag{5.6}$$

where X_i is input data, $\hat{X}_i$ is output data, $\mathcal{E}$ is the encode process, $\mathcal{D}$ is the decode process.

5.3.2 Latent Diffusion Model

Diffusion models (DMs) have advanced rapidly in recent years and have emerged as a leading paradigm among modern generative models alongside GANs. Conceptually, a DM learns the parameters of a Markovian denoising process that progressively transforms a Gaussian noise prior into samples from the data distribution. This framework consists of two complementary components: a forward diffusion process that incrementally corrupts real data by adding Gaussian noise according to a predefined schedule, a reverse denoising process that reconstructs data by sequentially removing the injected noise. The forward diffusion process is fully defined by the chosen noise schedule and thus involves no learnable parameters. In contrast, the denoising process is learned, most commonly by training a neural network to estimate the injected noise at each diffusion timestep. Nevertheless, standard diffusion models typically operate and sample in the high-dimensional data space, which incurs substantial computational and memory overhead. the latent diffusion model (LDM) is adopted to improve efficiency, where the denoising process is learned and executed in a lower-dimensional latent space. This design relies on pretrained autoencoder. The encoder compresses the input data into a compact latent feature map, and the decoder subsequently maps the denoised latent representation back to the original data domain for reconstruction.

As shown in Figure 5.7, the encoder $\mathcal{E}_l$ and $\mathcal{E}_v$ encode the LCS data L and velocity field V into feature map z_0. The latent code z_0 is progressively corrupted with Gaussian noise via the forward diffusion process, yielding the noisy latent z_T. In this formulation, the condition y is provided as an input guidance. A condition encoder τ maps y to a latent feature representation

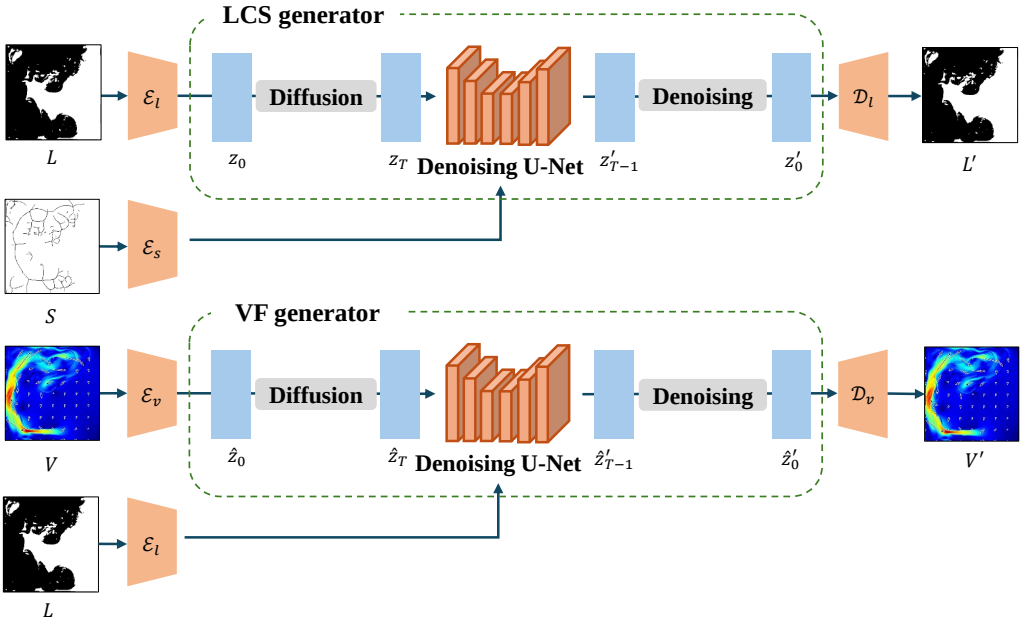

Figure 5.7: The training structure of the two-stage LDM-based generators. In the LCS generator training process, the LCS data L is encoded to z_0 by the LCS encoder $\mathcal{E}_l$, after which noise is iteratively added until reaching z_T in latent space. The condition sketch S is encoded by the sketch encoder$\mathcal{E}_s$, which then modulates the denoising process. The denoised z_0' is decoded into recovered LCS data L' by LCS decoder $\mathcal{D}_l$. The VF generator shares the same network structure as the LCS generator, whereas the control condition is changed to LCS data L that encoded by the LCS encoder $\mathcal{E}_l$. The latent feature $\hat{z_0}$ is encoded from the velocity field V. The generation target is set to the recovered velocity field V', which is processed by encoder $\mathcal{E}_v$ and decoder $\mathcal{D}_v$.

$\tau(y)$, which is concatenated with z_T to form the conditioned latent input to the denoiser. During the denoising process, a U-Net iteratively removes noise from this conditioned latent and produces a reconstructed latent $z_0{}'$. The decoder $\mathcal{D}$ maps $z_0{}'$ back to the data space to obtain the outputs L' and V'. The loss function of LDM is given as follows:

$$L_{LDM} = \mathbb{E}_{\mathcal{E}(x),y,\epsilon\ \mathcal{N}(0,1),t}[||\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))||_2^2] \tag{5.7}$$

where $\mathcal{E}(x)$ is encoded feature map, y is condition, $\tau_\theta(y)$ is encoded condition, $\epsilon_\theta(..., t)$ is neural backbone that usually is implemented as time conditional U-Net [94].

5.4 Implementation Details

The evaluations are performed to assess the generation accuracy of the proposed framework and its consistency with the input sketches. The objective metrics was employed to measure the discrepancy between generated outputs and reference data, thereby providing a reproducible evaluation of sketch adherence and overall reconstruction accuracy.

5.4.1 Dataset Generation

Dataset generation was performed on a Windows system-based workstation equipped with an Intel Core i9-12900K CPU. The total simulation time was approximately 25 hours. The smoke plume simulation scenes was implemented using Phiflow2.3.0 [95], a simulation tool for the Python environment. The grid size of the simulation was set to 256 $\times$ 256, with a boundary enclosing the domain and smoke clusters positioned within. The simulation time step Δt was set to 0.5. The simulation parameters were randomly initialized to generate diverse smoke patterns, including the external force strength, external force direction, smoke cluster position, and smoke cluster radius. A vertical vector with randomly set values was applied to induce upward movement, with the upper bound of the vertical vector set to 1. A horizontal vector controls the sideways movements of smoke, with its value randomly set in [-2, 2]. These settings ensured the smoke flowed stably within the defined total simulation timestep.

A total of 1,000 scenes was simulated, each over 150 timesteps. Since the smoke dynamics evolve over time, the velocity fields was sub-sampled at a fixed interval of 10 timesteps, collecting from timestep 30 through timestep 120. Before the 30th time step, the smoke area remains small, resulting in sparse data with indistinct flow features. After the 120th time step, the smoke contacts the boundaries and compresses inward, causing the flow features to become chaotic. A total of 10,000 velocity fields were collected, each exported as *.npy* files. The generated velocity field consisted of two channels with a resolution of 256 $\times$ 256. Two channels represent the x direction and y direction of velocity, respectively. Before calculating the hyper-FTLE, each FTLE is computed by the forward motion with an integration time step ranging from 1 to 10. If the integration timestep is chosen excessively large, the extracted structures tend to become repetitive across samples, resulting in duplicated features and increased redundancy. The calculation

of M follows the central-difference-based equation below:

$$\mathbf{M} = \begin{bmatrix} \frac{x_{right}(t+T)-x_{left}(t+T)}{x_{i+1,j}(t)-x_{i-1,j}(t)} & \frac{x_{up}(t+T)-x_{down}(t+T)}{y_{i,j+1}(t)-y_{i,j-1}(t)} \\ \frac{y_{right}(t+T)-y_{left}(t+T)}{x_{i+1,j}(t)-x_{i-1,j}(t)} & \frac{y_{up}(t+T)-y_{down}(t+T)}{y_{i,j+1}(t)-y_{i,j-1}(t)} \end{bmatrix} \tag{5.8}$$

where x and y represent the x, y directions of velocity in 2D, and i and j represent the horizontal and vertical directions respectively.

After generating the FTLE set, the hyper-FTLE is composed and the LCS region from hyper-FTLE is seperated using the given GMM. Sketch images are created to correspond with the separated LCS region. The heat diffusion equation is solved using the finite-element method, which can be rewritten as follows:

$$\begin{aligned} \frac{\phi_{i,j}^{k+1} - \phi_{i,j}^{k}}{\Delta t} =& \phi_{i+1,j}^{k} + \phi_{i-1,j}^{k} + \\ & \phi_{i,j+1}^{k} + \phi_{i,j-1}^{k} - 4\phi_{i,j}^{k} \end{aligned} \tag{5.9}$$

where $\phi_{i,j}^{k}$ denotes the value at grid (i, j) after k iterations.

$$\begin{cases} \phi_{(i,j)} = 1 \times 10^{-16}, & (i,j) \text{ in the inner area} \\ \phi_{(i,j)} = 1, & (i,j) \text{ on the contour} \end{cases} \tag{5.10}$$

The temperature of the inner area is set to 1×10^{-16} to prevent numerical instability arising from the logarithm of zero during logarithmic calculations. The contour is defined as the heat source with a temperature of 1. After logarithmic calculation, the boundary has the smallest distance 0 in the distance map. In the experiment, Δt was set as 0.2, following the methodology in previous work [92]. Both the generated LCS maps and the sketch inputs are represented as single-channel images with a spatial resolution of 256×256.

5.4.2 Network Training

The auto-encoder training was conducted on the Linux system with NVIDIA RTX3090. The sketch auto-encoder, LCS auto-encoder, and VF auto-encoder shared the same network structure. All auto-encoders were trained for 500 epochs using an Adam optimizer with a batch size of 16. The dataset was partitioned into training process and testing process at an 7:1:2 ratio, using 7,000 samples for training, 1,000 for validation and 2,000 samples for testing. The training and test data are shuffled before the training process to enhance robustness. The normalization was applied to the velocity field

data to ensure the training stability. Although the sketch and LCS data are one-channel binary images, and the velocity field has a two-channel data structure. In the LDM, the conditions are incorporated into the noisy latent representation, thereby steering the denoising trajectory and guiding the generation process. Therefore, the feature channels of the sketch, LCS, and velocity field after encoding are all set to 2.

Each LDM in the two-stage framework was trained for 200 epochs using the Adam optimizer, requiring approximately 30 hours per model. A batch size of 16 was used, with 8,000 samples for training process and 2,000 samples for testing. In the training process, the data was segmented into 9:1 ratio. The LDM training continued for 59 hours. The LDM control condition is merged with noise to guide the generation process, with the number of channels in the control condition matching the number of channels in the noise data. The MSE loss was selected as the training loss for both the auto-encoder training and the LDM training process. The weight files with the lowest training loss were kept as pre-trained weights. This research also conducted a comparative analysis with the cGAN method, the one-stage method, and the FTLE-driven two-stage method. The compared methods were trained for 200 epochs.

5.5 Smoke design

This research implemented a smoke design process to explain the usage of the generated velocity field. The LCS data and velocity field are utilized to calculate the force field that drives the smoke-pluming simulation. The equation is defined as follows:

$$\mathbf{f} = \alpha \mathbf{y} + \mathbf{F}(\mathbf{p}) \tag{5.11}$$

where the $\mathbf{y} = (0, 1)$ is a unit vector pointing in the vertical upward direction, α is a positive control parameter. The guiding force $\mathbf{F}(\mathbf{p})$ is determined by the LCS region. The $\mathbf{F}(\mathbf{p})$ is calculated as follows:

$$\mathbf{F}(\mathbf{p}) = \begin{cases} \frac{c}{\Delta t}(\mathbf{V}_G - \mathbf{V}_S), & \text{if } \mathbf{p} \in \Omega; \\ 0, & \text{if } \mathbf{p} \notin \Omega. \end{cases} \tag{5.12}$$

where c donates a user-defined constant, which is set to 1 by default. A larger c strengthens the constraint from the guided velocity field while weakening the influence of external force. The experiments indicated that $c \in [0.3, 1.3]$ generates satisfactory results. $\mathbf{V}_G$ and $\mathbf{V}_S$ represent the velocity

	Sketch AE	LCS AE	VF AE
MSE Loss	1×10^{-4}	4×10^{-4}	0.0097

Table 5.1: The auto-encoder MSE loss of sketch, LCS and VF auto-encoder (AE) (from left to right). The MSE losses demonstrate the convergence of Auto-encoder training.

field generated by LDM and the velocity field in the simulation scene. The guiding force is applied exclusively within the LCS region. The vorticity confinement force [13] is applied to strengthen the details of the generated flow field.

5.6 Results

First, the performance of trained auto-encoders is evaluated. The visualized results are presented in Figure 5.8. The first row shows the sketch inputs, the second row presents the corresponding LCS data, and the third row displays the resulting velocity fields. Overall, both the sketch and LCS maps are reconstructed with high fidelity, only slight background-level fluctuations are observed, primarily manifested as minor intensity variations. The velocity fields are reconstructed with high overall accuracy, though small deviations remain in fine-scale details. The quantitative result is given in Table 5.1.

This research compare the proposed two-stage framework with the one-stage framework [9] and cGAN [4]. Figure 5.9 illustrates a visual comparison between the frameworks mentioned above. Although the cGAN generates velocity fields that align with the flow directions of ground truth, significant noise is present within the velocity fields. Additionally, the velocity magnitude is abnormally higher than the ground truth. The one-stage model generates velocity fields that align with the ground truth field structure. When the flow area is small, the generation quality deteriorates. The proposed method achieved similar results to the one-stage framework when the flow was advected over sufficiently long time intervals. For cases with large-scale flow patterns, the generated velocity fields preserve the dominant motion and closely reproduce the geometric structure of the ground truth. Since the generation task is inherently ill-posed, it is impossible for sketches to match the velocity field perfectly.

For quantitative evaluation, the discrepancy is measured between the generated outputs and the ground truth using the mean squared error (MSE). The results are presented in Table 6.1. The proposed framework achieves

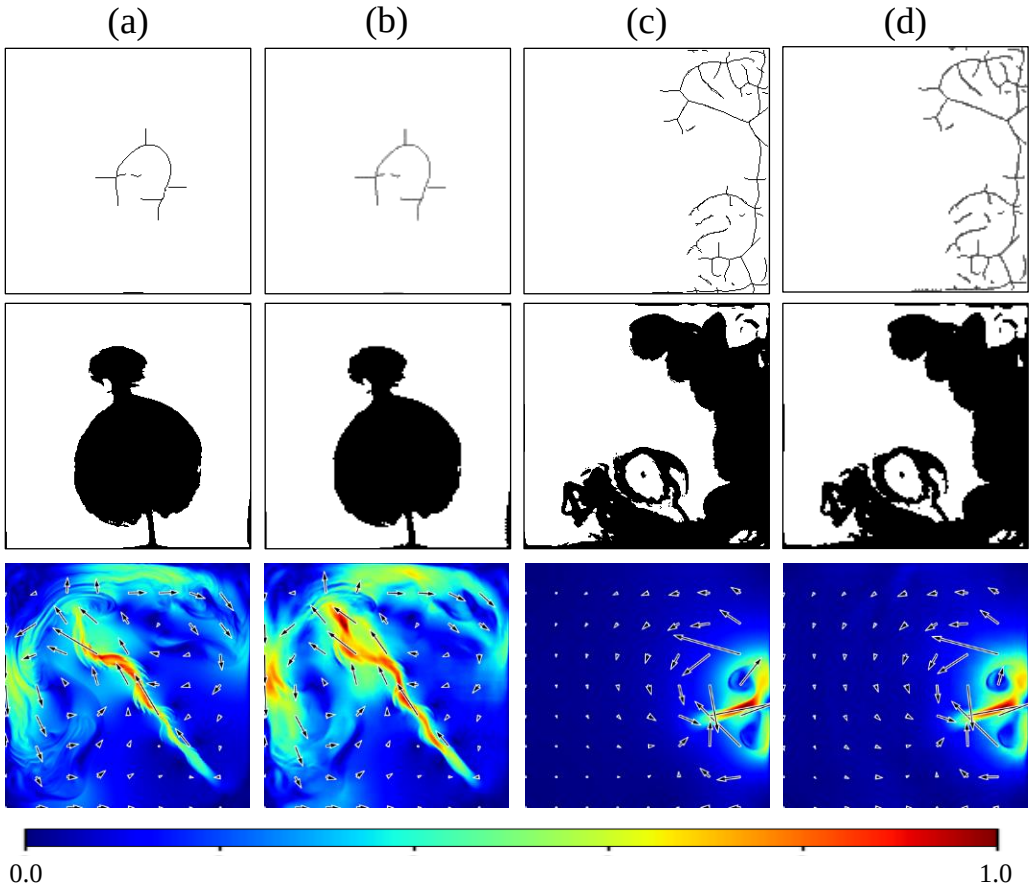

Figure 5.8: The visualized comparison of ground truth data and auto-encoder generation results. The first row represents the sketch, the second row corresponds to LCS data, and the third row depicts the velocity field. The velocity field is normalized to the [0, 1] range. Columns (a) and (c) represent ground truths. Columns (b) and (d) are recovered results.

