Learn2Fold: Structured Origami Generation with World Model Planning

YANJIA HUANG*†, UCLA, Texas A&M University
YUNUO CHEN*, UCLA
YING JIANG*, UCLA
JINRU HAN, UCLA
ZHENGZHONG TU, Texas A&M University
YIN YANG, University of Utah
CHENFANFU JIANG, UCLA

Fig. 1. **Teaser.** From simple planes to complex articulated forms, Learn2Fold plans origami folding sequences that respect geometric constraints and anticipate future consequences, enabling robust generalization across unseen crease patterns.

The ability to transform a flat sheet into a complex three-dimensional structure is a fundamental test of physical intelligence. Unlike cloth manipulation, origami is governed by strict geometric axioms and hard kinematic constraints, where a single invalid crease or collision can invalidate the entire folding sequence. As a result, origami demands long-horizon constructive reasoning that jointly satisfies precise physical laws and high-level semantic intent. Existing approaches fall into two disjoint paradigms: optimization-based methods enforce physical validity but require dense, precisely specified inputs, making them unsuitable for sparse natural language descriptions, while generative foundation models excel at semantic and perceptual synthesis yet fail to produce long-horizon, physics-consistent folding processes. Consequently, generating valid origami folding sequences directly from text remains an open challenge. To address this gap, we introduce **Learn2Fold**, a neuro-symbolic framework that formulates origami folding as conditional program induction over a crease-pattern graph. Our key insight is to decouple semantic proposal from physical verification. A large language model generates candidate folding programs from abstract text prompts, while a learned graph-structured world model serves as a differentiable surrogate simulator that predicts physical feasibility and failure modes before execution. Integrated within a lookahead planning loop, Learn2Fold enables robust generation of physically valid folding sequences for complex and out-of-distribution patterns, demonstrating that effective spatial intelligence arises from the synergy between symbolic reasoning and grounded physical simulation.

CCS Concepts: • **Computing methodologies** → **Shape modeling**.

Additional Key Words and Phrases: 3D generation, origami generation.

Yun

1 Introduction

Recent advances in generative AI have enabled the synthesis of increasingly complex visual content, including images, videos, and 3D assets [Chen et al. 2023; Gao et al. 2022; Li et al. 2025b; Lu et al. 2024; Nam et al. 2022]. However, most of these successes focus on generating static or perceptual representations, where physical feasibility and execution constraints are either ignored or only weakly enforced. Extending generative models beyond visual plausibility toward physically executable processes remains an open and largely unexplored challenge. This challenge becomes particularly pronounced in tasks that require long-horizon reasoning under strict geometric and topological constraints. While recent progress

*Equal contributors.
†Work done while visiting the AIVC Lab, UCLA.

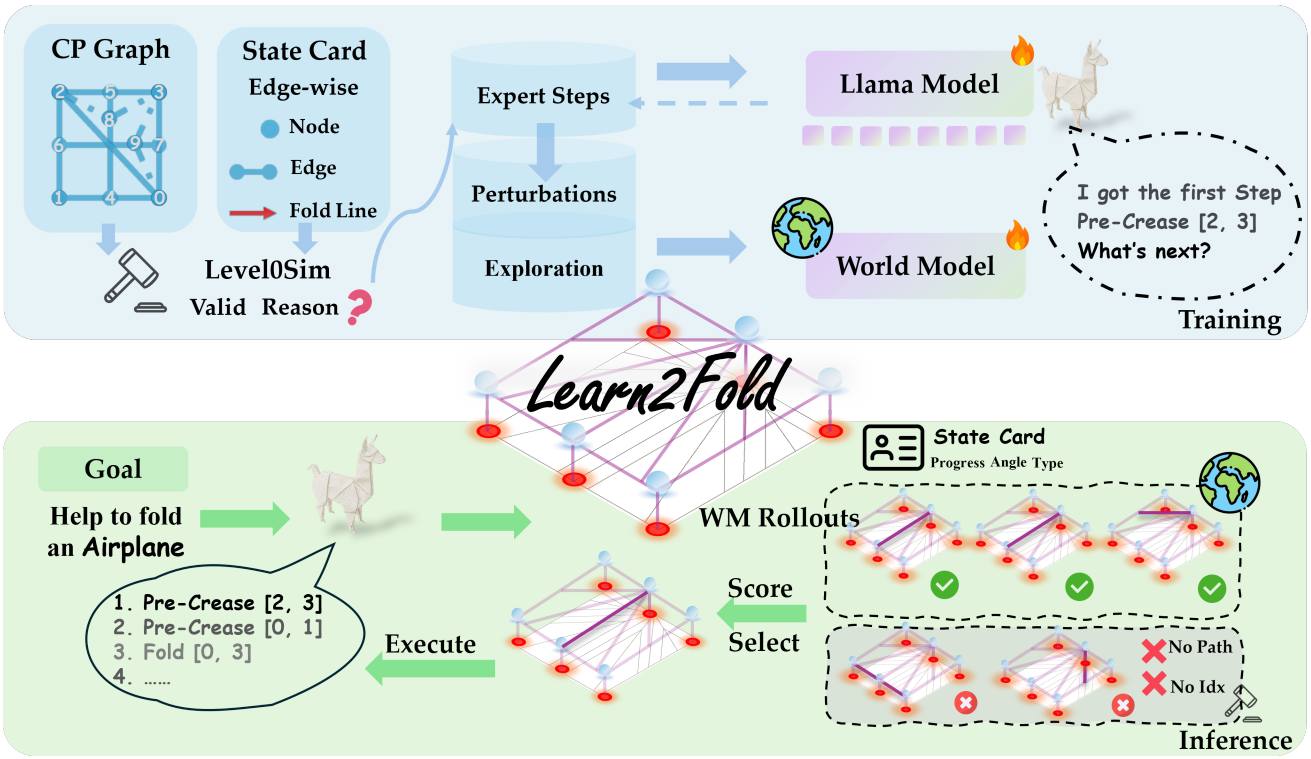

Fig. 2. **Overview of Learn2Fold.** Learn2Fold formulates origami folding as constraint-aware sequential program generation. During training, a symbolic Level-0 simulator enables scalable data generation and supervision for both a language-based proposal model and a learned world model. At inference time, Learn2Fold combines LM proposals with world-model rollouts and MPC to robustly plan folding sequences under hard constraints.

in deformable object manipulation, such as cloth folding [Lee et al. 2024; Li et al. 2015; Liu et al. 2025a; Team 2025a; Tian et al. 2025], has demonstrated impressive results, these settings benefit from the inherent compliance and error tolerance of amorphous materials. Garments can accommodate local inaccuracies through smoothing and deformation, allowing learning-based methods to recover from imprecise actions. In contrast, origami folding operates under a fundamentally different regime. Origami is the art of transforming a flat sheet into a three-dimensional structure through a sequence of folds, governed by strict geometric axioms and topological constraints [Lang 2011; Maitin-Shepard et al. 2010]. A single misplaced crease does not merely introduce a local artifact, but can violate surface topology or render all subsequent folding steps mathematically infeasible. As a result, origami demands precise coordination of discrete topological changes and continuous geometric motions over long horizons, with little tolerance for error.

