ScissorBot: Learning Generalizable Scissor Skill for Paper Cutting via Simulation, Imitation, and Sim2Real

Jiangran Lyu[1,2] **Yuxing Chen**[1,2] **Tao Du**[3] **Feng Zhu**[2] **Huiquan Liu**[2] **Yizhou Wang**[1] **He Wang**[1,2,†]
[1]CFCS,School of Computer Science, PKU [2]Galbot [3]IIIS,THU

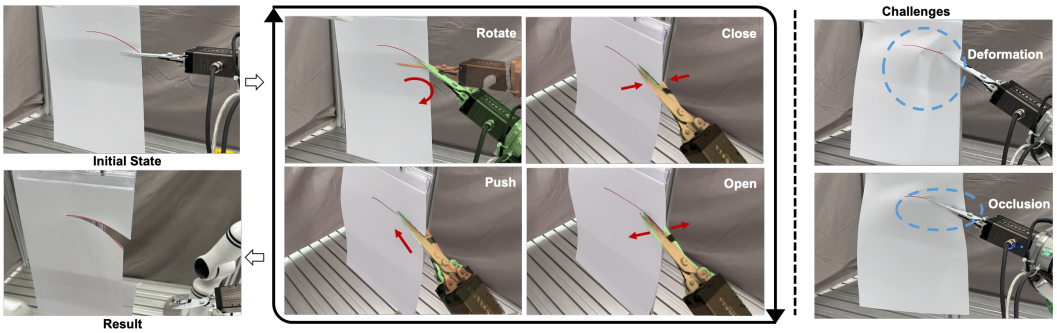


Figure 1: **Robotic paper cutting with scissors. Left:** The objective is to drive scissors to accurately cut curves drawn on the paper, which is hung with the top edge fixed. **Middle:** Our execution follows an action primitive sequence, namely *Rotate*, *Close*, *Open*, *Push*. The meticulous action, visualized as scissors before (orange) and after (green) each action, ensures accurate cutting in the real world. **Right:** During execution, large deformation of paper and severe occlusion between scissors and target curves occasionally occurs. **Please refer to the video in the supplementary materials.**

Abstract:

This paper tackles the challenging robotic task of generalizable paper cutting using scissors. In this task, scissors attached to a robot arm are driven to accurately cut curves drawn on the paper, which is hung with the top edge fixed. Due to the frequent paper-scissor contact and consequent fracture, the paper features continual deformation and changing topology, which is diffult for accurate modeling. To deal with such versatile scenarios, we propose ScissorBot, the first learning-based system for robotic paper cutting with scissors via simulation, imitation learning and sim2real. Given the lack of sufficient data for this task, we build PaperCutting-Sim, a paper simulator supporting interactive fracture coupling with scissors, enabling demonstration generation with a heuristic-based oracle policy. To ensure effective execution, we customize an action primitive sequence for imitation learning to constrain its action space, thus alleviating potential compounding errors. Finally, by integrating sim-to-real techniques to bridge the gap between simulation and reality, our policy can be effectively deployed on the real robot. Experimental results demonstrate that our method surpasses all baselines in both simulation and real-world benchmarks and achives performance comparable to human operation with a single hand under the same conditions.

Keywords: Deformable Object Manipulation, Imitation Learning, Sim-to-Real

1 Introduction

Paper cutting, an ancient craft dating back to at least the 6th century [1], has evolved alongside human civilization, serving as a medium for emotional and symbolic expression [2]. In modern society, it has wide applications ranging from decorative art [3] and education to advanced manufacturing and technology [4, 5]. Humans can use scissors to perform paper cutting, showcasing their dexterity

[1]† Corresponding Author (hewang@pku.edu.cn)

in tool usage. However, robots have yet to master this generalizable cutting skill, which indicates using scissors to cut painted patterns on paper with visual observation. The main obstacle lies in the intricate interaction between paper and scissors, characterized by the continual deformation and changing topology of the paper. Accurately modeling this dynamics based on first principles and achieving precise control using Linear Quadratic Regulator or Model Predictive Control is highly challenging. In contrast, learning-based methods, benefiting from data-driven, offer a promising alternative without the need for explicit modeling of this complex system and have the potential to achieve generalization across diverse cutting tasks.

Despite the strengths of learning-based approaches for many robotic tasks [6, 7, 8, 9], they fall short on this task partly due to the following challenges. First, there is insufficient data for paper-cutting tasks so far. Collecting accurate scissor-cutting demonstrations directly on a real-world robot system is laborious and hazardous, making simulation essential for data generation. However, existing simulators for thin-shell objects [10, 11, 12] do not support detailed interaction with scissors including contact and consequent fracture. Second, learning a generalizable and accurate scissor policy is intrinsically difficult for this paper cutting task. Given its contact-rich and deformable nature, even millimetric movements of the scissors can induce significant bending (as illustrated in Fig. 1) or lead to curves deviating substantially from the intended target. Meanwhile, the scissors occasionally occlude the target curve, leading to an ill-posed decision-making problem. In this scenario, reinforcement learning methods struggle with poor data efficiency. Third, the significant sim-to-real gap, both physical and visual, hinders the deployment of the policy in the real world. The physical gap, arising from the complex factors in the real world, sometimes leads to deviations in the cutting trajectory during execution. Additionally, the visual gap for this task is mainly due to the *edge bleeding artifact* [13], where real-world sensor depth blurs at object edges. This blurring causes jittery observations of the scissor-paper interaction, leading to incorrect policy decisions.