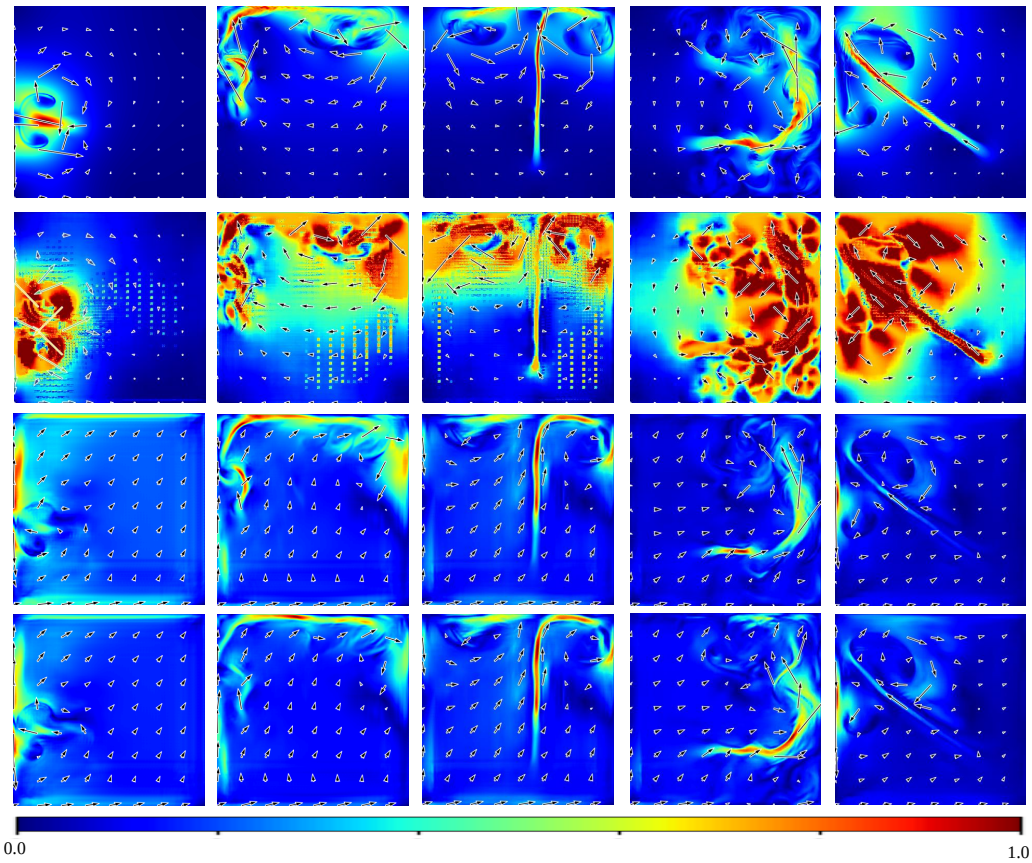

Figure 5.9: The visualized comparison results among the previous cGAN approach (second row), the one-stage model (third row), our method (fourth row), and the ground truth (first row). The second row is the results of cGAN, where the velocity values are significantly higher. The third row represents the results of the one-stage model, while the fourth row shows the results. The third and the fourth rows exhibit similar fluid shapes, but the fourth row achieves superior performance in quantitative evaluations.

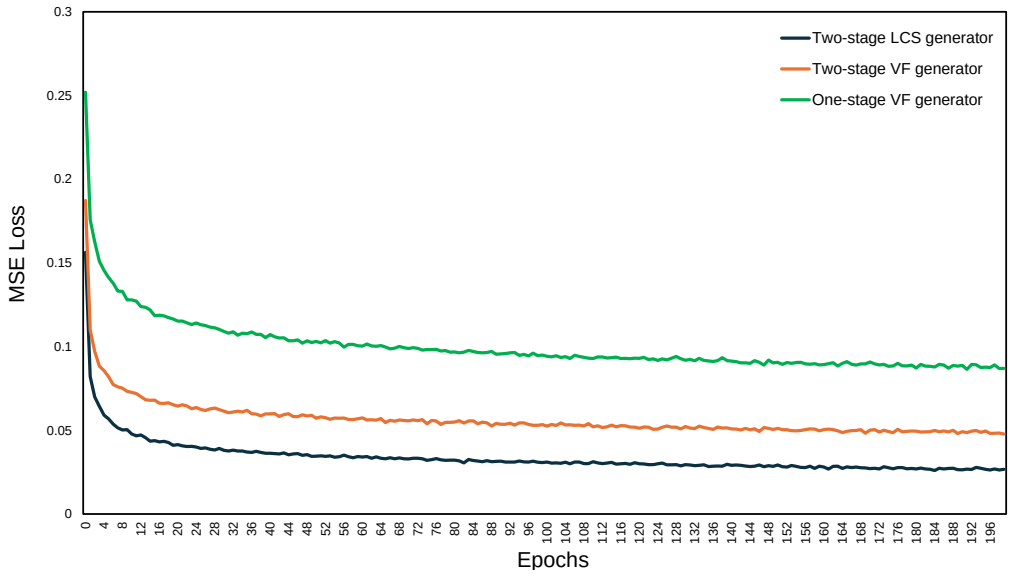

Figure 5.10: The chart of MSE training loss comparison result. The x-axis represents epochs, the y-axis represents MSE loss. The green line represents the one-stage VF generator loss, the orange line represents the two-stage VF generator loss, and the blue line represents the two-stage LCS generator. The two-stage generator exhibits a lower training loss compared to the one-stage generator.

an MSE of 0.047, whereas the one-stage framework and cGAN yield 0.089 and 26.372, respectively. From the quantitative perspective, the proposed method demonstrates higher accuracy in velocity field generation compared to the one-stage framework. During training, the loss decreases sharply over the first 30 epochs and then continues to decline more gradually until convergence. The training chart is shown in Figure 5.10. When the LCS region is overly small, the corresponding sketch becomes excessively sparse, which increases the uncertainty of the condition. As a result, the velocity field prediction exhibits larger errors and may fail to preserve the dominant flow structure, leading to noticeable fragmentation of the main velocity-field patterns. Additionally, increased artifacts are observed near the velocity-field boundaries, as illustrated in the first column of Fig. 5.9.

In addition, the generation controlability of the proposed method and the one-stage method is evaluated. The resulting samples are shown in Figure 5.11. When the input sketch consists of a single stroke-like line, the one-stage method generates unstable results and fails to reconstruct the geometric structure with high velocity magnitude estimated from the sketch.

Generator	MSE Loss
cGAN	26.372
One-Stage	0.089
Ours(LCS generator)	0.026
Ours(VF generator)	0.047

Table 5.2: The MSE loss is compared with the cGAN-based velocity field generation method (first row) and the one-stage diffusion-model-based velocity field generation method (second row). The generation accuracy of the proposed method is higher than that of both methods.

In contrast, the proposed method generates velocity fields that match the sketch shape. With more complex sketch inputs, the one-stage method results in velocity fields exhibiting unstable movement, whereas the two-stage model demonstrates improved alignment between velocity fields and the input sketches. In the first example, The one-stage framework fails to generate the main structure of the given sketch, the two-stage framework generates the flow structure matched with given sketch. When the input are multiple strokes, one-stage framework fails to generate multiple flows inside one velocity field, two-stage successfully generates multiple flows match with given sketches. In the third example, the given sketch image is composed by complex sketches. Although one-stage framework successfully generates a motion pattern follows the basic shape of given sketches, the main flow structure is shaking. Two-stage method generates a stable flow pattern from given sketches.

This research also evaluated the performance of hyper-FTLE and FTLE. The training process is the same as the model based on hyper-FTLE. As shown in Table 5.3, the FTLE-based model achieves a comparable MSE loss of 0.049 during training, which remains higher than that of proposed framework. In addition, the variance and coefficient of variation (CV) of sketch data generated from hyper-FTLE and FTLE were compared. The hyper-FTLE-based sketch is 4.37 whereas the FTLE-based sketch is 5.85, indicating that the former exhibits lower dispersion. Sketches with lower CV values present a lower level of complexity, making it uncomplicated for amateurs to draw.

The smoke designs are illustrated in Figure 5.12. Sketches absent from the dataset were drawn to guide smoke generation. The placement of multiple smoke sources on specific sketch locations was enabled to enhance the generation performance. The smoke scene results were generated by the

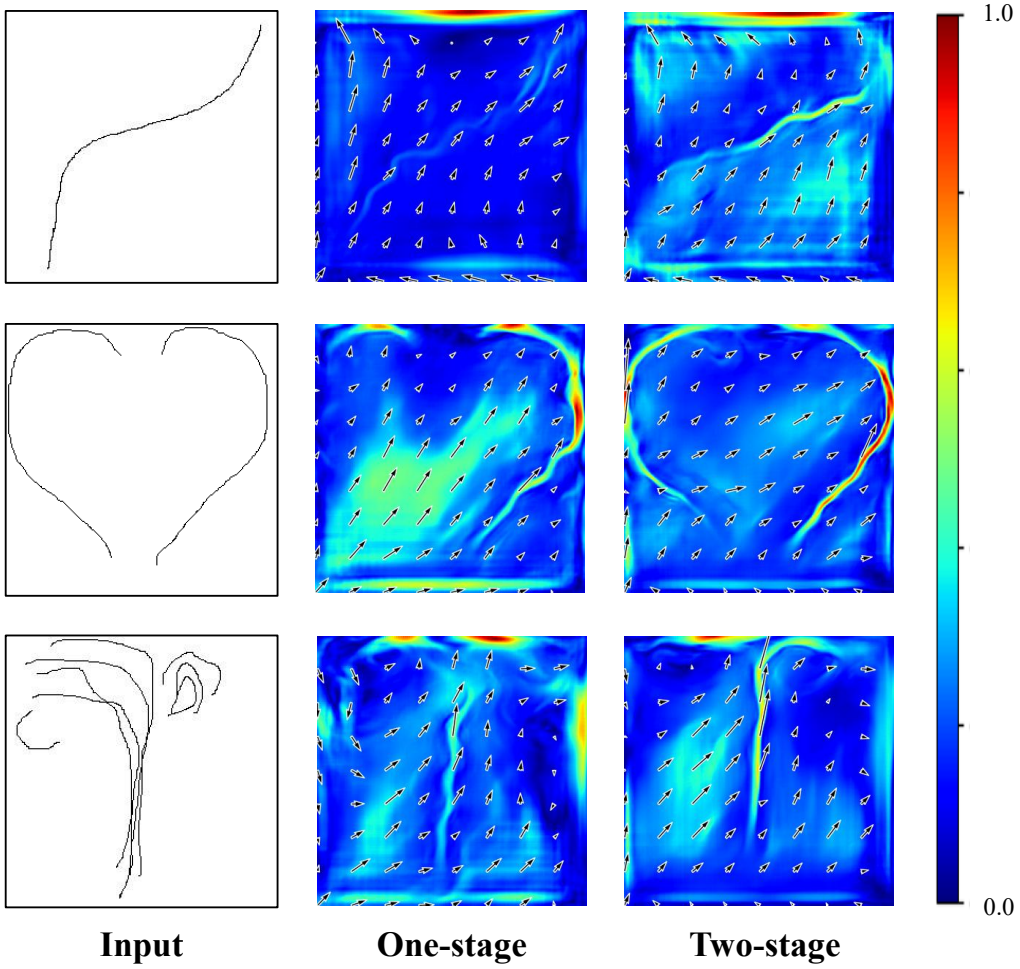

Figure 5.11: The examples of generated velocity fields comparison using hand-drawn sketches as input. The first column shows input hand-drawn sketches. The second column is visualized velocity field images generated by the one-stage model. The third column presents visualized velocity field images generated by the two-stage model.

Method	MSE Loss	
	LCS generator	VF generator
hyper-FTLE	0.026	0.047
FTLE	0.023	0.049

Table 5.3: The quantitative comparison results between hyper-FTLE and FTLE. The difference in the impact of hyper-FTLE and FTLE on velocity field generation is insignificant.

Phiflow [95] simulator. The advections were driven by the generated velocity fields and guided force fields. In Figure 5.12 (a), (b), and (f), scenarios were conducted with letter sketches. Additionally, horizontal movement was tested, as illustrated in Figure 5.12 (c). Simple geometric shapes, such as arch and heart shape, were adopted as inputs as depicted in Figure 5.12 (d) and (e). Figure 5.12 (g) and (h) depict scenarios involving smoke plumes in a house and a censer. Smoke and solid boundary collision simulation was performed in Figure 5.12 (g) and (h), where the velocity field and solid obstacles modulate the smoke dynamics. The generated smoke design illustrations matched the shape of hand-drawn sketches.

The proposed two-stage LDM-based framework exhibits superior stability and accuracy in velocity field generation compared to cGAN-based [5, 7] and one-stage methods [9]. The proposed framework exhibits superior accuracy compared to the cGAN-based framework while ensuring enhanced numerical stability in velocity field generation. Furthermore, the proposed framework demonstrates improved robustness and accommodates diverse input modalities compared to the LDM-based one-stage model.

5.7 Conclusion

This research proposes a two-stage sketch-guided framework for 2D velocity-field generation based on LDM. The proposed method produces velocity fields that are strongly consistent with the input sketches and preserves the sketch-implied geometry with high fidelity. The proposed method also allowed non-experts to create smoke motion illustrations or animations that conform to input sketches with simple strokes, without requiring prior expertise. This capability significantly reduced the cost and effort required for the artistic illustration design and creation.