In this work, we adopt origami folding as a challenging and principled testbed for studying constraint-aware generative planning. Digitally representing and generating origami processes requires modeling both a structured crease pattern and the progressive, constraint-driven folding dynamics that transform a flat sheet into a valid 3D shape. Despite its conceptual simplicity, origami exposes the core limitations of existing generative approaches and serves as a rigorous benchmark for evaluating long-horizon spatial reasoning under hard physical constraints.

Prior work on origami generation can be broadly categorized into learning based methods and optimization based approaches. Generative models, including large language models and vision language models [Team 2023, 2024, 2025b; Zhang et al. 2024], are trained on large scale multimodal data such as origami videos, images, and textual instructions. These models can produce descriptive tutorials or high level folding guidance conditioned on text prompts or images. However, they typically fail to generate physically executable origami processes, as they optimize for approximate visual plausibility rather than exact physical feasibility, often hallucinating geometries that appear visually coherent but violate folding constraints. In contrast, traditional optimization based methods [He et al. 2023; Lang 2011; Tachi 2010] formulate origami generation as a constrained optimization problem, employing techniques such as circle packing or tuck folding algorithms to mathematically guarantee that a target mesh can be folded from a single sheet. These approaches produce simulation ready, physically grounded crease patterns, but require precise 3D mesh inputs, making them difficult to apply to sparse inputs such as a single image or a text prompt. This raises a key question: can we retain the physical rigor and simulation ready representations of computational origami while leveraging the powerful priors of large language and vision language

models to reconstruct executable origami processes from enriching semantic descriptions?

To bridge these gaps, we introduce **Learn2Fold**, a neuro-symbolic framework that formulates origami folding as constraint-aware program induction. Our key insight is that robust generation requires separating *proposal* from *verification*. Instead of blindly decoding a sequence, Learn2Fold operates in a propose, verify loop. We leverage a Large Language Model (LLM) to propose high-level structured action tokens, utilizing its semantic planning capabilities. However, acknowledging that LLMs lack intrinsic physics grounding, we integrate a learned Graph-Structured World Model for lookahead planning. This world model acts as a differentiable surrogate simulator, allowing the system to *imagine* the geometric consequences of actions and prune branches that lead to invalid states before execution. We propose a symbolic simulator that performs final constraint verification, complementing neural proposal and learned lookahead with exact geometric feasibility checks.

Our contributions are summarized as follows:

- We propose Learn2Fold, a novel framework for origami process generation that integrates a Large Language Model (LLM) for high-level structured action proposal with a learned Graph-Structured World Model for physics-aware lookahead planning and verification.
- A scalable, simulation-driven data curation engine for origami that generates large-scale folding transitions using counterfactual perturbations and propose a new origami dataset, OrigamiCode dataset containing structured folding programs and verified transitions for learning origami folding dynamics.
- We validate the effectiveness of the proposed method through comprehensive experiments, demonstrating robust generalization to out-of-distribution physically valid and executable origami generation.

2 Related Work

2.1 Structured and Constraint-Aware Generation

Recent generative models have demonstrated remarkable proficiency in synthesizing high fidelity assets, ranging from static 3D shapes [Lan et al. 2025; Li et al. 2025a; Voleti et al. 2024; Wang et al. 2023] to dynamic video sequences [Bruce et al. 2024a; Huang et al. 2025; Ramesh et al. 2022; Rombach et al. 2022]. However, modeling progressive shape formation processes like origami folding still remains an open challenge. Unlike one-shot generation methods that directly predict a final geometry, origami folding is intrinsically an *executable, long-horizon action sequence*. This process operates on a complex hybrid discrete-continuous state space. This task involves discrete topological changes such as face layering, connectivity updates, coupled with continuous kinematic transformations. Crucially, this generation process is governed by strict physical validity. Every folding step must satisfy hard geometric and topological constraints, such as flat-foldability and self-intersection avoidance; a minor violation in early steps compounds, rendering the final result physically invalid. Consequently, this setting demands structured generation paradigms rather than unstructured end-to-end inference. To address similar structural challenges, recent works have adopted intermediate representations, such as scene graphs or layouts [Johnson et al. 2018; Liu et al. 2025b; Xu et al. 2017], to anchor object relations and reduce spurious outputs. Another line of research integrates constraint-aware decoding or verifier-guided search to ensure validity [Anderson et al. 2017; Pun et al. 2025; Yan et al. 2021]. For instance, recent structural synthesis models like BrickGPT [Pun et al. 2025] rely on reactive rollback to filter out physically unstable steps. While these assembly generation systems effectively combine auto-regressive proposals with rollback mechanisms, straightforward backtracking becomes computationally prohibitive for complex folding sequences. Distinguishing our work from these approaches, we propose a CP-grounded folding program equipped with diagnostic feedback. Instead of binary success or failure checks, our model performs causal attribution to identify why a fold failed, enabling efficient planning and recovery even on out-of-distribution crease patterns.

2.2 Computational Origami

Origami folding is fundamentally governed by rigorous mathematical rules concerning develop ability and flat-foldability like Kawasaki's and Maekawa's theorems[Bern and Hayes 1996; Demaine and O'Rourke 2007; Hull 2002]. To simulate these complex behaviors computationally, researchers have developed kinematic models that treat creases as rotational hinges. Early works focused on rigid origami, modeling the mesh as discrete rigid facets connected by joints [Tachi 2009, 2010]. To alleviate this, more recent approaches, such as the bar-and-hinge model used in Origami Simulator [Ghassaei et al. 2018], introduce compliance to approximate the elastic deformation of paper, enabling real-time folding visualization. While these simulators provide ground truth physics, they are purely forward-process tools where they calculate the geometric consequence of a given fold but do not posses the agency to plan a sequence or reason about hight-level semantic goals.