To address the above challenges, we introduce ScissorBot, the first learning-based robotic system for paper cutting with scissors via a combination of simulation, imitation learning and sim2real techniques. To mitigate the data scarcity issue, we develop PaperCutting-Sim, a paper-cutting simulator supporting interactive fracture coupling with scissors, enabling large-scale demonstration generation with a heuristic-based oracle policy. This oracle policy leverages privileged information that is inaccessible in the real world, allowing us to distill its knowledge into a vision-based imitation learning policy. To handle tasks of significant complexity with improved efficiency, we customize an action primitive sequence which constrains the action space of learning and allivating potential compounding errors. We also use multi-frame point clouds as input to complete the occluded and underlying dynamics information. To bridge the physical and visual gaps between simulation and reality, we propose data augmentation on deviation correction and artifact mimicry, respectively. The former method can adaptively correct compounding errors via adding out-of-distribution data which pairs deviation states with corresponding correction action. The latter aims to mimic edge bleeding artifact in the simulation to achieve visual alignment with reality.

Through extensive simulated and real-world experiments, we evaluate the efficacy and generalizability of our learning policy across three different task difficulty levels: *Easy*, *Middle*, and *Hard*. Our method improves cutting accuracy by at least fivefold compared to the best alternative methods, as measured by Chamfer distance. Furthermore, despite training only on *Easy* data from simulation, our method achieves a Chamfer distance of 2mm and an IoU of 89 for *Middle* and *Hard* patterns in the real world, a performance that is comparable to humans with single-hand operation. Our research opens up new oppotunities for contact-rich and fine manipulation of deformable objects.

2 Related Work

2.1 Deformable Object Manipulation

The manipulation of deformable objects, such as dough [14, 15], cloth [16, 8] and rope [17, 11] has been extensively studied in the in the scentific and engineering disciplines. Zhao et al. [18] train robots in the real world to learn paper-flipping skills, and Namiki et al. [19] explore paper folding using motion primitives. Other studies focus on kirigami [20, 21], the traditional art of

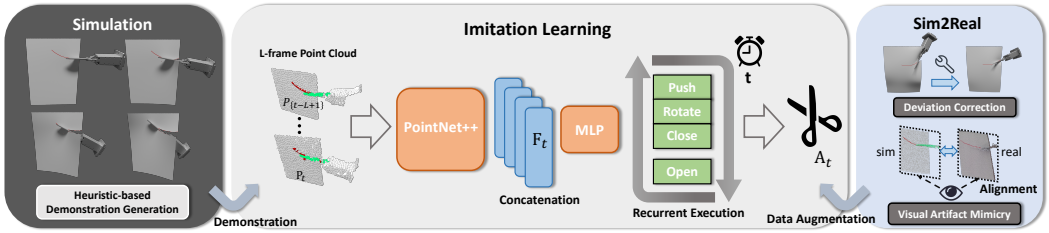


Figure 2: **An overview of the learning system.** The system first generates expert demonstrations in our built simulation which supports interactive fracture of the paper. These demonstrations are then used to train a vision-based imitation learning policy that inputs multi-frame point clouds (Blade point cloud is highlighted in green only for visualization) and outputs parameters of action primitive. Meanwhile, Deviation Correction and Visual Artifact Mimicry provide data augmentation to imitation learning which ensures a robust transfer from simulation to real world.

folding and cutting paper to create intricate designs. For paper cutting, these studies typically use desktop cutting plotters rather than fully automated robotic systems. Additionally, various robotic systems have been developed for cutting deformable objects in different domains, such as vegetables [22, 23], dough [14, 15], and soft objects with rigid cores [24]. However, these systems generally employ tabletop knife cutting, which differs from our approach of using scissors for cutting.

2.2 Simulation Environments for Paper Cutting

One line of works build simulators to boost robotic skill learning for thin-shell materials [11, 12, 25, 10], however they couldn't simulate the fracture during the scissors cutting. Some works focus on simulating the cutting process of soft materials [26, 24]. Other works studies paper fracture process either from the theoretical analysis [27, 28], re-meshing algorithm [29, 30] or its application in kirigami [31, 32]. Overall, none of the existing works implements the paper cutting simulation for robot learning, which combines dynamic modeling of paper and interactive fracture interweaving paper remeshing according to scissor motion.

2.3 Imitation Learning

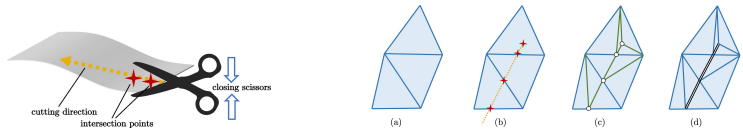
Imitation Learning (IL) [33, 6, 7, 34] is a supervised learning methodology for training embodied agents using expert demonstrations. The commonly used Behavior Cloning (BC) [33] strategy directly trains the policy to imitate expert actions. Despite its simplicity, this approach has demonstrated remarkable effectiveness in robotic manipulation [35, 36]. In this paper, we adopt imitation learning and ultilze action primitive sequence to ensure robustness during execution.