Future work will expand the dataset to include a broader range of shape

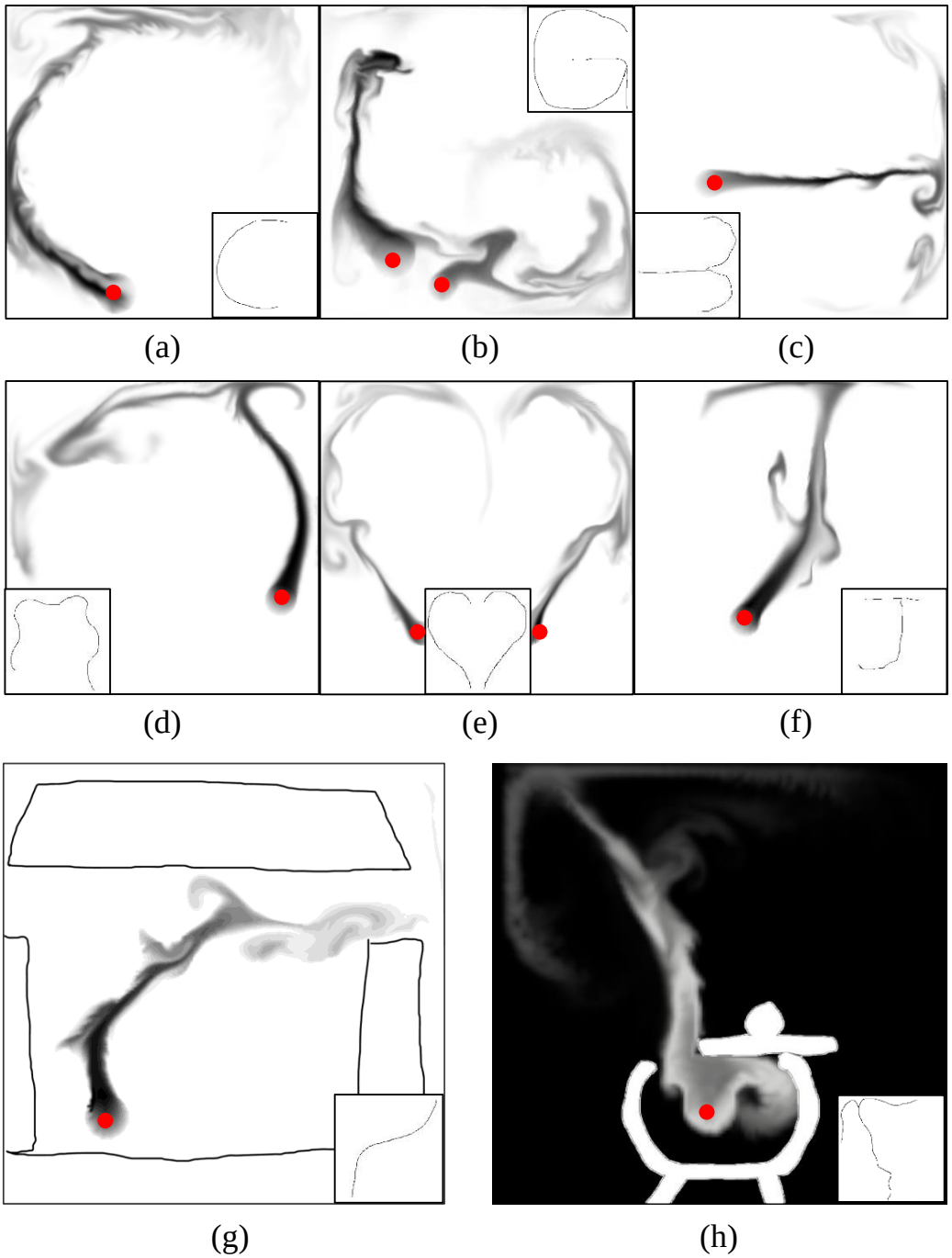

Figure 5.12: The examples of designed smoke illustrations. The red point represents the smoke source position. In (a), (b), and (e), the resulting smoke plumes and flow structures form the shapes of the letters ”C”, ”G”, and ”J”. A single smoke source is set for ”C” and ”J”, while two sources are set for ”G”. In (c), the smoke flows horizontally. In (d), the smoke moves in an arch shape. In (e), the smoke forms a heart-shaped plume with two set smoke sources. In (g), smoke occurs in a house scene and drifts out through the window. In (h), the smoke emerges from solid censer-shaped obstacles.

categories, including numbers, letters, and vortices to improve the robustness of the velocity field generalization. Furthermore, the LCS region has certain limitations in controlling the velocity field detail generation. Although the LCS region constrains motion direction of the main flow, the local rotational behavior and motion details fails to be synthesized correctly. In the next chapter, to address the issue of insufficient control ability of LCS, this dissertation uses a stream function as an intermediate representation connecting the sketch and the velocity field. The new proposed generation framework is enhanced to perceive the rotational motion information and local details in the velocity field.

Chapter 6

Sketch-Based Smoke Illustration Generation using Stream Function

This chapter presents a two-stage framework for smoke illustration generation using stream function as intermediate representation [11]. Physics-aware anime-style smoke illustration remains challenging due to the intricate flow dynamics and anime-style constraints. This chapter introduces a two-stage latent diffusion framework that embeds physical control into the smoke illustration design. In the first stage, a sketch-guided latent diffusion model (LDM) yields a stream function encoding smoke flow dynamics; in the second stage, the stream function guides velocity field synthesis via LDM. An integrated physics simulator and a large language model (LLM) then perform smoke simulation and anime-style transferring. By explicitly modeling geometric structure and flow characteristics, transcends the limitations of prior text-based and pixel-based controls. This chapter builds upon the two-stage generation framework proposed in Chapter 5, replacing the intermediate representations of the generation process with stream function. As an important representation of flow field motion information, stream function is used as conditions for motion control in Chapter 7.

This research proposes a sketch-based smoke illustration generation framework that leverages the stream function as guidance, built upon the latent diffusion model (LDM) [51]. As illustrated in Figure 6.1, the input sketch conditions the synthesis of a stream function, which is subsequently used as a control condition for velocity field generation. Both the conditioning input and the target representations are encoded into a latent space, where the diffusion-based generation is performed. The resulting latent features are then decoded to obtain the velocity field, which is used to drive the downstream simulation step and produce smoke illustrations whose motion is consistent with the prescribed flow direction. Streamlines, which capture the global dynamical structure of the flow, are used as sketch-based guidance during training. The stream function serves as an inter-

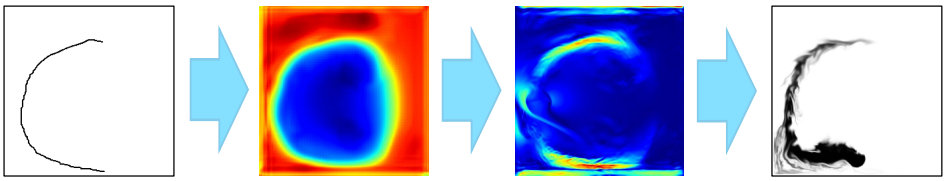

Figure 6.1: The framework of this study. The input sketch is first mapped to a stream function by the stream-function generator. The predicted stream function then conditions the velocity field (VF) generator to synthesize the corresponding velocity field. The generated velocity field is subsequently used as a guiding force to drive the smoke simulation.

mediate representation that provides smoothly varying control and encodes rotational characteristics of the flow that are not explicitly conveyed by the sparse streamline sketches. This research compares with DualSmoke [7] and DiffSmoke [10] to demonstrate the effectiveness of this study. Furthermore, this research provides a style transfer module facilitated by a large-language-model(LLM) agent to generate anime-stylized smoke illustrations.

6.1 Methods

6.1.1 Data Collection

This research first builds incompressible 2D smoke simulations to collect velocity field data for training. At each time step, the fluid velocity is advanced using the incompressible inviscid Navier–Stokes equations, which account for self-advection of the flow, pressure forces, and external body forces. The external force consists of constant horizontal and vertical components. Incompressibility is enforced by solving for a pressure field that makes the velocity field divergence-free. The flow domain is bounded by solid rectangular boundaries, the fluid is not allowed to move across the boundaries. The normal component of the velocity vanishes at the boundaries. Time integration of the velocity field is performed using a semi-Lagrangian scheme.

The stream function serves as a latent representation that encapsulates the rotational information and continuous dynamics feature of a 2D VF. Specifically, stream function encodes the incompressible component of the flow, where the sign of scalar value indicates the rotation direction-positive

and negative values correspond to opposite rotational senses. The Helmholtz-Hodge decomposition is adopted to extract stream function from a 2D VF by following the equation below [38]. The decomposition and stream function calculation are given as follows,

$$\mathbf{U} = \nabla P + \nabla \times \psi + \mathbf{H} \tag{6.1}$$

$$\nabla^2 \psi = -\nabla \times \mathbf{U} \tag{6.2}$$

In the 2D formulation, the stream function ψ is a scalar field in 2D representation, which provides a convenient parameterization of the divergence-free component of the velocity field. The scalar field P is calculated from the divergence of velocity field. Under the common assumptions of a simply connected domain with no flux boundary conditions, the P and $\mathbf{H}$ vanish due to the incompressibility. Consequently, the decomposition reduces to the divergence-free component, which can be fully characterized by ψ. The resulting linear system is solved from the discretized Poisson formulation using the Conjugate Gradient (CG) method.

Streamlines provide an intuitive visualization of the instantaneous direction and structural patterns of a VF. In this study, streamlines are utilized as sketch data for training, serving as a compact representation of global flow behavior. To generate streamlines, particles are seeded at grid centers, and the trajectories are integrated by the Runge-Kutta method. This research select the streamlines originating from 512 cells exhibiting the highest velocity magnitudes, thereby capturing the main flow features. The training data pairs are organized as illustrated in Figure 6.2.

6.1.2 Two-Stage Latent Diffusion Model

The training framework is built upon a two-stage LDM, as illustrated in Figure 6.3. The sketch, stream function, and VF data are encoded into latent features via the auto-encoder, which are then used to train a denoising diffusion probabilistic model. This research used the basic auto-encoder [41] in the proposed framework due to its flexibility and adaptability. In the stream function generation stage, sketch input guides the stream function synthesis. However, due to the stochastic nature of the diffusion process, residual noise remains in the synthesized stream function. The noise propagates through the curl operator into the derived velocity field, potentially degrading the physical consistency of the generated flow. Hence, in the VF generation stage, an LDM is employed for synthesizing the velocity field from the given stream function to attenuate the effect of noise. In addition, the LDM encodes the control conditions into latent features, mapping different

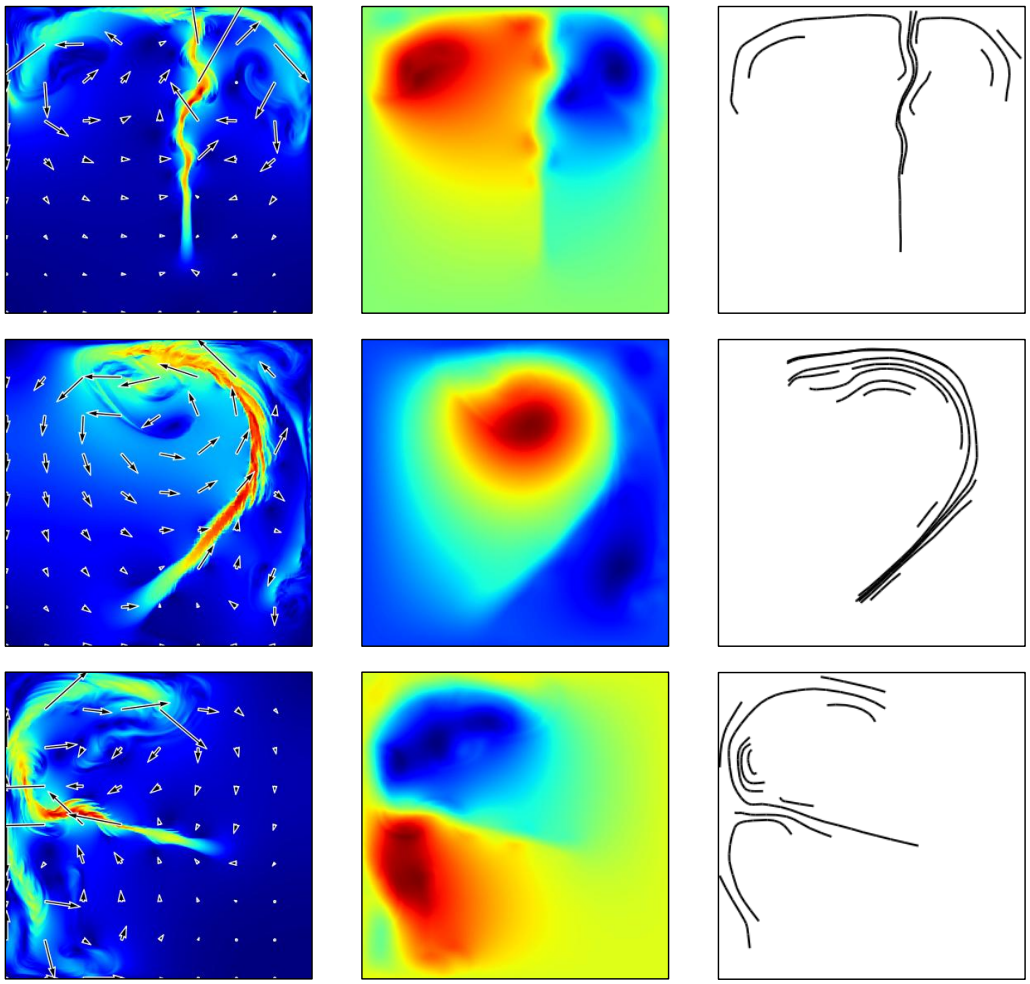

Figure 6.2: The left image is visualized VF, the middle image and the right image correspond to visualized stream function and streamlines.

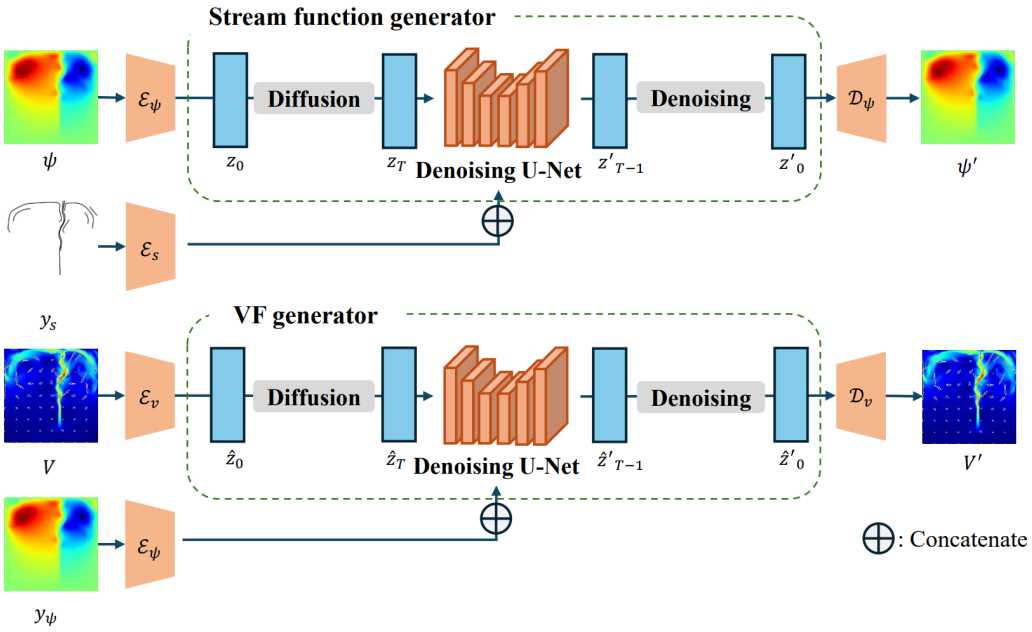

Figure 6.3: The training structure of the two-stage LDM generators. V is the velocity field. Sketch y_s and stream function y_ψ serve as control conditions. $\mathcal{E}_s$, $\mathcal{E}_\psi$, and $\mathcal{E}_\mathcal{V}$ are encoders. $\mathcal{D}_\psi$ and $\mathcal{D}_\mathcal{V}$ are decoders. All z and z' represent latent features.

types of information into the same latent space, thereby enabling efficient integration of information while reducing computational consumption.

6.2 Experiments

The training data was collected by conducting the incompressible smoke simulation with the Phiflow simulator [95]. The sketch autoencoder, stream function autoencoder, and velocity field (VF) autoencoder were trained for 500 epochs on an NVIDIA RTX 3090 using the Adam optimizer with a batch size of 16. The LDM training was conducted on an Nvidia A100 in a Linux environment. The dataset was shuffled and split in an 8:2 ratio, with 8,000 data allocated for training process and 2,000 for testing. In training process, the dataset was split in an 9:1 ratio for training and validation. Each LDM in the two-stage training network was trained for 100 epochs with the Adam optimizer, each epoch spent 7 minutes. The MSE loss was selected as the training loss for both the auto-encoder and the LDM. This research conducted a comparative analysis with DualSmoke [7] and DiffSmoke [10].