The problem of generation a crease patter (CP) for a target 3D shape has traditionally been formulated as a geometric optimization problem. Pioneering systems like TreeMaker [Lang 2011] and Origamizer [Tachi 2009] use circle packing or tuck-folding algorithms to mathematically guarantee that a specific mesh can be folded from a single sheet. However, these methods are strictly geometry-centric and deterministic. They lack the flexibility to handle ambiguous semantic descriptions and are often sensitive to topological errors, where a slight violation in the CP graph renders the entire optimization infeasible. Unlike these optimization-based solvers which require a perfect final mesh as input, our approach treats generation as a sequential decision-making process. This allows for robust recovery from intermediate errors and generalization to out-of-distribution patterns via previous trained structures.

2.3 World Models

World models learn action-conditioned dynamics to enable planning via imagined rollouts. This paradigm spans from classical latent-dynamics methods in model-based RL [Hafner et al. 2020, 2019; Rafailov et al. 2020] to recent foundation-scale video simulators that model physics in rich visual domains [Bruce et al. 2024a; Huang et al. 2025; Rigter et al. 2024]. However, pixel-based or latent world

models do not directly enforce hard discrete geometric constraints, nor do they naturally produce structured, executable programs. Furthermore, collecting action-labeled interaction data for specialized domains like origami remains prohibitive [Ai et al. 2025]. In our work, we learn a state-level world model over CP-graph states, supervised by scalable synthetic transitions from a deterministic constraint engine. Crucially, our training data includes near-boundary perturbations, exposing the model to both feasible and infeasible outcomes. This learned dynamics model enables efficient model-predictive lookahead, allowing the system to verify action feasibility and recover from proposal errors on out-of-distribution crease patterns.

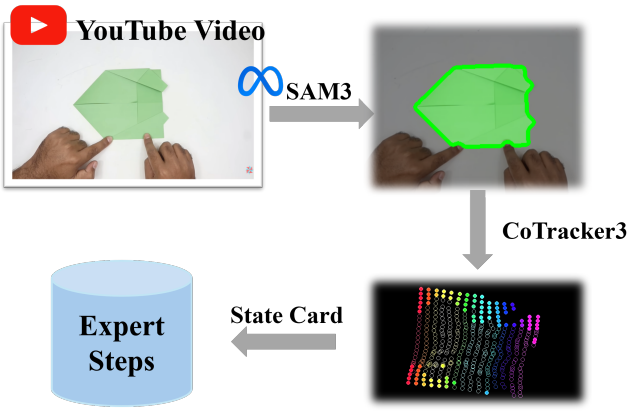

Fig. 3. **Deriving Expert Trajectories from Videos.** We show one data source for obtaining expert folding trajectories. In-the-wild instructional videos are processed into State Cards and folding steps, which are then augmented through perturbation and exploration for training.

3 Method

We target *physically valid generation* for Computational Origami: at inference time, our agent augments its base proposal policy with a graph-based world model that *imagines* future manifold states and converts them into validity scores for planning (Fig. 2). Our approach, Learn2Fold, tightly couples three components: ❶ a *Canonicalized Graph Representation* that ensures structural invariance; ❷ a *Generative Proposal* Policy that suggests candidate folds based on semantic goals; and ❸ a *Graph-based World Model* that rolls out short-horizon geometric futures. At test time, we do not only rely on the policy's likelihood; instead, the world model's predictions are fused at the score level via model predictive control (MPC) to rank candidate actions, ensuring strict geometric feasibility without sacrificing generative flexibility.

In the following sections, Sec. 3.1 details the canonicalized state representation. Sec. 3.2 formalizes the language-conditioned proposal policy. Sec. 3.3 introduces the graph world model, which acts as a differentiable surrogate simulator. Finally, Sec. 3.4 describes the MPC planning strategy that integrates these signals for robust action selection.

3.1 State Representation and Canonicalization

We formulate the origami folding process as a sequential manipulation of a graph-structured manifold. An origami instance is represented by a tuple $O_t = (\mathcal{G}, s_t)$, where $\mathcal{G}$ denotes its static topology and s_t denotes a dynamic state.

Static Graph Topology. The crease pattern (CP) is a planar graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with points $\mathcal{V} = \{v^i \in [0,1]^2\}_{i=1}^{N_v}$ and edges $\mathcal{E} = \{e^j\}_{j=1}^{N_e}$. Each edge may carry an initial crease type label $z_0^j \in \{\mathrm{M}, \mathrm{V}, \mathrm{U}\}$ (M: mountain, V: valley, U: unknown).

Canonicalization. Raw CP data often contains arbitrary vertex indexing, which hinders learning. To ensure permutation invariance and robust generalization, we apply a deterministic canonicalization process $\Phi : \mathcal{G} \to \mathcal{G}^*$. Specifically, we (i) reindex vertices via lexicographical sorting of coordinates, and (ii) reindex edges based on the sorted endpoint indices. To further eliminate orientation bias, we augment the training data by applying dihedral symmetries (rotations and reflections) to $\mathcal{V}$ prior to canonicalization. This ensures that structurally identical patterns map to the same index space.

Dynamic State. We track the folding status using a state vector $s_t = (\alpha_t, \rho_t, z_t, \psi_t, b_t, t)$, where $\alpha_t \in [-\pi, \pi]^{|\mathcal{E}|}$ are signed dihedral angles, $\rho_t \in [0,1]^{|\mathcal{E}|}$ are progress ratios, $z_t \in \{\mathrm{M}, \mathrm{V}, \mathrm{U}\}^{|\mathcal{E}|}$ are crease types, ψ_t is the global frame angle, b_t is the MV-flip flag, and t is the step counter.

3.2 Policy Learning via Language Models

We frame origami folding as a conditional program induction task. The objective is to learn a policy $\pi_\theta(a_t \mid C_t)$ that generates a valid folding operation a_t given the current context C_t.