3 Method

3.1 Task Formulation and Method Overview

The objective of this task is to accurately cut paper along drawn curves using scissors, guided by single-view point cloud input. The scissors are mounted on a single robotic arm, with the top edge of the paper fixed and the bottom edge free, enabling generalization to various scenarios.

In order to learn such generalizable scissor skills, we introduce ScissorBot, a robotic system designed to learn visuo-motor policies via simulation, imitation learning and sim2real techniques. As the system depicted in Fig. 2, we first present our training data source in Sec. 3.2, where we develop the first paper-cutting simulator, PaperCutting-Sim, and heuristic-based demonstrations generation in Sec. 3.3. We then detail the vision-based imitation learning design in Sec. 3.4 and sim2real techniques in Sec. 3.5. Finally, we present a hardware setup for deploying our policy in real-world scenarios in Sec. 3.6.



(1) Detection of intersection points (2) Remeshing

Figure 3: **Interactive Fracture in our PaperCutting-Sim.** (1): As the scissors close, the fracture occurs along the cutting direction. Intersection points (**red star**) can be computed from edge-edge detection and vertex-face detection between the cutting direction (**orange dashed**) and the paper mesh. (2): (a) The original paper mesh (**blue triangles**). (b) Intersection point (**red star**) and cutting direction (**orange dashed**). (c) According to the intersection points, new vertices are added on the existing edges and the endpoint is inserted inside the triangle. The new edges (**green solid**) are connected between the new inserted vertex and the opposite vertex in the triangle. (d) The edges between these newly added vertices are split into two pieces (**black solid**).

3.2 PaperCutting-Sim

We build a paper-cutting simulator, PaperCutting-Sim, to support the modeling of both paper and scissors, as well as their contacts and the consequent fracture. The simulator is implemented in Python and Taichi [37], which supports parallel computation on GPUs.

Dynamic Model. Following the Kirchhoff-Love shell theory, we model the elastic energy of the paper as the sum of stretching elastic energy and bending elastic energy. The stretching elastic energy is modeled as co-rotational linear elasticity, and the bending elastic energy is calculated using squared difference of mean curvature [38]. We perform spatial discretization using the Finite Element Method [29], and update positions and velocities through implicit time integration [39], which optimizes the incremental potential via Newton's method. To model the contact between the paper and the scissors, we represent the scissors using a signed distance field, utilize the cubic of the signed distance to calculate collision energy, and apply Coulomb friction.

Interactive Fracture. Different from cutting simulation with predefined fracture surfaces [26], interactively handling fracture coupled with contact is a non-trivial problem. To address this, we design a two-phase geometry-based approach, as illustrated in Fig. 3. First, during the closing process of the scissors, we propose using edge-edge detection and vertex-face detection to detect the intersection points between the cutting trajectory and the paper mesh. Then these intersection points are added to the paper mesh and related edges are connected and split according to the vertex position relationship inside triangles. We refer the reader to Appendix B for more details.

3.3 Demonstration Generation

In this section, we devise an action primitive sequence and a heuristic-based Oracle policy for large-scale demonstration generation. For distillation, we preserve high-quality demonstrations measured by chamfer distance.

Action Primitive Sequence. We designed four action primitives for the scissors in a heuristic manner, namely *Open*, *Push*, *Rotate*, and *Close*. These primitives can be combined into a sequence to cut a straight line on the paper as follows: (1) Open the scissors to the maximum extent. (2) Push the scissors to the starting point of the line. (3) Rotate the scissors towards the endpoint of the line. (4) Close the scissors breaking the paper. Note that the pushing action is a 1D translation along the cutting direction as we approximate that the starting point is in the scissor cut direction.

Oracle Policy. The oracle policy initially discretizes the target smooth curve into several line segments, with this approximation scarcely impacting visual appearance. Subsequently, the entire curve can be cut by multiple action sequences for line segments iteratively. As oracle policy can access the 3D position of target line during each step, the pushing distance ($p \in \mathbb{R}^1$), rotation matrix ($\mathbf{R} \in \mathrm{SO}(3)$), and closed angle ($c \in \mathbb{R}^1$) can be computed by relative position between scissors and target line. Please refer to Appendix C for more details.

3.4 Vision-based Imitation Learning

This section presents the design of our learning framework, with multi-frame point clouds as input and action parameters as ouput. The complete network architecture is depicted in Fig. 2.

Spatial-Temporal Observation Encoding. We first pre-process raw single-view point cloud using bounding-box cropping and FPS sampling. Then sequential L-frame point clouds $\{P_{t-i-1}\}_{i=1}^{L}$, along with a binary mask indicating whether a visible point originates from the target curve, are fed into a shared PointNet++ encoder [40] to obtain features $\{\mathcal{F}_{t-i+1}\}_{i=1}^{L}$. These features are concatenated and passed through a shallow MLP to regress actions parameters.