The compared methods were also trained for 100 epochs. The quantitative evaluation results are given in Table 6.1. The visualized result is given as

Methods	DualSmoke	DiffSmoke	Ours
MSE Loss (Intermediate variable generator)	0.050	0.031	**0.005**
MSE Loss (Velocity field generator)	0.572	0.052	**0.035**

Table 6.1: MSE loss comparison result. The first stage is the middle presentation generation. The second stage is velocity field generation. The generation accuracy of the proposed method is higher than that of both methods.

Figure 6.4 shows.

The DualSmoke model underperforms on the provided dataset, although the fluid spatial distribution was successfully captured, the field magnitudes were noisy. Although DiffSmoke successfully recovers the global flow topology, its reconstruction of fine-scale structures remains limited, with noticeable deviations in both local flow direction and velocity magnitude. The incompressibility of the generated velocity fields was evaluated by measuring the divergence loss. The proposed approach achieves a divergence loss of 0.014, whereas DiffSmoke yields 0.0143. The lower divergence indicates that the proposed generated velocity fields are closer to being divergence-free, thus better satisfy the incompressibility constraint. This improvement suggests that the proposed method more closely matches the underlying distribution of flow fields in the dataset, leading to enhanced physical consistency relative to the baseline. After velocity field generation, the synthesized velocity field was adopted as a guidance force field to drive the smoke simulation with the Mantaflow simulator [96]. The proposed framework allows users to input a hand-drawn sketch as the condition.

In addition, this research apply an anime-style transfer LLM interface after the simulation module for generating smoke illustrations. An animation stylization low rank adaption module (LoRA) [53] is trained for transfer operation. The LoRA training framework is given as Figure **??** shows. LoRA is a lightweight model fine-tuning module for enhancing the generation ability on specific contents or styles. LoRA aims to steer the generative model toward a particular visual style or content domain. An animation style smoke images dataset is required to train a LoRA module, along with corresponding text prompts. A large dataset is not necessary; indeed, an excessively large collection may dilute the influence of the prompts and reduce the controllability of the learned adaptation. The text prompts are obtained

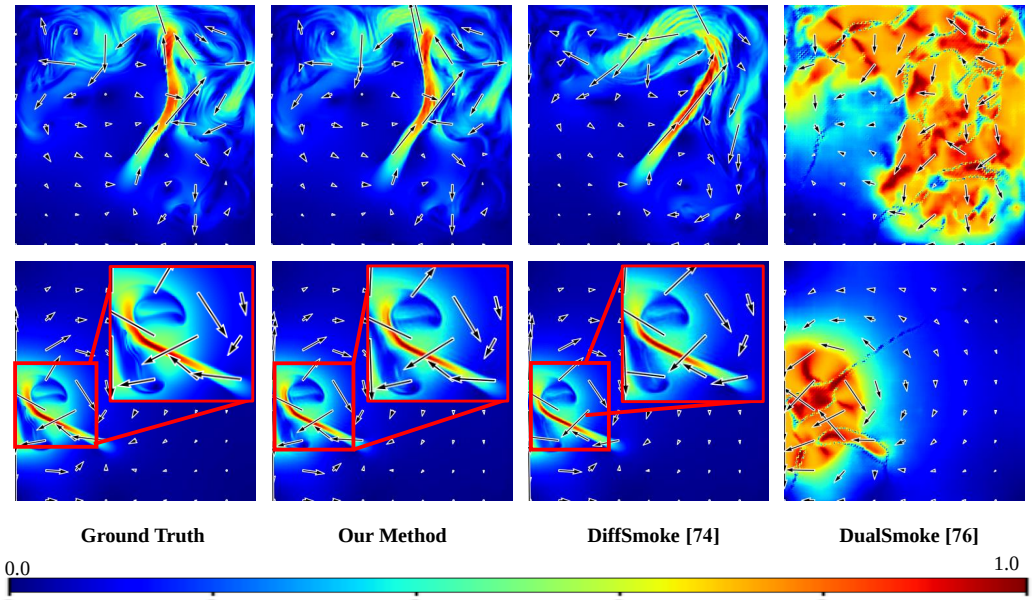

Figure 6.4: The qualitative evaluation result samples. The visualized velocity field is normalized into the [0, 1] range. The first column is ground truth, the second column is generated results by proposed method, the third column is generated results by DiffSmoke, the fourth column is generated results by DualSmoke. The images in red boxes are enlarged visualized generated velocity fields.

using a pretrained tagger or captioner. However, the resulting captions for smoke images are often overly verbose, which making them suboptimal for LoRA training. Therefore the text prompts is cleaned and simplified, and prepend a dedicated trigger token at the beginning of prompt sequences to reliably activate the LoRA during generation. In addition, smoke-relevant attributes such as color and stylistic descriptors are placed early in the prompt to strengthen conditioning on these key factors.

The style transfer is conducted under the guidance from text prompts. The results are given in Figure 6.6. In addition, the rendered density is stylized to a smoke illustration sequence as shown in Figure 6.7. By taking the rendered smoke illustration and text prompts that specify the style as inputs, the LLM successfully transfers the smoke illustration into the specified style while preserving the smoke shape. For style transfer on consecutive smoke illustrations extracted from simulations, the specified stylistic characteristics are generally preserved. However, the control over local details and background color remains inconsistent.

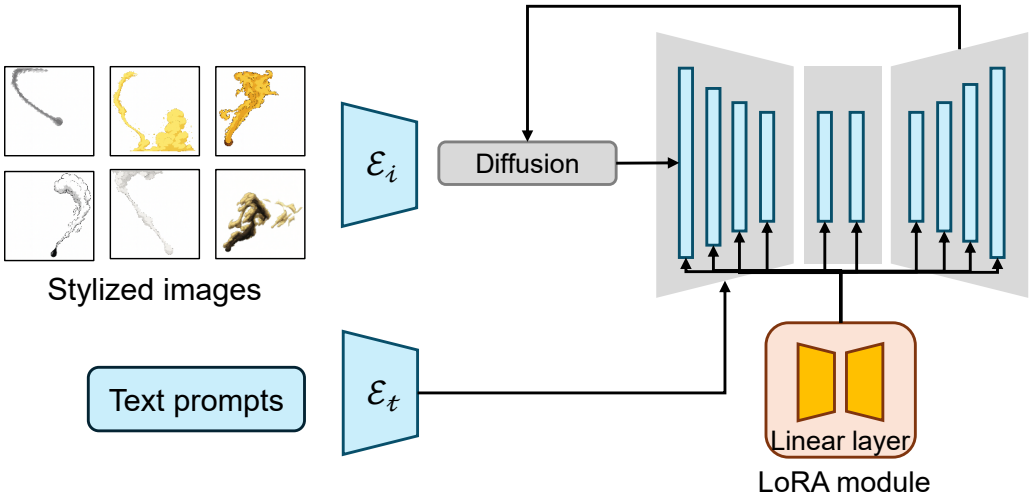

Figure 6.5: The example of anime style LoRA training framework.

6.3 Conclusion

In this research,a two-stage sketch-guided framework for smoke illustration generation using the stream-function is proposed. Experiments have demonstrated that the stream function has a stronger ability to capture motion behavior, as an intermediate representation connecting the sketch and the velocity field. Moreover, compared to the LCS region obtained through threshold filtering in the FTLE/hyper FTLE, the stream function has stronger physical consistency in the generation of constrained velocity field. However, the proposed method still faces several limitations. First, the diversity of fluid data patterns remains insufficient for synthesizing complex smoke illustrations such as humans and animals. The simulation was restricted to a rectangular domain without complex boundary conditions or obstacles inside the flow field. This research plans to incorporate complex-shaped solid obstacles into the simulation to facilitate the generation of intricate smoke patterns. Second, relying solely on MSE loss limits the efficiency of LDM. Incorporating physics-informed constraints, such as divergence-free condition, is essential for reducing noise and improving generation fidelity [55]. Third, the current generation framework generates only single-frame velocity fields, while the smoke illustration and video generation still rely on fluid simulation. This research aims to synthesize smoke videos from the input and control conditions via the video diffusion model, thereby simplifying the generation framework.

In future work, the research direction will focus on controlling the

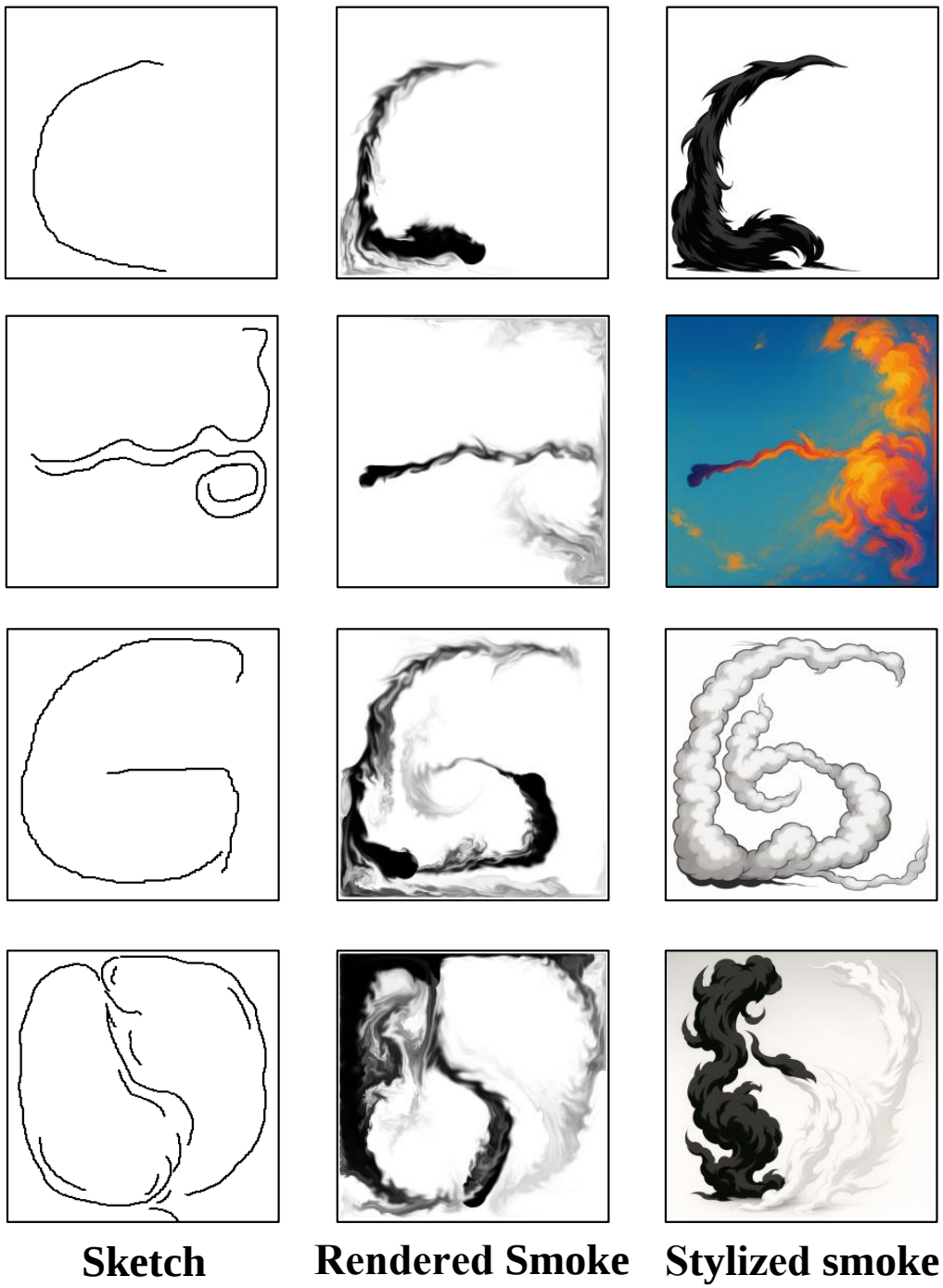

Figure 6.6: The anime style transferred smoke scene examples. The first column depict input hand-drawn sketches; the second column show the corresponding smoke illustrations; the third column display stylized smoke illustrations, which preserve the structural consistency of the original smoke illustrations while incorporating the target anime style.

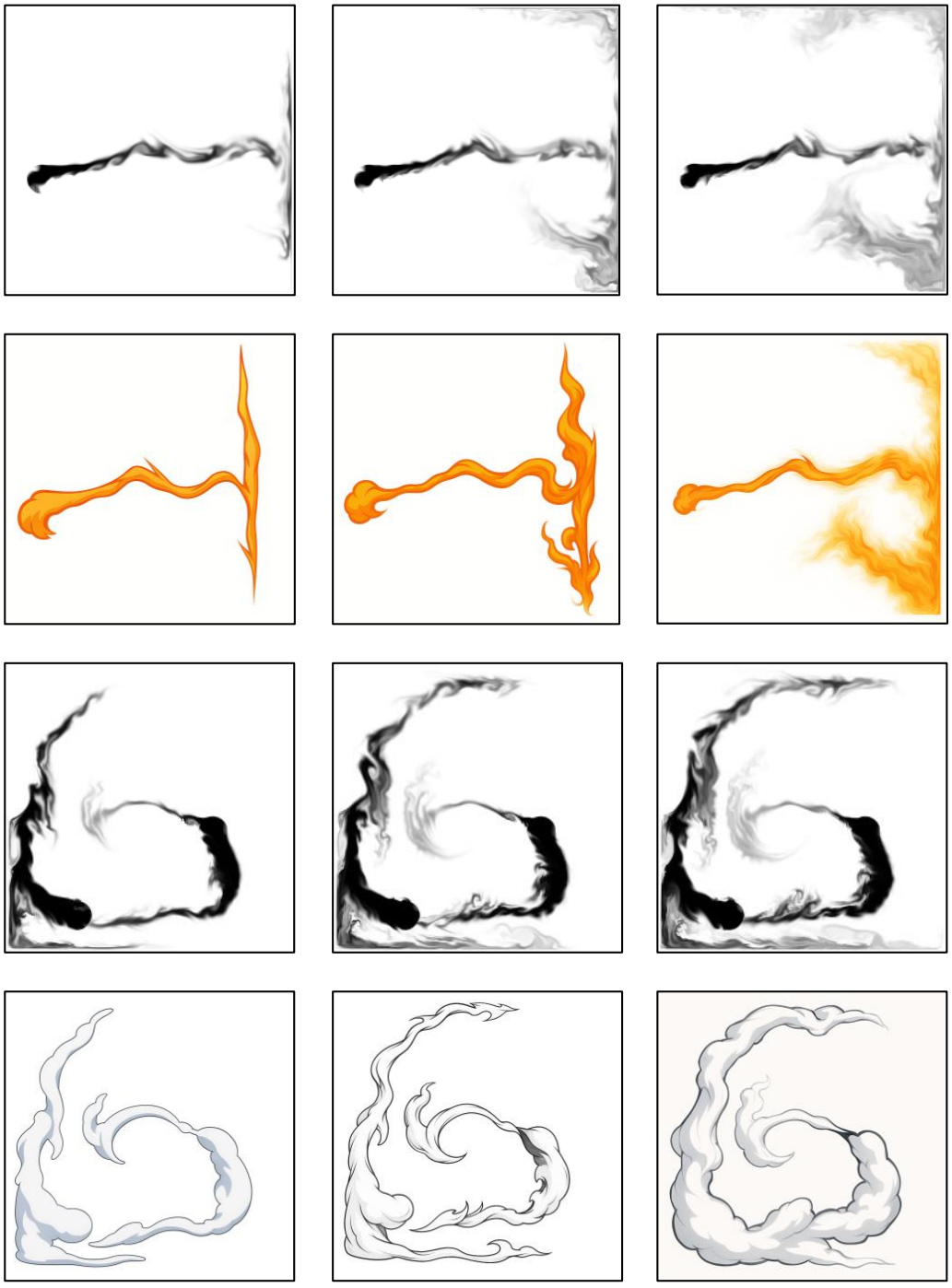

Figure 6.7: The anime style transferred smoke sequences examples. The first row presents the original smoke illustrations, while the second row shows the corresponding stylized versions. The first and fourth columns correspond to earlier stages in the sequence, while the third and sixth columns correspond to later stages.

generation of the rendered smoke density field sequence instead of a single velocity field by adopting the video generation diffusion model. In addition, drive the generation of stylized smoke motion videos with flow field motion information such as stream function is a feasible research direction. This idea is inspired by the experiment of style transfer on the rendered smoke density field. Current video generation models produce natural videos, and their control conditions are mostly pixel-based. Physical consistency is unable to be guaranteed in smoke video generations. Validating the usage of physics-based flow motion information for video motion control is a meaningful research direction. In the next chapter, this dissertation proposes a sketch-guided smoke video generation model using a stream function as the motion control condition. The effectiveness of the stream function in controlling the generation of smoke motion videos will be verified.