Unified Token Space. The action space of folding is inherently hybrid, requiring the selection of discrete graph elements (e.g., target edges) and continuous parameters (e.g., fold angles). To leverage the reasoning capabilities of Transformer-based LLMs, we unify these modalities into a homogeneous vocabulary $\Sigma = \Sigma_{\text{ops}} \cup \Sigma_{\text{graph}} \cup \Sigma_{\text{geo}}$. Continuous geometric parameters are quantized into discrete bins Σ_{geo}, while canonicalized graph indices are mapped to semantic tokens Σ_{graph}. This formulation transforms the complex control problem into an autoregressive sequence modeling task, enabling the model to capture joint dependencies between topological intent and geometric specifications.

Context and Objective. The policy is conditioned on a context $C_t = (g; \mathcal{G}^*, s_t)$, where g denotes the high-level semantic goal. By operating on the canonicalized graph $\mathcal{G}^*$, the policy learns structure-invariant *folding motifs* (e.g., "rabbit-ear fold") rather than overfitting to instance-specific identifiers (e.g., vertex indices). We train the model using Maximum Likelihood Estimation (MLE) on expert demonstrations $\mathcal{D}$:

$$\mathcal{L}_{\text{policy}}(\theta) = \mathbb{E}_{(C,a^*)\sim\mathcal{D}}\left[-\sum_k \log \pi_\theta(a_{t,k} \mid C_t, a_{t,<k})\right], \quad (1)$$

where $a_{t,k}$ denotes the k-th token of the action sequence at step t. This supervised pre-training instills the *grammar* of valid folding operations.

3.3 Graph-Based World Model

While the policy proposes plausible actions, ensuring strict physical feasibility requires rigorous verification. To enable efficient lookahead planning without computationally expensive mesh-based simulations, we learn a differentiable world model $\mathcal{M}_\phi$ that acts as a surrogate simulator.

Residual Graph Dynamics. Unlike pixel-based world models [Bruce et al. 2024b] which lack explicit geometric constraints, our model operates directly on the graph state s_t. We formulate the transition as a sparse residual update:

$$\Delta\hat{s}_t,\ \hat{m}_t,\ \hat{c}_{t+1} = \mathcal{M}_\phi(\mathcal{G}^*, s_t, a_t), \qquad \hat{s}_{t+1} = s_t + \Delta\hat{s}_t \odot \text{expand}(\hat{m}_t), \tag{2}$$

where $\hat{m}_t \in [0,1]^{|\mathcal{E}|}$ is a locality mask and $\hat{c}_{t+1} \in [0,1]^{|\mathcal{E}|}$ estimates per-edge constraint violation likelihood. expand$(\cdot)$ broadcasts the per-edge mask to all state channels.

3.4 Inference via Graph-Guided MPC

At test time, we perform a constrained lookahead search on the CP graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. At each step t, our proposal policy π_θ generates candidate structured actions, which are filtered by a hard verifier (Level-0 simulator) and ranked by the learned world model.

Candidate Sampling. We sample K candidate actions from the proposal distribution using nucleus sampling:

$$\mathcal{A}_t = \{a_t^{(k)}\}_{k=1}^K, \qquad a_t^{(k)} \sim \pi_\theta(\cdot \mid C_t). \tag{3}$$

Hard Verification (Level-0). Each candidate is first evaluated by a deterministic constraint kernel:

$$(\tilde{s}_{t+1}^{(k)}, v_t^{(k)}, r_t^{(k)}, m_t^{(k)}) = \textsc{Level0Sim}(\mathcal{G}^*, s_t, a_t^{(k)}), \tag{4}$$

where $v_t^{(k)} \in \{0,1\}$ indicates fold validity, $r_t^{(k)}$ denotes the reason for invalidity, and $m_t^{(k)} \in \{0,1\}^{|\mathcal{E}|}$ is the affected-edge mask. We discard invalid candidates and retain $\mathcal{A}_t^{\text{valid}} = \{a_t^{(k)} \in \mathcal{A}_t \mid v_t^{(k)} = 1\}$.

World-Model Rollout. For each valid candidate, the world model predicts residual state updates and a soft violation mask:

$$\Delta\hat{s}_t^{(k)},\ \hat{c}_{t+1}^{(k)} = \mathcal{M}_\phi(\mathcal{G}^*, s_t, a_t^{(k)}), \qquad \hat{s}_{t+1}^{(k)} = s_t + \Delta\hat{s}_t^{(k)}, \tag{5}$$

where $\hat{c}_{t+1}^{(k)} \in [0,1]^{|\mathcal{E}|}$ estimates per-edge constraint violation likelihood (a soft counterpart of m_t).

Action Selection. We choose the action maximizing a fused objective of proposal likelihood, goal progress, and feasibility:

$$a_t^* = \arg\max_{a_t^{(k)} \in \mathcal{A}_t^{\text{valid}}} \frac{1}{|a_t^{(k)}|} \log \pi_\theta(a_t^{(k)} \mid C_t) - \lambda_{\text{goal}}\, U_{\text{goal}}(\hat{s}_{t+1}^{(k)}) + \lambda_{\text{cst}}\, \log\Big(\epsilon + 1 - \|\hat{c}_{t+1}^{(k)}\|_\infty\Big). \tag{6}$$

Here $\lambda_{\text{goal}}, \lambda_{\text{cst}}$ balance goal pursuit and constraint satisfaction, and $\epsilon > 0$ avoids numerical instability.

Failure and Re-sampling. In the case when $\mathcal{A}_t^{\text{valid}} = \emptyset$ or $\max_k J^{(k)} < \tau$, we construct a negative constraint from the predicted violation mask (e.g., top-M edges with highest $\hat{c}$) and re-sample candidates under the updated constraint set.

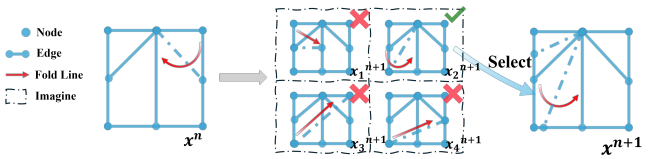

Fig. 4. **Folding with Reasoning.** Learn2Fold incrementally constructs origami folding programs in CP-graph space. At each step, multiple candidate actions are evaluated through world-model rollouts, infeasible options are discarded, and the best action is selected for execution, enabling robust folding and recovery under hard constraint

4 Experiments

4.1 Experiment Setup

Implementation Details. We train the world model (WM) using large-scale synthetic folding data generated by the Level-0 simulator. Specifically, we collect approximately 76,000 transitions through expert demonstrations and constraint-guided perturbations, and train the WM with supervised learning for 50 epochs, which takes about 30 hours on a single NVIDIA RTX Pro 6000 GPU. The language model (LM) is a lightweight decoder-only transformer fine-tuned to generate structured folding actions under a fixed JSON schema. It is trained using roughly 10^4 expert folding steps augmented with simulator-verified perturbations, and converges within 6 hours using LoRA adapters on the same hardware. At inference time, Learn2Fold runs in a model predictive control (MPC) loop, where the LM proposes $N = 8$ candidate actions per step, the simulator filters invalid ones, and the WM scores the remaining candidates via short-horizon rollouts to select the final action. All experiments are conducted with fixed random seeds for reproducibility.