Primitive Learning. In contrast to selecting the direct 7 DoF scissor pose as our output action space, the output action parameters are associated with the designed action primitives mentioned in Sec. 3.3. These actions are recurrently executed for each stage which keeps the order of *Push*, *Rotate*, *Close*, *Open* repeatedly. We employ Mean Squared Error (MSE) loss for the *Push* and *Close* terms, and 9D L1 Loss [41] for *Rotate*. The overall loss is formulated as:

$$\mathcal{L} = \lambda_p(p - \hat{p})^2 + \lambda_c(c - \hat{c})^2 + \lambda_{\mathbf{R}} \sum_{i,j} |\mathbf{R}_{ij} - \hat{\mathbf{R}}_{ij}| \tag{1}$$

where λ_p, λ_c, λ_R are respective weights, and $\hat{p}$, $\hat{c}$, $\hat{\mathbf{R}}$ are ground truth action values.

3.5 Sim-to-Real Transfer

Deviation Correction. We introduce deviation correction to enhance the robustness for drifting scenarios which somestimes occurs in the real world. In this approach, we fine-tune the trained model using correction data, which comprises out-of-distribution states paired with corrective actions. These data are generated by introducing random rotation perturbances to the action during oracle policy execution. As the oracle policy consistently cuts towards the endpoint of each line segment, the next action naturally corrects minor drifting errors.

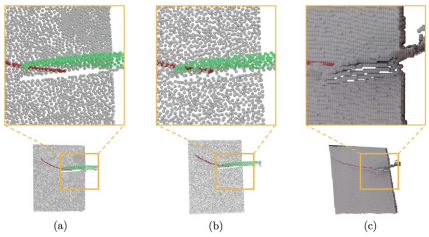


Figure 4: **Visualization of Visual Artifact Mimicing** (a) Perfect point cloud in simulation with scissors blade highlighted (**green points**) (b) Point cloud with our proposed visual artifact mimicry. (c) Point cloud captured in the real world with artifact.

Visual Artifact Mimicry. We propose a simple yet effective method to mimic *edge bleeding artifact* in the simulation. To create continuous value at the edge between foreground and background in simulated depth image, we preprocess the depth with an average pooling kernel and add random noise perpendicular to the surface of the paper to the point cloud of the blade. In this way, the artifact can be mimicked (Fig. 4(b)) in our training data thus reaching a visual alignment from simulation to reality.

3.6 Hardware System Design

We design a hardware system for the paper-cutting task, as Fig. 5 shown. The setup includes a Realman robot equipped with a scissor extension for manipulation and a single Kinect DK camera to capture RGBD observations. To secure the paper, we use plastic clips to fix the top edge, leaving the lower edge free. The target curves are drawn in red on white paper, with corresponding binary masks obtained through simple RGB-based segmentation. We use A4 printer paper ($210\,\mathrm{mm} \times 297\,\mathrm{mm}$, $75\,\mathrm{g/m^2}$) as the material for the following experiments.

4 Experiments

In this section, we first evaluate the cutting performance of our proposed method through comparison with various baselines and variants in simulated environments. We further validate our approach in the real-world. Please refer to Appendix A for abaltion studies and more qualitative results.

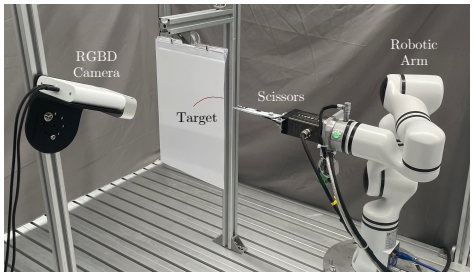


Figure 5: Hardware System

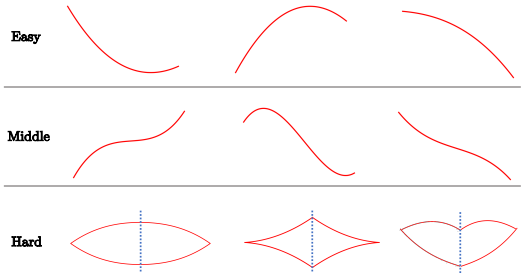


Figure 6: Example curves for Easy, Middle and Hard tracks.

4.1 Benchmark

Task Datasets. We focus on simple smooth curve cutting and split it into three distinct tracks, illustrated in Fig. 6. In each track, curves are generated using Bézier curves parameterized by four control points. By manipulating the positional relationship of these control points, we can control the second-order derivative of the curve, which in turn determines the complexity of the scissors' motion. The discussion on non-smooth and non-simple curves can be found in the Sec. 5.

- *Easy*: In this track, the second-order derivatives of curves are consistently positive or negative.
- *Middle*: Curves in this track exhibit varying positive and negative second-order derivatives.
- *Hard*: This track comprises several patterns, each composed of two curves from the *Easy* track. In real-world settings, this track can be further required to cut the origami sheet to obtain an axisymmetric closed-shape pattern.

To demonstrate the generalization capability of our policy, our training set consists of approximately 5k trajectories solely from the Easy track. There are 100 curves of each track for testing.

Evaluation Metrics. In our evaluation process, we utilize various metrics to gauge the quality of our results across different difficulty levels. For the all three tracks, we employ the chamfer distance as a measure of deviation between the cropped curve and the target curve. Additionally, we report the Recall metric under different thresholds of chamfer distance, indicating the proportion of well-cut instances. For trials completing closed shapes in the *Hard* track, we further assess the quality by calculating the mean Intersection over Union (mIoU) between the cropped pattern and the target pattern, providing a comprehensive measure of similarity and accuracy.