Chapter 7

Sketch-Guided Smoke Video Generation using Stream Function

This chapter proposes a sketch-guided smoke video generation framework using stream function as motion control condition. When generating videos of effects such as smoke, user control is typically expressed through textual prompts and limited image conditions, which offers low precision and cannot accurately reflect users' detailed creative intent. Moreover, existing motion-control methods for physical phenomena often rely on optical flow fields extracted from video as control signals. Optical flow represents pixel motion in a video, which lacks motion consistency of physical phenomena and fails to faithfully represent fluid dynamics.

To address these issues, this study proposes a sketch-guided smoke video generation framework based on a two-stage stream-function-guided latent video diffusion model. In the first stage, this study trains a stream function generator and a first-frame generator from pairs of sketches, stream functions, and rendered smoke density fields. In the second stage, the proposed video diffusion model is built upon the AnimateDiff [58] with an Object Motion Control Module [97]. The stream function is visualized first, then the stream function is encoded through motion control modules into multi-scale motion features, which are injected into the U-Net backbone as condition. For style control, this study further applies a pre-trained anime-style LoRA to transform the first smoke frame into a stylized illustration and then use the proposed video generation framework to propagate motion, achieving anime-style smoke videos driven by physical motion control. Experimental results demonstrate that the proposed method achieves improved quantitative results compared with previous studies. This chapter integrates methods proposed in the chapter 4, chapter 5 and chapter 6. The proposed method reduces the semantic gap and generation complexity between sketch and smoke data with intermediate representation. The stream function is used for smoke video motion control, which maintains physical consistency of motion information.

Generative models have democratized high-quality content creation, en-

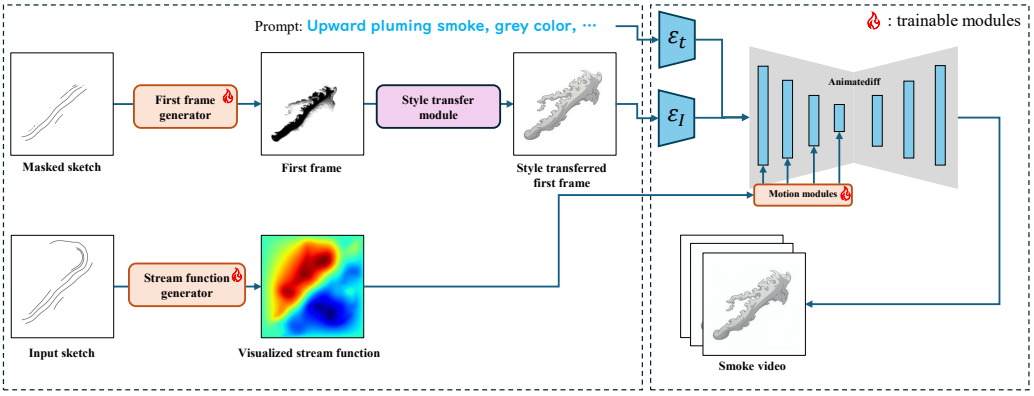

Figure 7.1: The framework of this study. $\mathcal{E}_t$,$\mathcal{E}_I$ represent encoders. The orange block represent trainable generator and motion module blocks; the blue blocks represent pre-trained frozen U-Net blocks; the purple block represents optional selected pre-trained style transferred block. The given text prompt is an example for showing how to set the keywords for generating smoke pattern and color pattern.

abling non-experts to synthesize various video contents including humans, objects, landscapes, and even physical phenomena. However, current text-driven video models lack precise control over local details in video generation. Motion-guided video generation research typically employs optical flow as the motion condition. The optical flow represents pixel movement features of video, which prioritizes the visual appearance and lacks underlying consistency while synthesizing physical phenomena. Sketches provides an intuitive yet precise approach to define scene layouts and motion trajectories with sparse line art. Integrating sketch-guided approach into video generation workflows allows creators to directly dictate fluid dynamics, simplifying the creation process and significantly enhancing the accessibility of smoke video synthesis.

In this study, a two-stage sketch−guided smoke latent video diffusion model is proposed. The proposed generation framework is built upon the latent video diffusion model (LVDM) [58]. As shown in Fig. 7.1, the sketch is used to condition the synthesis of the first video frame. In parallel, the sketch also guides the generation of a stream function, which is subsequently employed as a motion-control condition for smoke video synthesis. The text prompts, referenced first frame and motion condition are combined as the guidance of smoke video generation. During the training process, streamline sketches that capture the global dynamical structure of the flow are used as conditioning inputs, while the stream function serves as an intermediate

representation that complements these sparse sketches by providing additional motion cues that are not explicitly encoded in the sketch. In 2D fluid data, the stream function describes the fluid rotational motion and spatial structure. By solving for vorticity from the stream function, a divergence-free velocity field can be obtained, satisfying structural consistency while containing the overall motion information of the fluid. Therefore, this research is tend to use the stream function as the physical condition for motion control. Animation-style smoke images corresponding to the shape of a 2D fluid density field are generated from pre-trained stylized LoRA [53]. This allows for style control of the generated video. Experiments demonstrate that the anime-stylized fluid illustrations can be effectively influenced by motion control, thus enabling the generation of anime-style smoke videos.

7.1 Methods

This research proposes a sketch-guided smoke video generation framework using stream function to synthesize the artistic smoke video from the input sketches and text prompts. The whole generation process is given as 7.1 shows. The input sketch is converted into the stream function by the stream function generator, which subsequently drives motion control of smoke video generation by transmitting into the object motion control modules [58, 97]. The masked sketch gains from erasing input sketch locally for generating rendered density field at different frame. A optional choice is applying the style transfer for generating artistic stylized smoke videos. Text prompt is used for setting the generation to smoke pattern and video color information control.

7.1.1 Smoke Data Collection

The velocity fields used in this research are obtained from incompressible inviscid 2D smoke simulations, following the standard grid-based formulation in [76]. Given that the smoke simulation occurs within a rectangular obstacle domain, where the boundary is regarded as solid. The boundary condition enforces the confinement of the smoke simulation within the defined boundaries. The semi-Lagrangian scheme [12] updates the velocity field. Two types of forcing configurations are considered in this research. In one subset of simulations, the velocity field is driven by oblique wind forces. In the other subset, the flow is influenced by a combination of a global Perlin-noise-based force field and an upward wind field, which together prevent the smoke from moving in the direction of gravity. The smoke source is set

in the areas with high velocity field magnitudes. The velocity field drives the smoke movement with the Primal-Dual optimization method [98]. The vorticity confinement method is applied to enhance the smoke rotational details. The time step must be chosen below an upper bound: an excessively large time step amplifies numerical errors at each update and can eventually destabilize the simulation. Likewise, the magnitude of the external force $\mathbf{f_e}$ is also constrained, as overly strong forcing can lead to unrealistic distortions in the smoke motion.

7.1.2 Stream Function and Streamline Collection

The stream function is a latent representation of the rotation information continuous dynamics feature of the velocity field. The positive and negative values indicate opposite directions of rotation. The Helmholtz-Hodge decomposition is adopted to extract stream function from a 2D velocity field by following the scheme from [38]. Streamlines intuitively depict the instantaneous direction and structure of a velocity field. Hence, streamlines are employed as sketch data for training in this study.

The streamlines originating from 200 cells with the highest velocity magnitude was selected. This operation ensures the main movement structure of velocity field will be kept and decreases the disturbance from the low velocity field area. The stream function include the rotational information details, the positive value represents rotate direction, the negative value represents rotate direction.

First Frame and Sketch Data Pair The movement of smoke can be viewed as a gradual moving until a steady state is reached, visually presenting a self-circulating state. Therefore, the shape and movement trend of smoke differs at different time frames. To address this issue, streamlines describing global motion are first extracted from the global velocity field, and then the density field is used as a mask to occlude these streamlines, obtaining corresponding sketches at different time frames. The sketch and smoke first frame data pair examples in one simulation scenario is given in the Figure7.2. The sketch and smoke first frame data pair examples in different simulation scenarios are given in Figure7.3. 20 frames of smoke density fields are randomly selected from each video slice to construct the data pairs and ensure the diversity of dataset.

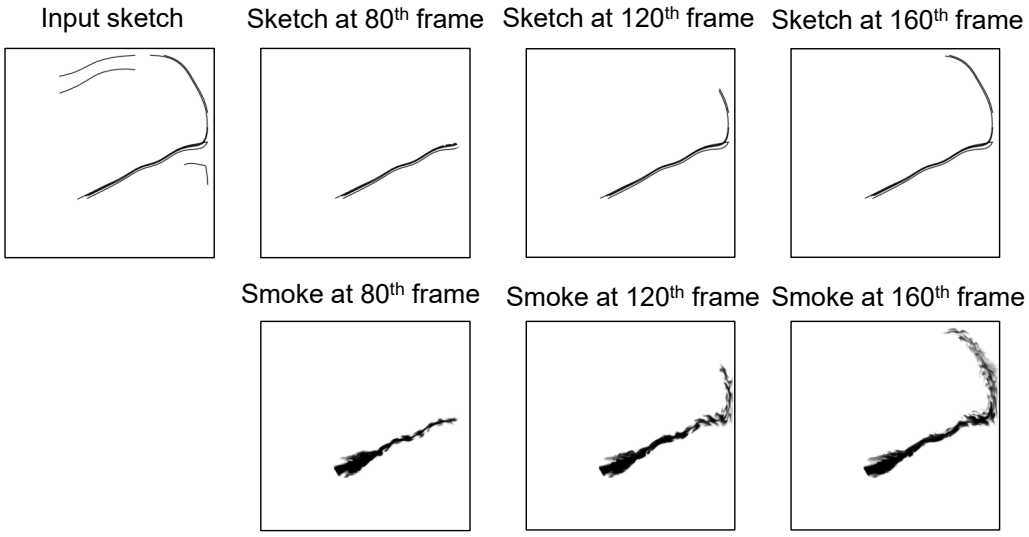

Figure 7.2: Samples of sketch-density data pairs with a given specific sketch. The first and the fourth column is the original input sketch. The second and the fifth column is the masked sketch. The third and the sixth column is the paired smoke density field.

7.1.3 Latent Video Diffusion Generation with Object Motion Control Module

The AnimateDiff framework is adopted as the base video diffusion model, providing a robust backbone for learning temporally coherent motion patterns from text and image inputs. The introduction of AnimateDiff framework is given as Figure 7.4 shows. Building upon the idea of using optical flow fields as target object motion control condition (OMCM) in video generation, as proposed in [97], the extracted stream function is reinterpret as a visualized image data for explicit control the smoke movement over the temporal sequences. The motion control structure is given as Figure7.5 shows. Concretely, the visualized stream function is encoded by OMCM into a hierarchy of multi-scale feature maps, which are then injected into the U-Net backbone. The stream function are mapped with rainbow colormap. The red color means positive value, the blue color means negative value. These colors implicitly determines the rotaional direction locally on the visualized stream function. The core idea of motion control is combing motion condition blocks with down sampling blocks for transmitting the motion information into latent space. Motion control modules serves as an encoder to compress the stream function. The encoded stream function condition is additively

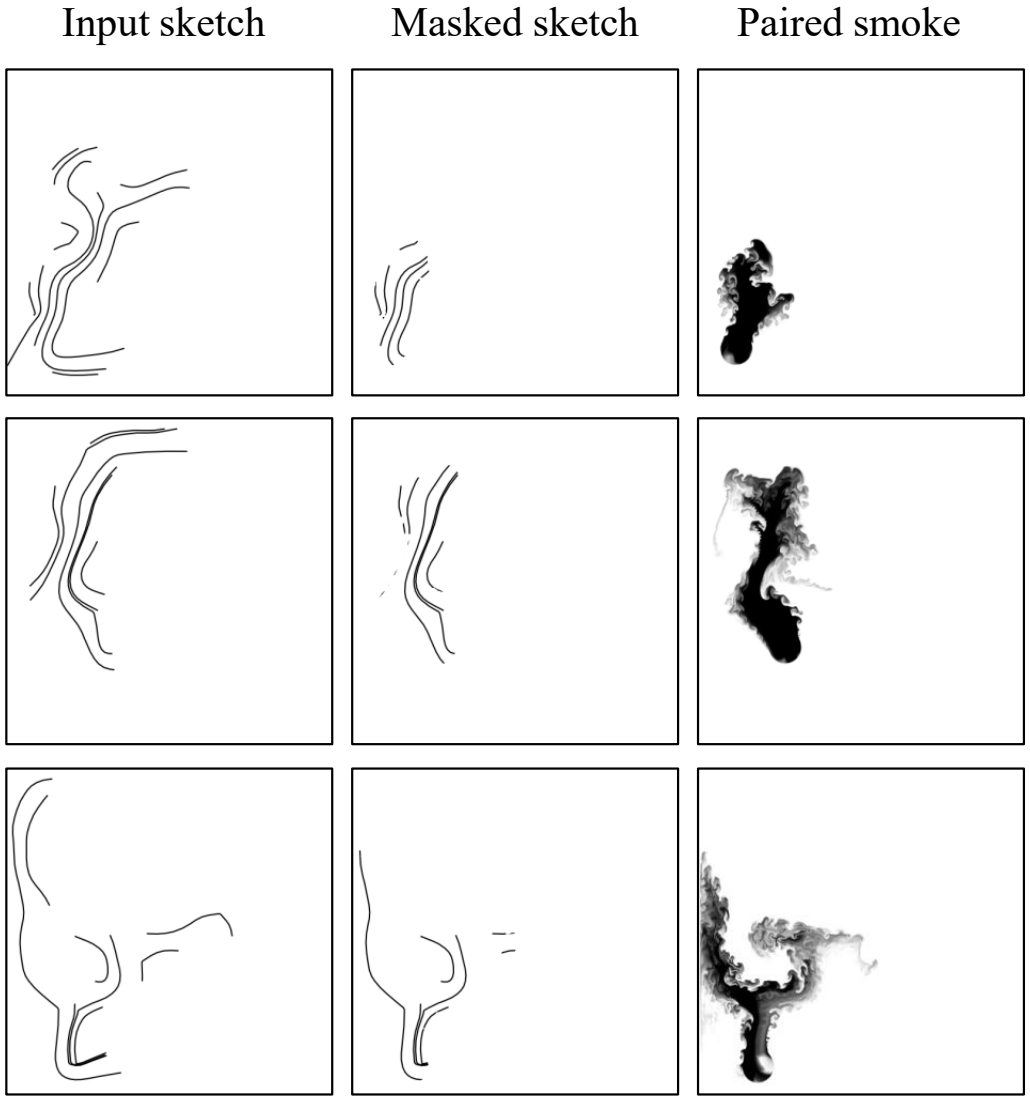

Figure 7.3: Samples of sketch-density data pairs. The first and the fourth column is the original input sketch. The second and the fifth column is the masked sketch. The third and the sixth column is the paired smoke density field.