Dataset. To rigorously evaluate topological generalization, we curate a held-out benchmark of 25 distinct origami categories that span the full spectrum of folding complexity. Unlike previous datasets dominated by simple shapes, our benchmark is carefully stratified into three difficulty tiers based on step count and non-local dependency: **Simple (10 categories):** Basic rigid folding structures with minimal layering (e.g., *Paper Airplanes*, *Hearts*, *Cups*). **Intermediate (10 categories):** Standard models requiring moderate spatial planning and box-pleating (e.g., *Boats*, *Flowers*). **Complex (5 categories):** High-frequency folding sequences with intricate appendage management and strict circle-packing constraints (e.g., *Insects, Cranes, Dragons*). This taxonomy allows us to disentangle basic instruction following from complex physical reasoning. Each instance provides a canonicalized CP and a ground-truth program. In total, we collect 5,760 origami process sequences and 75,000 trajectories in the OrigamiCode dataset. Following a standard train–test split, 80% of the data is used for training, while the remaining 20% is reserved for evaluation.

Baselines. We compare our approach against three representative methods. First, we evaluate BrickGPT [Pun et al. 2025], a reactive baseline adapted from assembly synthesis that employs a physics-aware rollback mechanism to filter unstable steps through trial-and-error execution and is trained on the proposed OrigamiCode dataset. Second, we benchmark against GPT-5.1 and GPT-5.2, the

Table 1. Main comparison across methods. Precision/Recall/F1 are computed on step-level structured tokens under a unified action schema (reported as the evaluator's category-averaged micro scores). Edge-IoU is computed between the affected-edge set induced by the predicted action and the simulator mask. Cat-SR is the macro-averaged trajectory success rate across categories.

Method	Precision$_\mu$ ↑	Recall$_\mu$ ↑	F1$_\mu$ ↑	Edge-IoU ↑	Cat-SR$_{\text{macro}}$ ↑
Gemini (prompted)	0.2874	0.4213	0.3420	0.1126	0.4942
GPT-5.1 (prompted)	0.2625	0.2996	0.2663	0.0937	0.6753
GPT-5.2 (prompted)	0.1243	0.3575	0.1648	0.1322	0.1600
BrickGPT (finetuned)	0.3969	0.2250	0.2461	0.0505	0.5455
Learn2Fold (Ours)	0.7661	0.7113	0.7394	0.5820	0.8912

latest state-of-the-art foundation models. These general-purpose agents are provided with in-context examples to output structured folding programs, representing the upper bound of unconstrained semantic planning without specialized geometric modules. Finally, we compare these against Learn2Fold (Ours), which generates actions under explicit graph-based lookahead verification.

Metrics. We evaluate performance at both the step level and the trajectory level. At the step level, we report Precision, Recall, and F1 to measure how accurately each method predicts structured folding actions under a unified action schema, capturing both the correctness and coverage of discrete decisions. At the trajectory level, we report Category Success Rate (Cat-SR) and Edge-IoU to evaluate long-horizon execution performance and structural alignment, respectively. Cat-SR is defined as the fraction of folding sequences that successfully complete the target origami within each category and is macro-averaged across categories to mitigate class imbalance. Edge-IoU measures whether a predicted action affects the correct set of creases by computing the intersection-over-union between the predicted affected-edge set and the simulator-derived ground truth.

4.2 Quantitative Evaluation

In the absence of standard benchmarks for origami process generation, following [Pun et al. 2025], we construct a custom test set comprising 3,840 text prompts spanning 25 categories. From this set, we select 1,150 cases for validation and perform two independent runs per prompt for each method, yielding 7,680 results per method. As shown in Table 1, our method outperforms all baselines across all metrics in both step-level accuracy and trajectory-level success. At the step level, Learn2Fold achieves a Precision$_\mu$/Recall$_\mu$/F1$_\mu$ of 0.766/0.711/0.739, substantially exceeding the strongest baseline (GPT-5.1, F1$_\mu$ = 0.266), corresponding to a +47.3 point absolute improvement in F1. In contrast, LLM-based baselines exhibit a pronounced precision–recall imbalance. For example, GPT-5.2 achieves a relatively high Recallμ (0.358) but very low Precisionμ (0.124), suggesting that while many relevant actions are proposed, they are often imprecise or misaligned with the required structural context since LLMs operate at a coarse semantic level, lacking detailed visual guidance. As a result, they can outline plausible folding intentions but cannot resolve the fine-grained, step-specific details. As for BrickGPT, while it benefits from explicit rollback-based execution and achieves higher precision than LLM-based baselines, it still suffers from limited recall, indicating that its reactive trial-and-error strategy produces coarse and incomplete folding actions and fails to consistently recover the full sequence of required steps.

At the trajectory level, Learn2Fold achieves an Edge-IoU of 0.582, demonstrating significantly improved alignment between predicted actions and their induced affected-edge sets. In contrast, LLM-based baselines lack edge-level causal grounding, leading to diffuse or incorrect predictions of which creases are affected by each operation. BrickGPT, despite incorporating a physics-aware rollback mechanism, attains only marginal Edge-IoU improvement (0.0505), indicating that reactive trial-and-error execution alone is insufficient to recover accurate edge-level structure. In terms of long-horizon performance, Learn2Fold attains the highest Cat-SR of 0.891, indicating strong robustness over extended folding sequences. BrickGPT achieves a moderate Cat-SR of 0.546, suggesting that while rollback mechanisms can occasionally prevent catastrophic failures, they do not ensure consistent global progress. Overall, these results highlight the importance of explicit state modeling and stepwise feasibility enforcement for reliable long-horizon origami process generation.