4.2 Policy Evaluation in Simulation

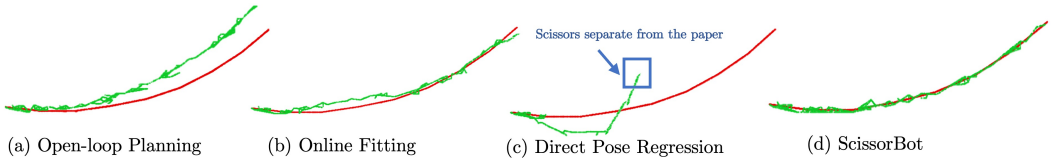
Non-learning Baselines.

- *Open-loop Planning* detects the target goal curve prior to cutting. Then it discretizes the detected curve into isometric line segments and plans the scissor translation and rotation at each step.
- *Online Fitting* employs an step-by-step line fitting utilizing RANSAC. The fitted line target from the captured point cloud determines the movement distance and scissor rotation at each step.

Learning based Baselines.

- *Direct Pose Regression*. This methodology directly regresses the 7 Degrees of Freedom (DoF) scissor pose (6D Pose and 1D joint angle) .
- *Action Chunking* [7]. In this policy, actions for the next k timesteps are predicted. The current action to execute is determined from weighted averages across the previous overlapping action chunk. We adopt the implementation from [7].

Comparison to Non-learning based Baselines. Our method exhibits significantly superior cutting accuracy when compared to non-learning based approaches. As depicted in Fig. 7, the cutting trajectories of *Open-loop Planning* substantially deviates from the target curve. The reason lies in that *Open-loop Planning* lacks adaptability to environmental changes and accumulates a lot of errors during the highly non-linear interaction between scissors and paper. Although *Online Fitting*



(a) Open-loop Planning (b) Online Fitting (c) Direct Pose Regression (d) ScissorBot

Figure 7: **Visualization of Cutting Results on UV plane of the paper.** Target curves are in red while cropped lines by scissors are in green.

Methods	Easy			Middle			Hard	
	Chamfer (mm)	Recall@1.5	Recall@5.0	Chamfer (mm)	Recall@1.5	Recall@5.0	Chamfer (mm)	mIoU
Open-loop Planing	10.8	9.3	25.0	6.8	10.5	36.3	18.1	31.2
Online Fitting	5.5	31.4	73.6	5.3	21.0	53.1	10.3	63.0
Ours	**1.1**	**85.1**	**98.6**	**1.5**	**79.6**	**96.6**	**1.9**	**91.3**
Oracle	1.4	83.1	98.2	1.4	80.1	98.2	1.9	92.2

Table 1: Comparison with non-learning based baselines in simulation.

demonstrates some adaptability by adjusting its cutting action based on current observations, it falls short of making optimal decisions, particularly in instances of occasional occlusion. In contrast, our learning policy showcases remarkable adaptability to dynamic conditions, resulting in notably precise operations, with a chamfer distance of $1.1\,\mathrm{mm}$ compared to $10.8\,\mathrm{mm}$ and $5.5\,\mathrm{mm}$ for *Open-loop Planning* and *Online Fitting* respectively.

Effect of the Primitive Learning. The comparison with learning-based baselines exhibits the effectiveness of our primitive learning. As shown in Fig. 7(c), the alternative frequently causes separation between scissors and paper and thus fails to perform the paper-cutting tasks to the end. We only report its chamber distance with the first one-third of the target curve in Tab.2. Even incorporating *Action Chunking*, designed to mitigate compounding errors, there is no substantial improvement in completing the target curves. This is due to the drastic state transitions at each step, which complicate the accurate prediction of future actions. These results highlight the highly nonlinear nature of the task. In contrast, our designed primitive constraints the action space of scissors, which minimizes possible errors during execution.

Method	Chamfer (mm)	Recall@1.5	Recall@5.0
Direct Pose Regression	11.2*	9.0*	24.4*
Action Chunking [7]	11.4*	8.9*	25.1*
Ours	1.1	85.1	98.6
Ours + Action Chunking [7]	1.2	84.2	98.5

Table 2: Comparison with learning-based baselines in simulation. For *Direct pose regression* and its variance, we only calculate its chamber distance with the first one-third of the target curve (*).

Generalization to Novel Curves and Patterns. Middle and Hard targets pose additional challenges due to more varied patterns, leading to more complex deformation and fratcure during cutting. Despite being trained on the *Easy* track of curves, our policy demonstrates robustness and achieves performance comparable to the oracle policy when handling *Middle* and *Hard* targets. This success is due to its generalizability from careful system design.

4.3 Policy Evaluation in the Real World

We evaluate the performance of our sim2real model on a real-world platform. Quantitative and qualitative results are presented in Table 3 and Fig. 8, Fig. 10, respectively.

Experiment Details. To understand the capability of our system, we include experimental results from human operations with single hand using standard scissors. We further introduce the concept of *Finished rate*, subjectively defining "Finished" for trials which executes smoothly without falling into tearing, folding, or significant deviation from the target. After each trial for both policy and human, we capture the cropped paper using an RGBD camera and compute metrics such as chamfer distance and IoU. We report the median and extremum of these metrics in the form of $x \pm u$.