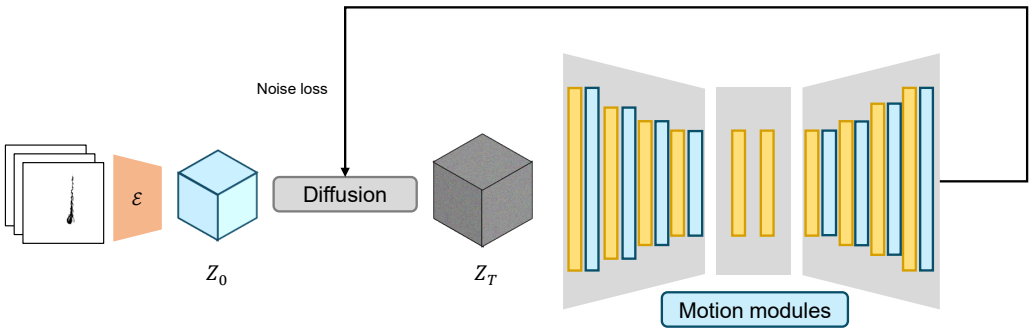

Figure 7.4: The structure of AnimateDiff.

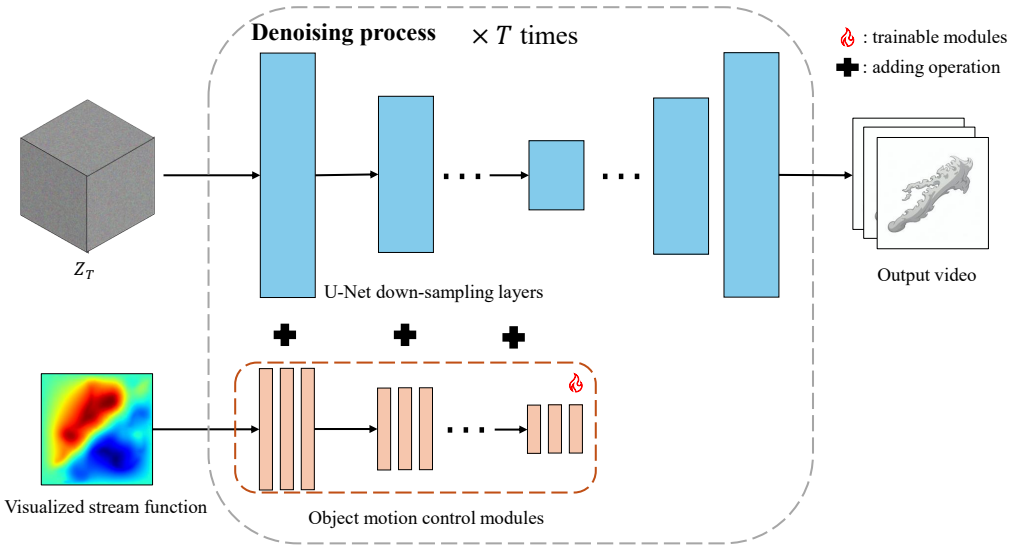

Figure 7.5: The structure of OMCM. This structure merge the latent motion features with the latent video features during the denoising process. The OMCM is added onto the down-sampling layers of U-Net backbone. The orange blocks are trainable OMCMs. The blue blocks are frozen U-Net. The dimension size of convolution layer decreases gradually.

fused with the down sampling latent features. The merged new motion-informed latent features guides the generated smoke video to follow the prescribed flow pattern.

The loss function of SLVDM is given as follows:

$$L_{SLVDM} = \mathbb{E}_{\mathcal{E}(x),y,s,\psi\epsilon\ \mathcal{N}(0,1),t}[||\epsilon - \epsilon_\theta(z_t, t, \mathcal{E}_{text}(y), \mathcal{E}_\theta(s), \tau_{omcm}(\psi))||_2^2] \quad (7.1)$$

where $\mathcal{E}(x)$ is encoded feature map, $\mathcal{E}_\theta(y)$ is encoded text prompt condition, $\mathcal{E}_\theta(s)$ is encoded first frame condition, $\tau_{omcm}(\psi)$ is encoded motion condition, $\epsilon_\theta(..., t)$ is neural backbone.

7.1.4 Generator Training

Before training the video generation model, first-frame generator and stream function generator are required to train. The generator structure is given as Figure7.6 shows. The input sketch auto-encoder, masked sketch auto-

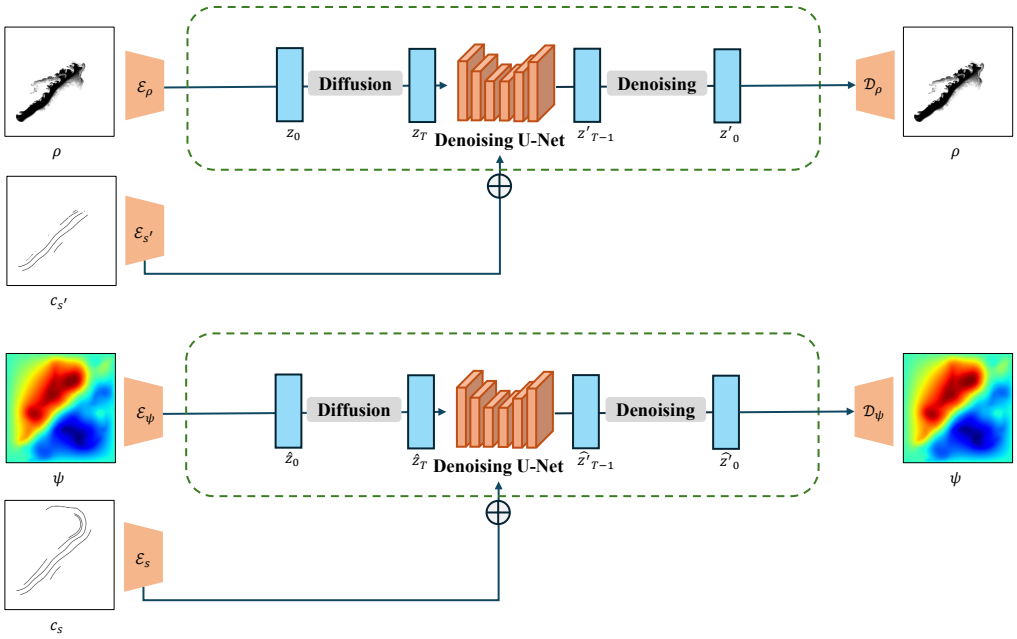

Figure 7.6: The training structure of the two-stage LDM generators. ρ is the rendered smoke density field. Input sketch c_s, masked sketch c'_s serve as inputs of the generator. $\mathcal{E}_s$, $\mathcal{E}_\psi$, and $\mathcal{E}_\rho$ are encoders. $\mathcal{D}_\psi$ and $\mathcal{D}_\rho$ are decoders. All z, z', $\hat{z}$ and $\hat{z'}$ represent latent features. z and $\hat{z}$ exists in different latent spaces.

encoder and density field auto-encoder requires to be trained before generator

training. In the training process, the encoded stream function and density field feature adds Gaussian noise in each diffusion step. The encoded sketch condition is injected into each U-Net for the denoising process control. In the inference process, the sketch condition recovers the target visualized stream function and smoke density frame from noise.

7.2 Results

7.2.1 Experiments

The video diffusion model training was operated on an Nvidia H100, Linux system. A total of 580 smoke simulation videos was collected, each video has 320 frames. This study also prepared the corresponding stream function sequences and sketch sequences. Each stream function and sketch aligns with the corresponding video frame. The Mantaflow was used to finish the simulation; each video requires at least 2 minutes. In order to make sure the variety of the smoke movement, the video clips is selected from the 80th frame to the 240th frame. If the frame index is too small, the smoke density is too sparse for training. If the frame index is too large, the movement of smoke is significantly reduced. The training of stream function generator took 25 hours for 100 epochs. The training of first frame generator took 25 hours for 100 epochs. The base model for training video diffusion model is Stable Diffusion v1.5 [51]. The smoke video diffusion model were implemented with Diffusers [99] based on motionCtrl [97]. The resolution of training data was set to 384×384. The sampling frame was set to 17, which each 8 frames per second. The training of the proposed framework took 8 hours for 50 epochs.

7.2.2 Evaluation

This study compares the generation accuracy with VidSketch [100] and animatediff [58]. The evaluation results are given in the table below. This study chooses FVD [101], LPIPS [102], SSIM [103], and PSNR [104] as the evaluation schemes. The result shows as Table 7.1. The proposed

Table 7.1: Quantitative comparisons for generated smoke video quality.

Method	FVD↓	LPIPS↓	PSNR↑	SSIM↑
VidSketch	159.6444	0.1286	**19.7962**	0.8933
AnimateDiff	665.9049	0.4087	10.0158	0.7303
Ours	**149.5317**	**0.0887**	18.7597	**0.8980**

method consistently outperforms the baseline models across FVD, LPIPS, and SSIM metrics. In contrast, PSNR is the only metric on which the proposed approach falls short compared to the prior work. In addition, this study compared standard deviations (std) of SSIM, PSNR, and LPIPS to quantify performance variability across the test set. The proposed framework consistently yields the lowest standard deviation for all three metrics, indicating the generation quality of proposed framework is more stable and less sensitive to changes in input content or scenes. This reduced variance suggests that the proposed method generates more reliable outputs and exhibits fewer extreme failure cases compared to prior approaches. The result shows as Table 7.2

Table 7.2: Standard deviation comparisons for video generation.

Method	LPIPS std	PSNR std	SSIM std
VidSketch	0.13069	3.5088	0.072693
AnimateDiff	0.12060	2.3379	0.09929
Ours	**0.03807**	**1.5279**	**0.05685**

The visualized result is given as Figure7.7 shows. The output of VidSketch still being unstable after training for 50 epochs, The background of generated video flickers occasionally, and the generated smoke is fragmented. The output of AnimateDiff has a noisy background, the noise around the smoke and at the edges of the video is quite noticeable. Instead, the proposed framework generates stably moved smoke patterns after training the same epochs compared with previous research.

In addition, this study chooses cartoon style transferred smoke image as the referenced first frame. The samples of rendered smoke density and style transferred smoke data pairs are given in Figure 7.8. The style transfer was conducted by introducing a pre-trained animation stylized smoke generation LoRA based on SDXL model [105].

The generation result is shown in Figure7.9. The result shows that even for the style-transferred smoke first frame, the motion control module is able to transfer the motion information from smoke density field into a smoke video with different styles. However, the color information of input smoke first frame fails to be preserved. The overall color of the generated video tends to resemble the rendered smoke density field of the used for training.

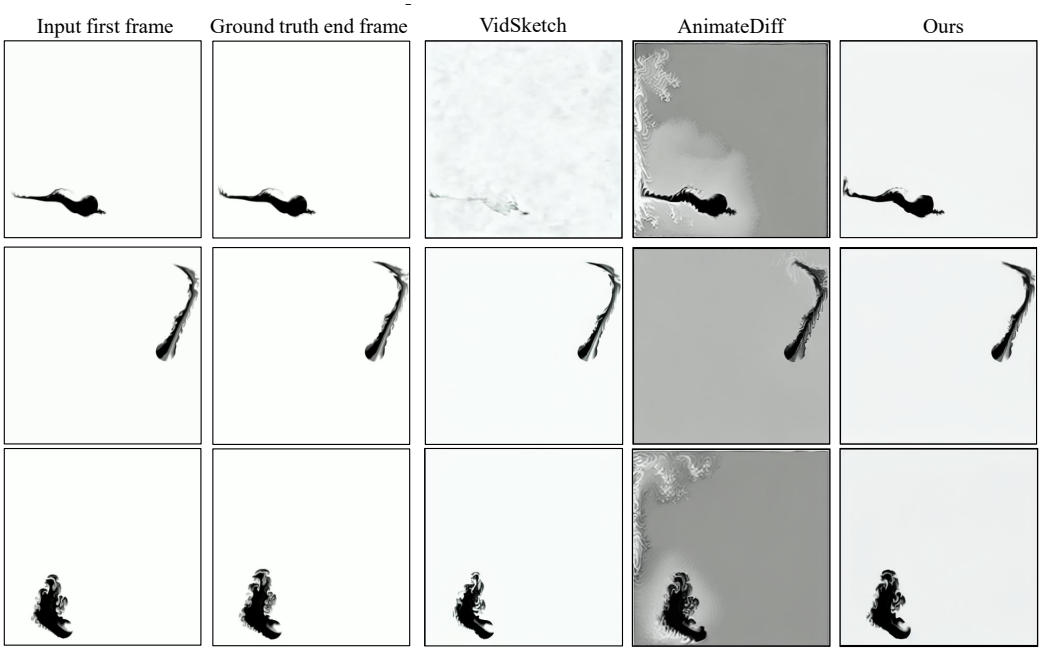

Figure 7.7: The visualized comparison results. The first column is input first frame. The second column is the extracted frame from the ground truth video. The third column is the generation result of the extracted frame by VidSketch. The fourth colum is the generation result of the extracted frame by AnimateDiff. The fifth column is the generation result of extracted frame by proposed method.

7.3 Conclusion

This research generates a visual stream function and the initial frame of the smoke video from a semantic sketch. By using the visualized stream function as a motion control condition, a rendered smoke motion video is successfully generated while maintaining the accuracy of the smoke motion. Furthermore, this paper transforms the first smoke frame into an anime style using a pre-trained style transfer LoRA, and successfully generates an anime style smoke motion video through a motion module.

This research has the following limitations. First, the U-Net-based video generation framework is computationally expensive. Long video segments is unable to sample for training when the data is high-resolution. Second, the framework lacks control over the color consistency of the first input frame, making the generated results highly dependent on the dataset. Third, although the stream function is highly sensitive to rotational information in

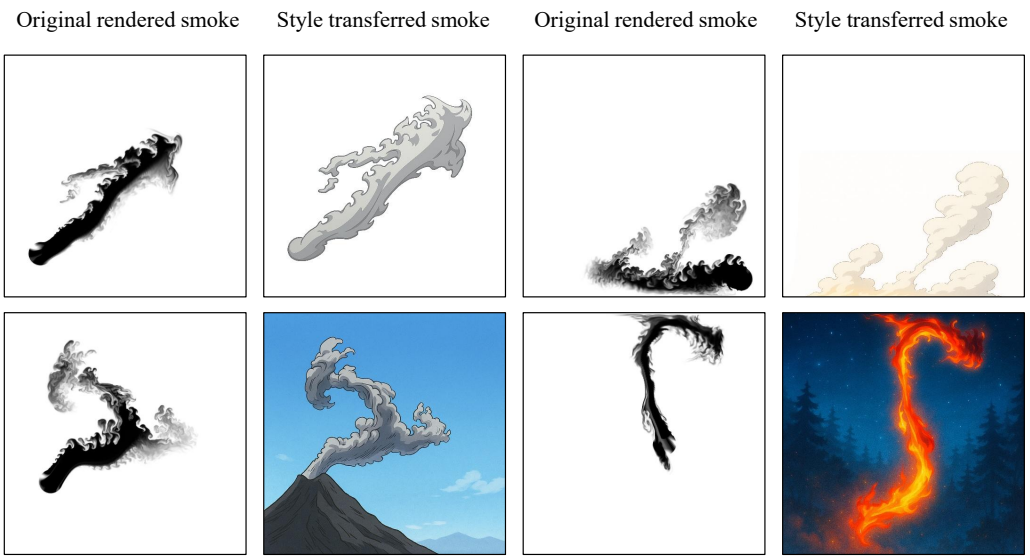

Figure 7.8: Samples of style transfer result. The first and the third column represents the rendered smoke, the second and the fourth column represents the style transferred smoke. The first row is generated within blank background. The second row is generated within a complex background.