4.3 Qualitative Study

Fig. 5 presents qualitative comparisons between Learn2Fold and baseline methods on representative examples from the same test set described in Sec. 4.2. LLM-based baselines typically fail after only a few steps. While the initial actions are often semantically plausible, errors quickly accumulate due to the lack of explicit geometric state tracking. As a result, many predicted steps are either structurally incorrect or misaligned with the underlying crease pattern, causing premature termination of the folding process. BrickGPT exhibits improved stability in the early stages of execution. In several examples, the first few predicted actions (e.g., the first three to four steps) are valid and physically feasible, benefiting from its rollback-based mechanism. However, as folding sequences grow longer, BrickGPT struggles to maintain long-term consistency, as it fails to capture fine-grained dependencies between distant steps, resulting in incorrect or incomplete long-horizon folding processes and limiting its ability to represent detailed step-by-step origami procedures for complex models. While Learn2Fold consistently produces coherent and fine-grained folding sequences across the entire trajectory. By explicitly modeling the folding state and verifying feasibility at each step, Learn2Fold maintains structural consistency and accurately captures the intended origami process, even for long and intricate folding sequences. These qualitative results corroborate

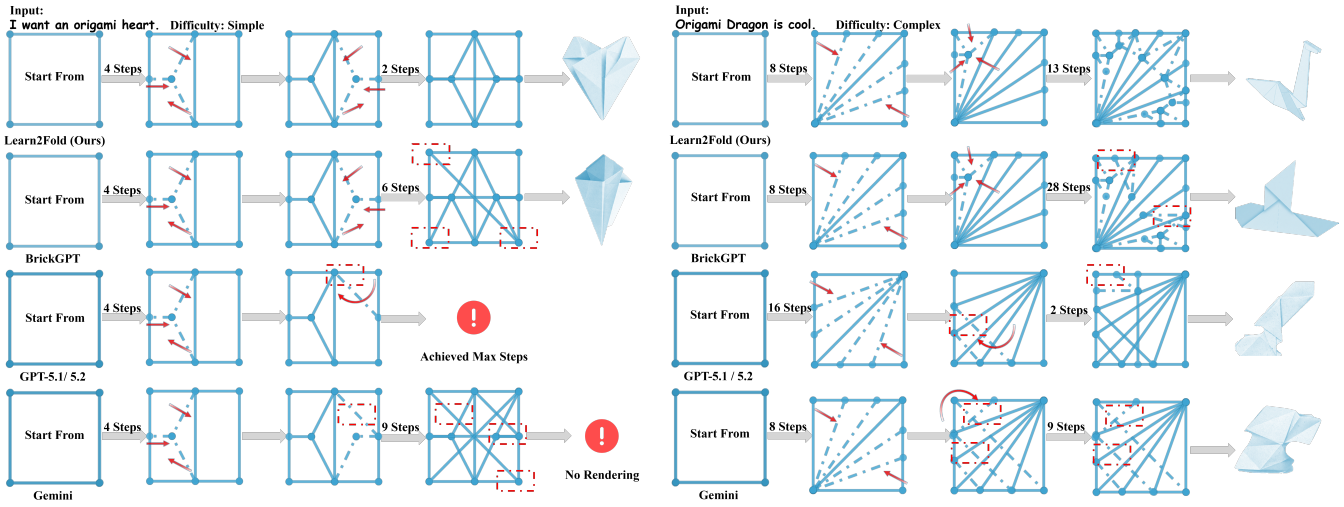

Fig. 5. **Qualitative comparison of folding behaviors across methods.** Learn2Fold produces concise, physically feasible folding trajectories on both simple and complex origami tasks. Baseline methods frequently fail due to invalid actions, early termination, or inability to recover from long-horizon errors, especially on complex crease patterns.

Method	Step Valid ↑	Traj SR ↑	Goal Dist ↓
LM	70.8% ± 45.5%	22.2% ± 45.5%	0.796 ± 0.194
LM+WM	54.2% ± 49.8%	25.0% ± 43.3%	0.759 ± 0.214
LM+WM+Level0Sim (Ours)	64.2% ± 41.8%	33.3% ± 47.1%	0.855 ± 0.196

Table 2. IID results. Blue and teal indicate the best and second-best results.

Method	Step Valid ↑	Traj SR ↑	Goal Dist ↓
LM	47.6% ± 29.2%	20.7% ± 55.5%	0.633 ± 0.192
LM+WM	32.3% ± 28.7%	17.8% ± 51.7%	0.560 ± 0.248
LM+WM+Level0Sim (Ours)	41.2% ± 32.3%	27.7% ± 50.1%	0.487 ± 0.353

Table 3. OOD results. Blue and teal indicate the best and second-best results.

the quantitative findings and highlight the advantage of explicit state modeling for reliable origami process generation.

4.4 Ablation

We conduct an ablation study of the key components of our framework to evaluate the contribution of each proposed component to the final origami process generation's performance.

Framework Design. We progressively ablate the system by comparing three configurations: (i) an LLM-only proposer, (ii) LLM augmented with a learned world model (LM+WM), and (iii) the full system Learn2Fold that further incorporates the Level0Sim constraint kernel (LM+WM+Level0Sim). These variants are evaluated under both in-distribution (IID) and out-of-distribution (OOD) CP holdout settings using step-level validity, trajectory success rate (Traj SR), and final goal distance.

Incorporating the world model alters the decision-making behavior by introducing short-horizon lookahead. Compared to the LLM-only baseline, LM+WM exhibits a modest improvement in trajectory-level success (IID Traj SR: 22.2% → 25.0%), and consistently reduces the final goal distance in both IID (0.796 → 0.759) and OOD settings (0.633 → 0.560). However, this improvement comes with a decrease in step-level validity, indicating that the world model prioritizes global progress over local action safety, occasionally selecting actions that are locally risky but potentially beneficial for long-horizon objectives.

Adding Level0Sim consistently improves long-horizon performance. The full system achieves the highest trajectory success in both IID and OOD settings, while recovering step-level validity and further reducing final goal distance under distribution shift.

Overall, the ablation indicates complementary roles of the LLM proposer, the world model, and Level0Sim, whose combination is required for robust long-horizon folding.