Result Analysis. As evidenced in Table 3, our policy consistently fails without any sim2real, as a result of confused perception with the interactions between blades and paper. Our mimicry strategy mitigates the visual gap and minimize erratic action prediction, thereby achieve successful deploy-

Methods	Easy		Middle		Hard	
	Finished Rate	Chamfer (mm)	Finished Rate	Chamfer (mm)	Finished Rate	IoU
Human	10/10	2 ± 1	10/10	2 ± 1	10/10	92 ± 3
Ours w/o Visual Artifact Mimicry	0/10	-	0/10	-	0/10	-
Ours w/o Deviation Correction	7/10	$\mathbf{2 \pm 1}$	7/10	3 ± 1	7/10	84 ± 8
Ours	**9/10**	$\mathbf{2 \pm 1}$	**8/10**	$\mathbf{2 \pm 1}$	**8/10**	$\mathbf{89 \pm 5}$

Table 3: Quantitative results in the real world.

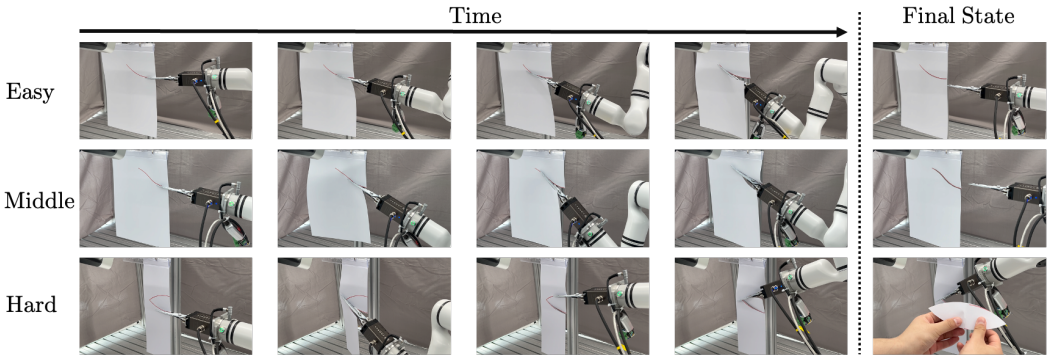
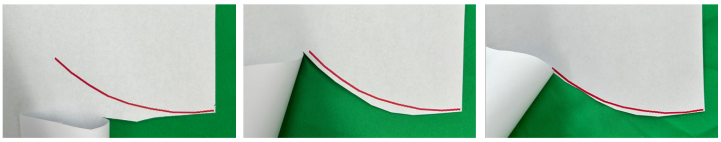


Figure 8: Realworld cutting process on Easy, Middle and Hard tracks.



(a) Difting Failure (b) Persistent Error (c) Deviation Correction

Figure 9: **Qualatative result of our Deviation Correction.** Without using Deviation Correction, policy trained from simulation frequently falls into drifting failures (a) or keeps perisitent errors (b). Our Deviation Correction (c) leads to a accurate and robust deployment.

ment. However, the performance is still still unsatisfactory. As illustrated in Fig. 9, policies sometimes experience drifting failures (Fig. 9a) or persistently exhibit errors (Fig. 9b). To this end, the correction mechanism addaptively corrects the deviation and enhances the stability and accuracy of the deployed policy. For example, it enhances the finished rate from 7/10 to 9/10 in the "Easy" track and reducing deviation by $1\,\mathrm{mm}$ in the Middle track. Combining the above sim2real techniques, our system achieves comparable performance to human single-hand manipulation under same condition, which has only $2\,\mathrm{mm}$ error from the target curve. Furthermore, it achieves an IoU of 89 on the *Hard* track, which is particularly challenging due to more drastic bending.

5 Conclusion, Limitations and Future Directions

We introduce ScissorBot, the first learning-based robotic system for generalizable paper cutting using scissors. The system utilizes demonstrations collected in our newly developed paper-cutting simulator to train a primitive-based imitation learning policy and combines sim2real techniques to achieve robust deployment in the real world. Extensive experiments exhibit the generalizability and accuracy of our system on simple smooth curves which cover most of cutting scenarios. However, scissor cutting for non-simple or non-smooth curves is still an open problem. Non-smooth curves like zig-zag patterns must result in the sudden relative pose change frequently for scissors while non-simple curves will result in a circle in the scissor trajectory. Both two kinds of scissors' motion can't be achieved considering our single-arm workspace. To solve these challenging situations, it may be a dexterous dual-arm cooperation system containing one arm holding and rotating the paper and the other manipulating scissors. We leave this bimanual hardware system with policy design as future works.

Acknowledgments

If a paper is accepted, the final camera-ready version will (and probably should) include acknowledgments. All acknowledgments go at the end of the paper, including thanks to reviewers who gave useful comments, to colleagues who contributed to the ideas, and to funding agencies and corporate sponsors that provided financial support.

References

[1] M. Sullivan. *Art and Artists of Twentieth-Century China*. University of California Press, 2023. ISBN 9780520911611. URL https://books.google.com.hk/books?id=G0_hEAAAQBAJ.