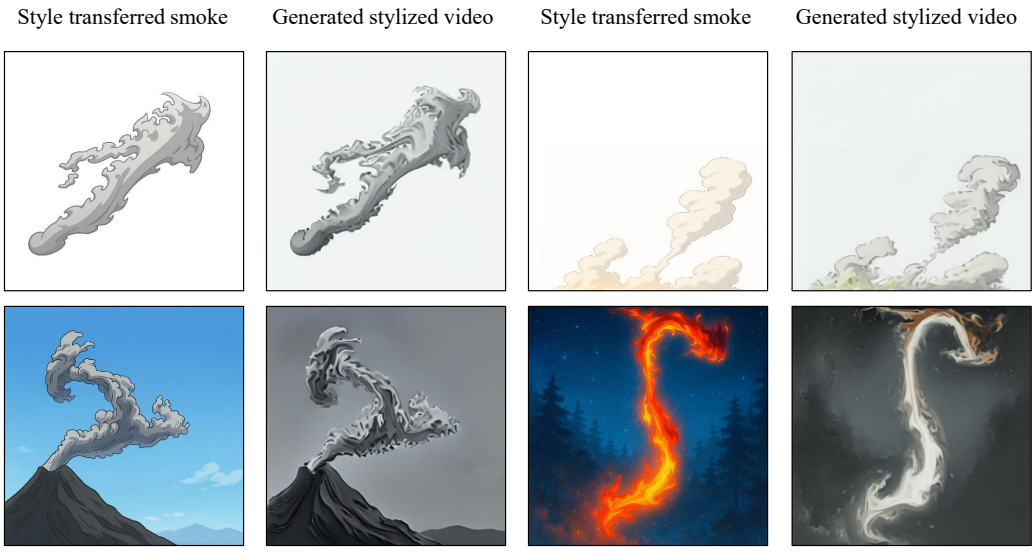

Figure 7.9: Samples of style transferred smoke video generation. The first and the third column represents the stylized smoke, the second and the fourth column represents the final frame of generated video. Although the motion information has been successfully attached, the color information is lost during the training process.

the velocity field, motion information including velocity potential existing in the harmonic field fails to be captured by the stream function. In the future, we first plan to render the smoke density or particles with different artistic styles, thereby enhancing the robustness of stylized smoke video generation. Secondly, it is expected to use a DiT-based video generation model to improve the length and accuracy of the generated videos. Building upon previous chapters, this chapter utilizes the idea of decomposing complex generation tasks and employs stream function for smoke motion video generation. The proposed method reduces the semantic gap between the sketch and the smoke field, the physics-based control condition ensures the physical plausibility of the motion.

Chapter 8

Conclusion

This dissertation combines generative models with hand-drawn sketches and traditional fluid creation, provide a sketch-guided data-driven 2D smoke effect generation system to decrease the complexity in mapping between sketch and smoke effect. In details, this dissertation first follows a direct generation of velocity fields whose shape and flow direction are consistent with a user-drawn sketch for smoke effect design. This research first attempts to use the LDM to learn the mapping between sketches and velocity fields. By comparing our approach with prior GAN-based velocity-field generation methods, the stability of diffusion models for learning the mapping between sketch and velocity is demonstrated. Building upon this, the complex mapping from sketch to velocity field is decomposed into a two-stage sub-mapping to decrease the ambiguity of velocity field generation. The LCS region is chosen to serve as the intermediate presentation for connecting the mapping between sketch and velocity. The LCS derived from hyper-FTLE provides a integrated spatial representation of flow field structure over defined temporal intervals, serving as control condition. Compare with one-stage model, two-stage model generates more precise velocity field by leveraging regional-based control condition as explicit constraints. Furthermore, the stream function is introduced for further improving the velocity field generation consistency. Stream function contains both the rotation dynamics features and structural information of the velocity field, while LCS region is a binary mask for constraining the velocity field. Finally, the input sketch serves as the guidance for the smoke video synthesis. The pipeline begins by generating the video first frame from the sketch, while simultaneously deriving a stream function that is integrated into the motion control module as a motion constraint. The proposed framework facilitates cartoon-style video generation by applying style transfer to the initial frame. Notably, the stream function motion priors successfully drives the stylized smoke movements. The quantitative analysis is conducted by comparing the proposed framework with prior research, demonstrating the effectiveness of the proposed framework.

In summarization, our research successfully combines physical plausible

smoke effect with semantic sketch input through a diffusion model, providing a feasible solution for non-professional users to create fluid illustrations and videos. By controlling the geometry and motion characteristics of the flow field through physically based intermediate representations, the physical accuracy of generated smoke effect is maintained while preserving the user creative intent.

The proposed studies reduce user interaction complexity at the input stage. In conventional 3D modeling software, users typically create 3D curves using dedicated modeling tools, which are then algorithmically converted into a relatively coarse force field. A further refinement is commonly performed through parameter-control interfaces or programming to edit the force field. In contrast, standard fluid simulation tools often require users to possess prior knowledge in fluid dynamics as well as programming skills to design force fields that control smoke motion. In our framework, users draw simple line strokes using a built-in painting tools on a computer system by dragging painting brush with a mouse. The resulting sketch is provided as input to our pipeline, which automatically generates a corresponding velocity field. This velocity field is then integrated into our implemented fluid simulation script to produce the final smoke simulation results.

8.1 Remaining Challenges and Future Works

This dissertation lacks further refinement of hand-drawn sketches. Since the sketches lacks information indicating the direction of motion. The diversity in motion direction guidance is insufficient. Furthermore, this dissertation achieves simple control over the smoke motion through global external forces and solid boundary conditions outside the flow field. However, more detailed control over the flow field lacks in the simulation, resulting in relatively simple smoke patterns. For complex sketch inputs such as human shapes and animals, The velocity field generation fails to preserve structural characteristics when the input consists of intricate sketches. Examples of this issue are shown in Figure 8.1, where complex sketches, such as a human figure and a swirl, are incorrectly recognized as flow plumes in the velocity field. Therefore, developing a more comprehensive sketch-to-velocity dataset would be a feasible improvement for this dissertation.

During data collection, the simulated smoke exhibits relatively fast motion, leading to noticeable differences between consecutive frames. However, sampling velocity fields from each simulation at a fixed temporal interval can still introduce locally continuous sequences into the dataset. Such temporal correlations may bias the training process by over-representing

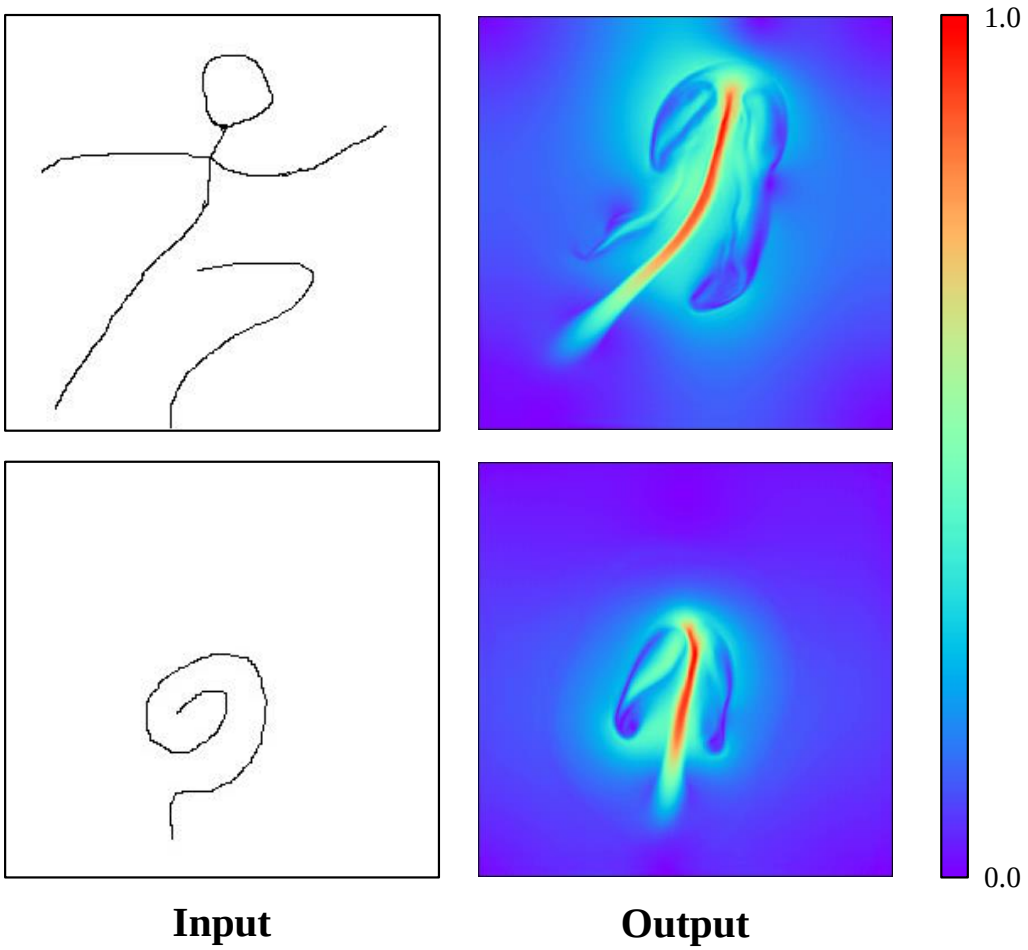

Figure 8.1: The examples of failure results. The first column shows the input hand-drawn sketches, while the second column visualizes the corresponding output velocity fields. In given instances, the input sketches generate velocity fields that mismatch the given sketch shapes.

highly similar samples and thus affect the learnable parameters. In the experiments, when the model on a dataset that contained these consecutive sequences is evaluated, the MSE of the generated results was 0.007 lower on average than that obtained using a randomly shuffled dataset. This suggests that data redundancy can artificially inflate quantitative performance and may not reflect true generalization. Hence, Future work will increase both the diversity and the number of simulation scenarios, and avoid fixed-step sampling during dataset construction. Instead, sampling strategies that reduce temporal redundancy will be adopted, thereby preventing the collection of highly similar consecutive samples.

Current studies have not yet developed a complete interactive user interface that encapsulates the pipeline from sketches to velocity-field generation. At present, the generated velocity fields are primarily integrated into fluid simulation libraries such as PhiFlow [95] and Mantaflow [96], where the velocity serve as guiding velocity inputs to control the simulated flow motion. These tools provide limited support for graphical user interfaces and are mainly intended for Python-based simulation programming. The future work plans to wrap and extend these simulation backends with a more intuitive visualization and interaction interface, thereby reducing user effort and lowering the design cost.

This dissertation focuses primarily on incompressible inviscid smoke phenomena. Future plans include developing a more versatile fluid creation system, the viscid fluid, fluid-structure interaction, and multiphase flow will be considered. For example, when the training data are extended from incompressible inviscid flows under no-penetration solid-wall boundary conditions to more complex fluid regimes, such as viscous flows and flows with free-slip boundaries, the explicit boundary conditions will be incorporated as controllable condition during model training. Concretely, the training samples with additional inputs can be augmented that encode boundary information, such as boundary mask or signed distance field (SDF), and provide them as extra conditioning channels. In addition, the boundary type and physical parameters can be specifically annotated such as the viscosity coefficient. During training process, these additional physical descriptors can be injected into the generative model through mechanisms such as cross-attention module, enabling the network to adapt its generation to different boundary configurations and fluid properties. Furthermore, additional physics-informed loss terms will be introduced, such as divergence loss for incompressibility and boundary-condition losses for enforcing velocity constraints at boundaries. These loss functions can be used to fine-tune the training model.

In this dissertation, the generation pipeline is decomposed into more

interpretable sub-processes by introducing explicit physics-based intermediate variables. However, the current network architecture does not include dedicated modules that are specifically tailored to data with strong physical priors. In future work, physics-informed generation frameworks is aimed to be proposed that further integrates physical priorities and constraints. Related LDM-based research typically incorporate additional modules or functional transformations to derive auxiliary physical information from the target quantities, which is then used either as an extra conditioning signal or as part of the training objective. For example, an advection residual is regarded as an additional loss term to impose physics-informed constraints on training model. LDM-based frameworks first encode data into a latent space and learn the generative process in this compressed representation. As a result, the latent features are not guaranteed to preserve or satisfy the physical constraints that exist in the original data domain. Physics-based explicit control is challenging to impose directly on latent representations within LDM-based frameworks. Moreover, the training data are typically natural images or videos for most generation tasks, from which reliable physical quantities are not readily extractable. To compensate for this limitation, some studies control the generation with textual prompts that describe implicit physical phenomena, thereby constraining the generated dynamics in a semantic space. In addition, a practical strategy is to collect high-physical-confidence data from physical experiments or physics-based simulators, and use them for optimization or fine-tuning the model.

Current generative models focuses excessively on the quality of the generated result, while designing is a continuous process of editing. Inspired by two-stage generative models, The goal of multi-stage models should not be limited to merely generating control conditions that govern the final result. Multi-stage models can also be integrated into the creative process itself. By editing and modifying intermediate expressions, the combination of generative models and creative processes can be further improved, allowing user intents to align more accurately with the creative workflows of professional designers.

An additional main focus will be on generating 3D fluid materials from 3D sketches. The 3D fluid provides crucial references for industrial design, which extends beyond multimedia. Currently, the sketches are primarily 2D. 3D sketch-guided 3D flow field generation will be considered in the future. Intermediate representations such as LCS and stream functions have corresponding 3D features, allowing for accurate control of 3D fluid geometry and motion features through a two-stage approach. A strategy for painting 3D sketches conveniently will also be considered.

References

[1] J. Schpok, W. Dwyer, and D. S. Ebert, "Modeling and animating gases with simulation features," in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2005, pp. 97–105.

[2] B. Zhu, M. Iwata, R. Haraguchi, T. Ashihara, N. Umetani, T. Igarashi, and K. Nakazawa, "Sketch-based dynamic illustration of fluid systems," vol. 30, no. 6. New York, NY, USA: Association for Computing Machinery, dec 2011, p. 1–8. [Online]. Available: https://doi.org/10.1145/2070781.2024168

[3] 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.

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," pp. 1125–1134, 2017.

[5] 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*, vol. 30, 2019.

[6] G. Yan, Z. Chen, J. Yang, and H. Wang, "Interactive liquid splash modeling by user sketches," *ACM Trans. Graph.*, vol. 39, no. 6, nov 2020. [Online]. Available: https://doi.org/10.1145/3414685.3417832

[7] H. Xie, K. Arihara, S. Sato, and K. Miyata, "Dualsmoke: Sketch-based smoke illustration design with two-stage generative model," *Computational Visual Media*, pp. 1–15, 2024.

[8] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016.

[9] H. Chang, Y. Peng, S. Sato, and H. Xie, "Sketch-guided flow field generation with diffusion model," in *International Workshop on Advanced Imaging Technology (IWAIT) 2024*, vol. 13164. SPIE, 2024, pp. 290–295.

[10] H. Chang, T. Zhang, S. Sato, and H. Xie, "Diffsmoke: Two-stage sketch-based smoke illustration design using diffusion models," *IEEE Access*, vol. 13, pp. 44 997–45 009, 2025.

[11] H. Chang, X. Xie, S. Sato, and H. Xie, "Two-stage sketch-based smoke illustration generation using stream function," in *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters*, 2025, pp. 1–2.

[12] J. Stam, "Stable fluids," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '99. USA: ACM Press/Addison-Wesley Publishing Co., 1999, p. 121–128. [Online]. Available: https://doi.org/10.1145/311535.311548

[13] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual simulation of smoke," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 15–22.

[14] W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Crespo, V. Leroy, and Z. Zhang, "Mitsuba 3 renderer," 2022, https://mitsuba-renderer.org.

[15] J. J. Monaghan, "Smoothed particle hydrodynamics," *In: Annual review of astronomy and astrophysics. Vol. 30 (A93-25826 09-90), p. 543-574.*, vol. 30, pp. 543–574, 1992.

[16] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *Journal of Visual Communication and Image Representation*, vol. 18, no. 2, pp. 109–118, 2007.

[17] M. Macklin and M. Müller, "Position based fluids," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–12, 2013.

[18] D. Sulsky, Z. Chen, and H. L. Schreyer, "A particle method for history-dependent materials," *Computer methods in applied mechanics and engineering*, vol. 118, no. 1-2, pp. 179–196, 1994.