5 Conclusion

In this work, we present **Learn2Fold**, a neuro-symbolic framework for physically valid origami process generation that unifies semantic reasoning with rigorous geometric constraint enforcement. By formulating origami folding as a constraint-aware program induction over a CP graph, Learn2Fold addresses a fundamental limitation of prior generative models: the inability to reliably generate long-horizon, executable action sequences under strict topological and kinematic constraints.

We view origami not merely as an application, but as a principled testbed for future spatial reasoning systems, where it exposes core challenges in reasoning over structured space: discrete topological decisions coupled with continuous geometry, irreversible constraints, and long-horizon dependency.

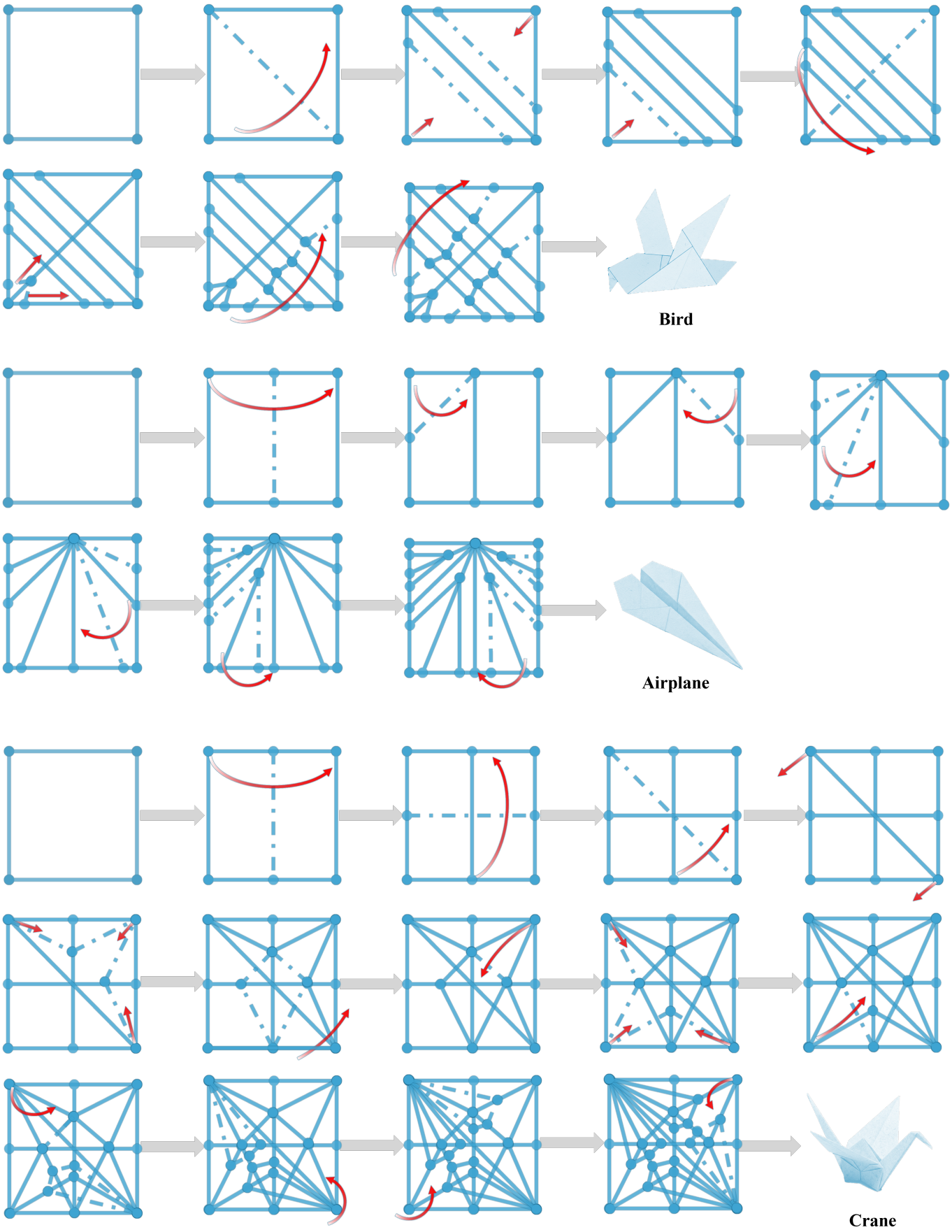

Fig. 6. **Learn2Fold results.**

References

Bo Ai, Stephen Tian, Haochen Shi, Yixuan Wang, Tobias Pfaff, Cheston Tan, Henrik I. Christensen, Hao Su, Jiajun Wu, and Yunzhu Li. 2025. A review of learning-based dynamics models for robotic manipulation. *Science Robotics* 10, 106 (2025), eadt1497. arXiv:https://www.science.org/doi/pdf/10.1126/scirobotics.adt1497 doi:10.1126/scirobotics.adt1497

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided Open Vocabulary Image Captioning with Constrained Beam Search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, Copenhagen, Denmark, 936–945. doi:10.18653/v1/D17-1098

Marshall Bern and Barry Hayes. 1996. The Complexity of Flat Origami. In *Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 175–183.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. 2024a. Genie: Generative Interactive Environments. arXiv:2402.15391 [cs.LG] https://arxiv.org/abs/2402.15391

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. 2024b. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*.

Zhaoxi Chen, Guangcong Wang, and Ziwei Liu. 2023. Scenedreamer: Unbounded 3d scene generation from 2d image collections. *IEEE transactions on pattern analysis and machine intelligence* 45, 12 (2023), 15562–15576.

Erik D Demaine and Joseph O'Rourke. 2007. *Geometric Folding Algorithms: Linkages, Origami, Polyhedra*. Cambridge University Press.

Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. 2022. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances in neural information processing systems* 35 (2022), 31841–31854.

Amanda Ghassaei, Erik D Demaine, and Neil Gershenfeld. 2018. Fast, Interactive Origami Simulation using GPU Compute Shaders. In *Proceedings of the 7th International Meeting on Origami in Science, Mathematics and Education (OSME7)*, Vol. 4. 1151–1166.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. 2020. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations (ICLR)*.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2019. Learning Latent Dynamics for Planning from Pixels. arXiv:1811.04551 [cs.LG] https://arxiv.org/abs/1811.04551

Can He, Lingxiao Meng, Zhirui Sun, Jiankun Wang, and Max Q. H. Meng. 2023. FabricFolding: Learning Efficient Fabric Folding without Expert Demonstrations. arXiv:2303.06587 [cs.RO] https://arxiv.org/abs/2303.06587

Yanjia Huang, Xianshun Jiang, Xiangbo Gao, Mingyang Wu, and Zhengzhong Tu. 2025. VISTAv2: World Imagination for Indoor Vision-and-Language Navigation. arXiv:2512.00041 [cs.RO] https://arxiv.org/abs/2512.00041

Thomas C. Hull. 2002. The Combinatorics of Flat Folds: A Survey. In *Origami^3: Third International Meeting of Origami Science, Mathematics, and Education*, Thomas C. Hull (Ed.). A K Peters, 29–38.