[2] T. Shen. Analysis of symbolic meanings of folk paper-cut from the perspective of semiotics. In *Proceedings of the 2018 2nd International Conference on Management, Education and Social Science (ICMESS 2018)*, pages 67–69. Atlantis Press, 2018/06. ISBN 978-94-6252-520-7. doi:10.2991/icmess-18.2018.14. URL https://doi.org/10.2991/icmess-18.2018.14.

[3] F. Temko. *Kirigami Home Decorations*. Tuttle Publishing, 2012.

[4] Y. Yang, K. Vella, and D. P. Holmes. Grasping with kirigami shells. *Science Robotics*, 6(54): eabd6426, 2021.

[5] M. Cianchetti, C. Laschi, A. Menciassi, and P. Dario. Biomedical applications of soft robotics. *Nature Reviews Materials*, 3(6):143–153, 2018.

[6] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[7] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[8] R. Wu, C. Ning, and H. Dong. Learning foresightful dense visual affordance for deformable object manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10947–10956, 2023.

[9] W. Wan, H. Geng, Y. Liu, Z. Shan, Y. Yang, L. Yi, and H. Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3891–3902, 2023.

[10] Y. Wang, J. Zheng, Z. Chen, Z. Xian, G. Zhang, C. Liu, and C. Gan. Thin-shell object manipulations with differentiable physics simulations. *arXiv preprint arXiv:2404.00451*, 2024.

[11] X. Lin, Y. Wang, J. Olkin, and D. Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 432–448. PMLR, 2021.

[12] S. Chen, Y. Xu, C. Yu, L. Li, X. Ma, Z. Xu, and D. Hsu. Daxbench: Benchmarking deformable object manipulation with differentiable physics. In *The Eleventh International Conference on Learning Representations*, 2022.

[13] S.-H. Park, G.-R. Park, and K.-R. Baek. Edge bleeding artifact reduction for shape from focus in microscopic 3d sensing. *Sensors*, 23(20):8602, 2023.

[14] X. Lin, Z. Huang, Y. Li, J. B. Tenenbaum, D. Held, and C. Gan. Diffskill: Skill abstraction from differentiable physics for deformable object manipulations with tools. In *10th International Conference on Learning Representations, ICLR 2022*, 2022.

[15] X. Lin, C. Qi, Y. Zhang, Z. Huang, K. Fragkiadaki, Y. Li, C. Gan, and D. Held. Planning with spatial-temporal abstraction from point clouds for deformable object manipulation. In *Conference on Robot Learning*, pages 1640–1651. PMLR, 2023.

[16] P. M. Scheikl, N. Schreiber, C. Haas, N. Freymuth, G. Neumann, R. Lioutikov, and F. Mathis-Ullrich. Movement primitive diffusion: Learning gentle robotic manipulation of deformable objects. *IEEE Robotics and Automation Letters*, 2024.

[17] C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Iterative residual policy: for goal-conditioned dynamic manipulation of deformable objects. *The International Journal of Robotics Research*, 43(4):389–404, 2024.

[18] C. Zhao, C. Jiang, J. Cai, M. Y. Wang, H. Yu, and Q. Chen. Flipbot: Learning continuous paper flipping via coarse-to-fine exteroceptive-proprioceptive exploration. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10282–10288. IEEE, 2023.

[19] A. Namiki and S. Yokosawa. Robotic origami folding with dynamic motion primitives. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5623–5628. IEEE, 2015.

[20] H. Shigemune, S. Maeda, Y. Hara, U. Koike, and S. Hashimoto. Kirigami robot: Making paper robot using desktop cutting plotter and inkjet printer. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1091–1096. IEEE, 2015.

[21] R. Liu, J. Liang, S. Sudhakar, H. Ha, C. Chi, S. Song, and C. Vondrick. Paperbot: Learning to design real-world tools using paper, 2024.

[22] X. Mu, Y. Xue, and Y.-B. Jia. Robotic cutting: Mechanics and control of knife motion. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3066–3072. IEEE, 2019.

[23] X. Mu, Y. Xue, and Y.-B. Jia. Dexterous robotic cutting based on fracture mechanics and force control. *IEEE Transactions on Automation Science and Engineering*, 2023.

[24] Z. Xu, Z. Xian, X. Lin, C. Chi, Z. Huang, C. Gan, and S. Song. Roboninja: Learning an adaptive cutting policy for multi-material objects. *arXiv preprint arXiv:2302.11553*, 2023.

[25] C. Gan, J. Schwartz, S. Alter, D. Mrowca, M. Schrimpf, J. Traer, J. De Freitas, J. Kubilius, A. Bhandwaldar, N. Haber, et al. Threedworld: A platform for interactive multi-modal physical simulation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[26] E. Heiden, M. Macklin, Y. Narang, D. Fox, A. Garg, and F. Ramos. Disect: A differentiable simulation engine for autonomous robotic cutting. *Robotics: Science and Systems XVII*, 2021.

[27] K. Niskanen. Strength and fracture of paper. *Products of Papermaking*, 2:641–725, 1993.

[28] R. Seth and D. Page. Fracture resistance of paper. *Journal of Materials Science*, 9:1745–1753, 1974.