[19] D. Ram, T. Gast, C. Jiang, C. Schroeder, A. Stomakhin, J. Teran, and P. Kavehpour, "A material point method for viscoelastic fluids, foams and sponges," in *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2015, pp. 157–163.

[20] C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin, "The affine particle-in-cell method," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–10, 2015.

[21] A. Treuille, A. McNamara, Z. Popović, and J. Stam, "Keyframe control of smoke simulations," in *ACM SIGGRAPH 2003 Papers*, 2003, pp. 716–723.

[22] A. McNamara, A. Treuille, Z. Popović, and J. Stam, "Fluid control using the adjoint method," *ACM Transactions On Graphics (TOG)*, vol. 23, no. 3, pp. 449–456, 2004.

[23] R. Fattal and D. Lischinski, "Target-driven smoke animation," in *ACM SIGGRAPH 2004 Papers*, 2004, pp. 441–448.

[24] L. Shi and Y. Yu, "Controllable smoke animation with guiding objects," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 1, pp. 140–164, 2005.

[25] G. Zhang, D. Zhu, X. Qiu, and Z. Wang, "Skeleton-based control of fluid animation," *The Visual Computer*, vol. 27, no. 3, pp. 199–210, 2011.

[26] B. Yang, Y. Liu, L. You, and X. Jin, "A unified smoke control method based on signed distance field," *Computers & graphics*, vol. 37, no. 7, pp. 775–786, 2013.

[27] Y. Chen, D. Levin, and T. Langlois, "Fluid control with laplacian eigenfunctions," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11.

[28] Y. Kim, R. Machiraju, and D. Thompson, "Path-based control of smoke simulations," in *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2006, pp. 33–42.

[29] A. Angelidis, F. Neyret, K. Singh, and D. Nowrouzezahrai, "A controllable, fast and stable basis for vortex based smoke simulation," in *Symposium on Computer Animation (SCA 06)*. ACM, 2006, pp. pages–25.

[30] R. Bridson, J. Houriham, and M. Nordenstam, "Curl-noise for procedural fluid flow," *ACM Transactions on Graphics (ToG)*, vol. 26, no. 3, pp. 46–es, 2007.

[31] M. B. Nielsen, B. B. Christensen, N. B. Zafar, D. Roble, and K. Museth, "Guiding of smoke animations through variational coupling of simulations at different resolutions," in *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2009, pp. 217–226.

[32] M. B. Nielsen and B. B. Christensen, "Improved variational guiding of smoke animations," in *Computer Graphics Forum*, vol. 29, no. 2. Wiley Online Library, 2010, pp. 705–712.

[33] M. B. Nielsen and R. Bridson, "Guide shapes for high resolution naturalistic liquid simulation," in *ACM SIGGRAPH 2011 papers*, 2011, pp. 1–8.

[34] Z. Yuan, F. Chen, and Y. Zhao, "Pattern-guided smoke animation with lagrangian coherent structure," in *Proceedings of the 2011 SIGGRAPH Asia Conference*, 2011, pp. 1–8.

[35] S. Sato, T. Morita, Y. Dobashi, and T. Yamamoto, "A data-driven approach for synthesizing high-resolution animation of fire," in *Proceedings of the digital production symposium*, 2012, pp. 37–42.

[36] S. Sato, Y. Dobashi, K. Iwasaki, T. Yamamoto, and T. Nishita, "Deformation of 2d flow fields using stream functions," in *SIGGRAPH Asia 2014 Technical Briefs*, 2014, pp. 1–4.

[37] S. Sato, Y. Dobashi, Y. Yue, K. Iwasaki, and T. Nishita, "Incompressibility-preserving deformation for fluid flows using vector potentials," *The Visual Computer*, vol. 31, no. 6, pp. 959–965, 2015.

[38] S. Sato, Y. Dobashi, and T. Kim, "Stream-guided smoke simulations," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021. [Online]. Available: https://doi.org/10.1145/3450626.3459846

[39] Z. Forootaninia and R. Narain, "Frequency-domain smoke guiding," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–10, 2020.

[40] D. Reynolds, *Gaussian Mixture Models*. Boston, MA: Springer US, 2009, pp. 659–663. [Online]. Available: https://doi.org/10.1007/978-0-387-73003-5_196

[41] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[42] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," *Advances in neural information processing systems*, vol. 27, 2014.

[43] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[44] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[45] Y. Xie, A. Franz, M. Chu, and N. Thuerey, "tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–15, 2018.

[46] W. Wei and S. Liu, "Interpolating frames for super-resolution smoke simulation with gans," in *International Conference on Computer Animation and Social Agents*. Springer, 2020, pp. 14–21.

[47] G. Yan, Z. Chen, J. Yang, and H. Wang, "Interactive liquid splash modeling by user sketches," *ACM Transactions on Graphics (TOG)*, vol. 39, pp. 1–13, 2020.

[48] M. Chu, N. Thuerey, H.-P. Seidel, C. Theobalt, and R. Zayer, "Learning meaningful controls for fluids," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–13, 2021.

[49] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[50] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[51] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," pp. 10 684–10 695, 2022.

[52] 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.

[53] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." *ICLR*, vol. 1, no. 2, p. 3, 2022.

[54] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 3836–3847.

[55] D. Shu, Z. Li, and A. B. Farimani, "A physics-informed diffusion model for high-fidelity flow field reconstruction," *Journal of Computational Physics*, vol. 478, p. 111972, 2023.

[56] M. Lienen, D. Lüdke, J. Hansen-Palmus, and S. Günnemann, "From zero to turbulence: Generative modeling for 3d flow simulation, 2024," *URL https://arxiv. org/abs/2306.01776*.

[57] 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.

[58] Y. Guo, C. Yang, A. Rao, Z. Liang, Y. Wang, Y. Qiao, M. Agrawala, D. Lin, and B. Dai, "Animatediff: Animate your personalized text-to-image diffusion models without specific tuning," *arXiv preprint arXiv:2307.04725*, 2023.

[59] Q. Cao, D. Wang, X. Li, Y. Chen, C. Ma, and X. Yang, "Teaching video diffusion model with latent physical phenomenon knowledge," *arXiv preprint arXiv:2411.11343*, 2024.

[60] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes, "Sketch: An interface for sketching 3d scenes," in *ACM SIGGRAPH 2006 Courses*, 2006, pp. 9–es.

[61] T. Igarashi, S. Matsuoka, and H. Tanaka, "Teddy: a sketching interface for 3d freeform design," in *ACM SIGGRAPH 2006 Courses*, 2006, pp. 11–es.

[62] O. A. Karpenko and J. F. Hughes, "Smoothsketch: 3d free-form shapes from complex sketches," in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 589–598.

[63] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Fibermesh: designing freeform surfaces with 3d curves," in *ACM SIGGRAPH 2007 papers*, 2007, pp. 41–es.

[64] S.-H. Bae, R. Balakrishnan, and K. Singh, "Ilovesketch: as-natural-as-possible sketching system for creating 3d curve models," in *Proceedings*

of the 21st annual ACM symposium on User interface software and technology, 2008, pp. 151–160.

[65] M. Thorne, D. Burke, and M. Van De Panne, "Motion doodles: an interface for sketching character motion," *ACM Transactions on Graphics (ToG)*, vol. 23, no. 3, pp. 424–431, 2004.

[66] W. Chen and J. Hays, "Sketchygan: Towards diverse and realistic sketch to image synthesis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9416–9425.

[67] C. Gao, Q. Liu, Q. Xu, L. Wang, J. Liu, and C. Zou, "Sketchycoco: Image generation from freehand scene sketches," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 5174–5183.

[68] B. Liu, K. Song, Y. Zhu, and A. Elgammal, "Sketch-to-art: Synthesizing stylized art images from sketches," in *Proceedings of the Asian conference on computer vision*, 2020.

[69] Y. He, H. Xie, C. Zhang, X. Yang, and K. Miyata, "Sketch-based normal map generation with geometric sampling," in *International Workshop on Advanced Imaging Technology (IWAIT) 2021*, vol. 11766. SPIE, 2021, pp. 261–266.

[70] X. Du, Y. He, X. Yang, C.-M. Chang, and H. Xie, "Sketch-based 3d shape modeling from sparse point clouds," in *International Workshop on Advanced Imaging Technology (IWAIT) 2022*, vol. 12177. SPIE, 2022, pp. 714–719.

[71] Y. He, H. Xie, and K. Miyata, "Sketch2cloth: Sketch-based 3d garment generation with unsigned distance fields," in *2023 Nicograph International (NicoInt)*. IEEE, 2023, pp. 38–45.

[72] X. Du, T. Zhang, and H. Xie, "Dualshape: Sketch-based 3d shape design with part generation and retrieval," *IEEE Access*, vol. 12, pp. 18 888–18 900, 2024.

[73] Y. Peng, C. Zhao, H. Xie, T. Fukusato, and K. Miyata, "Difffacesketch: High-fidelity face image synthesis with sketch-guided latent diffusion model," 2023.

[74] T. Zhang, X. Xie, X. Du, and H. Xie, "Sketch-guided scene image generation with diffusion model," *Computers & Graphics*, p. 104226, 2025.

[75] H. Jin and H. Xie, "Sketch-based fluid video generation using motion-guided diffusion models in still landscape images," in *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters*, 2025, pp. 1–2.

[76] R. Bridson, *Fluid simulation for computer graphics*. CRC press, 2015.

[77] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[78] J. C. Butcher, *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016.

[79] E. F. Kaasschieter, "Preconditioned conjugate gradients for solving singular systems," *Journal of Computational and Applied mathematics*, vol. 24, no. 1-2, pp. 265–275, 1988.

[80] J. Bolz, I. Farmer, E. Grinspun, and P. Schröder, "Sparse matrix solvers on the gpu: conjugate gradients and multigrid," *ACM transactions on graphics (TOG)*, vol. 22, no. 3, pp. 917–924, 2003.

[81] G. Haller and G. Yuan, "Lagrangian coherent structures and mixing in two-dimensional turbulence," *Physica D: Nonlinear Phenomena*, vol. 147, no. 3-4, pp. 352–370, 2000.

[82] G. Haller, "Finding finite-time invariant manifolds in two-dimensional velocity fields," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 10, no. 1, pp. 99–108, 2000.

[83] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.

[84] P. J. Bickel and K. A. Doksum, *Mathematical statistics: basic ideas and selected topics, volumes I-II package*. Chapman and Hall/CRC, 2015.

[85] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," pp. 234–241, 2015.

[86] P. Holl, V. Koltun, and N. Thuerey, "Learning to control pdes with differentiable physics," 2020.

[87] M. Spiegel and L. Stephens, *Schaums Outline of Statistics, Fourth Edition*, ser. Schaum's Outline Series. McGraw-Hill Education, 2011. [Online]. Available: https://books.google.co.jp/books?id=787DbwAACAAJ

[88] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

[89] A. Selle, R. Fedkiw, B. Kim, Y. Liu, and J. Rossignac, "An unconditionally stable maccormack method," *Journal of Scientific Computing*, vol. 35, no. 2, pp. 350–371, 2008.

[90] Z. Ding and X. Tricoche, "Improved characterization of lagrangian coherent structures through time-scale analysis," *arXiv preprint arXiv:2203.11452*, 2022.

[91] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," *International journal of computer vision*, vol. 30, pp. 117–156, 1998.

[92] F. Gao, G. Wei, S. Xin, S. Gao, and Y. Zhou, "2d skeleton extraction based on heat equation," *Computers & Graphics*, vol. 74, pp. 99–108, 2018.

[93] J. D. Furst and S. M. Pizer, "Marching ridges." in *SIP*, vol. 1. Citeseer, 2001, pp. 22–26.

[94] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[95] P. Holl, V. Koltun, K. Um, and N. Thuerey, "phiflow: A differentiable pde solving framework for deep learning via physical simulations," in *NeurIPS workshop*, vol. 2, 2020.

[96] N. Thuerey and T. Pfaff, "MantaFlow," 2018, *http://mantaflow.com*.

[97] Z. Wang, Z. Yuan, X. Wang, Y. Li, T. Chen, M. Xia, P. Luo, and Y. Shan, "Motionctrl: A unified and flexible motion controller for video generation," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11.

[98] T. Inglis, M.-L. Eckert, J. Gregson, and N. Thuerey, "Primal-dual optimization for fluids," in *Computer Graphics Forum*, vol. 36, no. 8. Wiley Online Library, 2017, pp. 354–368.

[99] P. von Platen, S. Patil, A. Lozhkov, P. Cuenca, N. Lambert, K. Rasul, M. Davaadorj, D. Nair, S. Paul, W. Berman, Y. Xu, S. Liu, and T. Wolf, "Diffusers: State-of-the-art diffusion models," https://github.com/huggingface/diffusers, 2022.

[100] L. Jiang, S. Chen, B. Wu, X. Guan, and J. Zhang, "Vidsketch: Hand-drawn sketch-driven video generation with diffusion control," *arXiv preprint arXiv:2502.01101*, 2025.

[101] T. Unterthiner, S. Van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly, "Towards accurate generative models of video: A new metric & challenges," *arXiv preprint arXiv:1812.01717*, 2018.

[102] 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*, 2018, pp. 586–595.

[103] 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*, vol. 13, no. 4, pp. 600–612, 2004.

[104] A. Hore and D. Ziou, "Image quality metrics: Psnr vs. ssim," in *2010 20th international conference on pattern recognition*. IEEE, 2010, pp. 2366–2369.

[105] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," *arXiv preprint arXiv:2307.01952*, 2023.

Publications

[1] **H. Chang**, T. Zhang, S. Sato and H. Xie, "DiffSmoke: Two-Stage Sketch-Based Smoke Illustration Design Using Diffusion Models," *IEEE Access*, vol. 13, pp. 44997–45009, 2025. doi: 10.1109/ACCESS.2025.3548433. (Peer reviewed)

[2] **H. Chang**, X. Xie, S. Sato and H. Xie, "Two-Stage Sketch-Based Smoke Illustration Generation Using Stream Function," in *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters*, 2025. (Peer reviewed)

[3] R. Miyauchi, **H. Chang**, T. Fukusato, K. Miyata and H. Xie, "Physics-Aware Fluid Field Generation from User Sketches Using Helmholtz-Hodge Decomposition," in *NICOGRAPH International 2025*, Jun. 2025. (Peer reviewed)

[4] **H. Chang**, S. Sato, H. Xie "Two-Stage Sketch-Based Velocity Field Design with Latent Diffusion Model. 4th International Symposium on Intelligence Design," *IEEE Access*, vol. 13, pp. 44997–45009, 2025. doi: 10.1109/ACCESS.2025.3548433. (Non-peer-reviewed)

[5] **H. Chang**, Y. Peng, S. Sato and H. Xie, "Sketch-guided flow field generation with diffusion model," in *International Workshop on Advanced Imaging Technology (IWAIT) 2024*, Jan. 2024. (Peer reviewed)

[6] **H. Chang**, X. Xie, S. Sato and H. Xie, "Stream-Guided Sketch-Based Smoke Illustration Design," in *Visual Computing 2025*, Sep. 2025. (Peer reviewed)

[7] 宮内竜一, **常恒遠**, 福里司, 宮田一乗, 謝浩然, ヘルムホルツ分解を用いたスケッチベース流体生成 in 情報処理学会コンピュータグラフィックスとビジュアル情報学研究会第 197 回研究発表会, Mar. 2025 (Non-peer-reviewed)

[8] **常恒遠**, 張天宇, 佐藤周平, 謝浩然, イラストデザインのための拡散モデルを用いたスケッチによる煙の流れの生成 in *Visual Computing 2024*, Sep. 2024. (Peer reviewed)

[9] **常恒遠**, 彭以琛, 佐藤周平, 謝浩然, 拡散モデルを用いたスケッチベース流体速度場の生成 in *Visual Computing 2023*, Sep. 2023. (Non-peer-reviewed)