Justin Johnson, Agrim Gupta, and Li Fei-Fei. 2018. Image Generation from Scene Graphs. arXiv:1804.01622 [cs.CV] https://arxiv.org/abs/1804.01622

Yushi Lan, Fangzhou Hong, Shangchen Zhou, Shuai Yang, Xuyi Meng, Yongwei Chen, Zhaoyang Lyu, Bo Dai, Xingang Pan, and Chen Change Loy. 2025. LN3DIFF++: Scalable Latent Neural Fields Diffusion for Speedy 3D Generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025), 1–18. doi:10.1109/tpami.2025.3633073

Robert J Lang. 2011. *Origami Design Secrets: Mathematical Methods for an Ancient Art* (2 ed.). CRC Press.

Robert Lee, Jad Abou-Chakra, Fangyi Zhang, and Peter Corke. 2024. Learning fabric manipulation in the real world with human videos. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 3124–3130.

Weiyu Li, Xuanyang Zhang, Zheng Sun, Di Qi, Hao Li, Wei Cheng, Weiwei Cai, Shihao Wu, Jiarui Liu, Zihao Wang, Xiao Chen, Feipeng Tian, Jianxiong Pan, Zeming Li, Gang Yu, Xiangyu Zhang, Daxin Jiang, and Ping Tan. 2025a. Step1X-3D: Towards High-Fidelity and Controllable Generation of Textured 3D Assets. arXiv:2505.07747 [cs.CV] https://arxiv.org/abs/2505.07747

Yinxiao Li, Yonghao Yue, Danfei Xu, Eitan Grinspun, and Peter K Allen. 2015. Folding deformable objects using predictive simulation and trajectory optimization. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 6000–6006.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. 2025b. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608* (2025).

Yiming Liu, Lijun Han, Enlin Gu, and Hesheng Wang. 2025a. Learning a General Model: Folding Clothing with Topological Dynamics. *arXiv preprint arXiv:2504.20720* (2025).

Yunlong Liu, Shuyang Li, Pengyuan Liu, Yu Zhang, and Rudi Stouffs. 2025b. From Pixels to Predicates Structuring urban perception with scene graphs. *arXiv preprint arXiv:2512.19221* (2025).

Yuanxun Lu, Jingyang Zhang, Shiwei Li, Tian Fang, David McKinnon, Yanghai Tsin, Long Quan, Xun Cao, and Yao Yao. 2024. Direct2. 5: Diverse text-to-3d generation via multi-view 2.5 d diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8744–8753.

Jeremy Maitin-Shepard, Marco Cusumano-Towner, Jinna Lei, and Pieter Abbeel. 2010. Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *2010 IEEE International Conference on Robotics and Automation*. 2308–2315. doi:10.1109/ROBOT.2010.5509439

Gimin Nam, Mariem Khlifi, Andrew Rodriguez, Alberto Tono, Linqi Zhou, and Paul Guerrero. 2022. 3d-ldm: Neural implicit 3d shape generation with latent diffusion models. *arXiv preprint arXiv:2212.00842* (2022).

Ava Pun, Kangle Deng, Ruixuan Liu, Deva Ramanan, Changliu Liu, and Jun-Yan Zhu. 2025. Generating physically stable and buildable brick structures from text. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14798–14809.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. 2020. Offline Reinforcement Learning from Images with Latent Space Models. arXiv:2012.11547 [cs.LG] https://arxiv.org/abs/2012.11547

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. arXiv:2204.06125 [cs.CV] https://arxiv.org/abs/2204.06125

Marc Rigter, Tarun Gupta, Agrin Hilmkil, and Chao Ma. 2024. AVID: Adapting Video Diffusion Models to World Models. arXiv:2410.12822 [cs.CV] https://arxiv.org/abs/2410.12822

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs.CV] https://arxiv.org/abs/2112.10752

Tomohiro Tachi. 2009. Simulation of Rigid Origami. In *Origami^4: Fourth International Meeting of Origami Science, Mathematics, and Education*. AK Peters/CRC Press, 175–187.

Tomohiro Tachi. 2010. Freeform Variations of Origami. *Journal for Geometry and Graphics* 14, 2 (2010), 203–215.

ByteDance Seed Team. 2023. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. arXiv:2307.15818 [cs.RO] https://arxiv.org/abs/2307.15818

Gemini Robotics Team. 2025a. Gemini Robotics: Bringing AI into the Physical World. arXiv:2503.20020 [cs.RO] https://arxiv.org/abs/2503.20020

OpenAI Team. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] https://arxiv.org/abs/2303.08774

Qwen Team. 2025b. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] https://arxiv.org/abs/2505.09388

Tongxuan Tian, Haoyang Li, Bo Ai, Xiaodi Yuan, Zhiao Huang, and Hao Su. 2025. Diffusion Dynamics Models with Generative State Estimation for Cloth Manipulation. *arXiv preprint arXiv:2503.11999* (2025).

Vikram Voleti, Chun-Han Yao, Mark Boss, Adam Letts, David Pankratz, Dmitry Tochilkin, Christian Laforte, Robin Rombach, and Varun Jampani. 2024. Sv3d: Novel multi-view synthesis and 3d generation from a single image using latent video diffusion. In *European Conference on Computer Vision*. Springer, 439–457.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in neural information processing systems* 36 (2023), 8406–8441.

Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5410–5419.

Kun Yan, Lei Ji, Huaishao Luo, Ming Zhou, Nan Duan, and Shuai Ma. 2021. Control Image Captioning Spatially and Temporally. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, Online, 2014–2025. doi:10.18653/v1/2021.acl-long.157

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-Language Models for Vision Tasks: A Survey. arXiv:2304.00685 [cs.CV] https://arxiv.org/abs/2304.00685