[29] T. Pfaff, R. Narain, J. M. De Joya, and J. F. O'Brien. Adaptive tearing and cracking of thin sheets. *ACM Transactions on Graphics (TOG)*, 33(4):1–9, 2014.

[30] T. Xin, P. Marris, A. Mihut, G. Ushaw, and G. Morgan. Accurate real-time complex cutting in finite element modeling. In *VISIGRAPP (1: GRAPP)*, pages 183–190, 2018.

[31] Z. Qi, D. K. Campbell, and H. S. Park. Atomistic simulations of tension-induced large deformation and stretchability in graphene kirigami. *Physical Review B*, 90(24):245437, 2014.

[32] Q. Zhang, J. Wommer, C. O'Rourke, J. Teitelman, Y. Tang, J. Robison, G. Lin, and J. Yin. Origami and kirigami inspired self-folding for programming three-dimensional shape shifting of polymer sheets with light. *Extreme Mechanics Letters*, 11:111–120, 2017.

[33] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[34] S. Belkhale, Y. Cui, and D. Sadigh. Data quality in imitation learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[35] Z. Wang, J. Chen, Z. Chen, P. Xie, R. Chen, and L. Yi. Genh2r: Learning generalizable human-to-robot handover via scalable simulation, demonstration, and imitation. *arXiv preprint arXiv:2401.00929*, 2024.

[36] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali, et al. Deep imitation learning of sequential fabric smoothing from an algorithmic supervisor. in 2020 ieee. In *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9651–9658.

[37] Y. Hu, T.-M. Li, L. Anderson, J. Ragan-Kelley, and F. Durand. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6):201, 2019.

[38] E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schröder. Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 62–67. Citeseer, 2003.

[39] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 767–778. 2023.

[40] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[41] J. Levinson, C. Esteves, K. Chen, N. Snavely, A. Kanazawa, A. Rostamizadeh, and A. Makadia. An analysis of svd for deep rotation estimation. *Advances in Neural Information Processing Systems*, 33:22554–22565, 2020.

A More Experiments and Results

A.1 More Qualatative Results

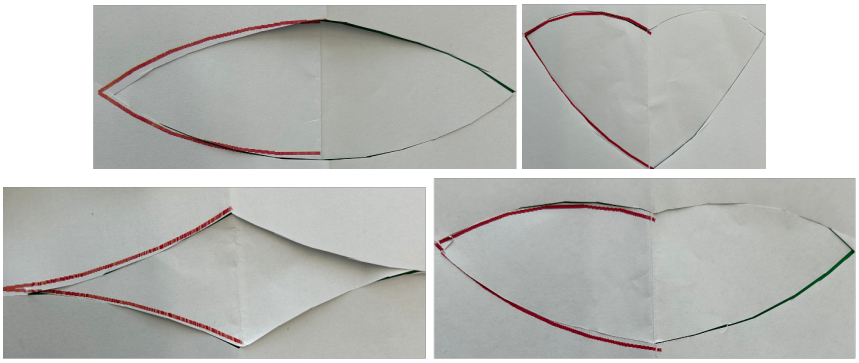


Figure 10: **Realworld results on patterns from Hard tracks.** The cropped pattern from our system has a accurate overlap with the target pattern.

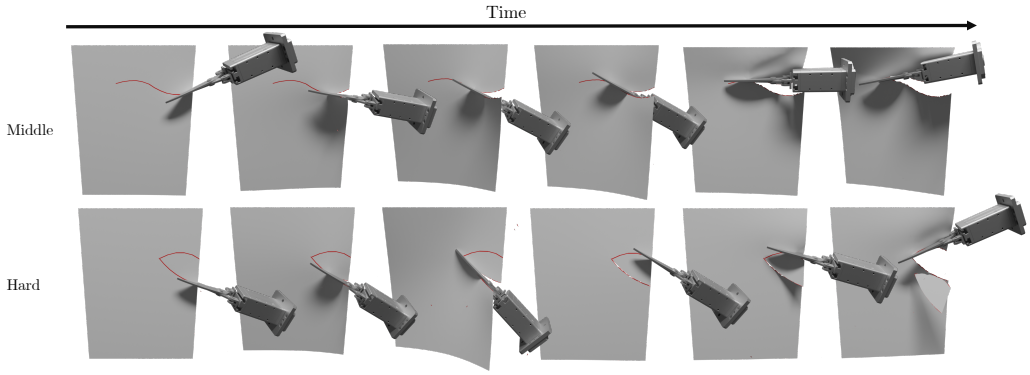


Figure 11: Cutting process of our method in the simulation on Middle and Hard tracks.

A.2 Ablation Studies

Basic length (mm)	Steps	Chamfer (mm)
20	~ 29	1.9
15	~ 37	1.4
10	~ 53	1.3

Table 4: Effects of Discretization Granularity for curves, i.e, the basic length of line segments.

In this section, we conduct experiments to analyze the effects of critical parameters and design choices.

Discretization Granularity. During the generation of expert data, smooth curves are discretized into line segments as an approximation. We aim to study the impact of discretization granularity on efficiency and quality, as reported in Table 4. It's evident that as the length of each line segment decreases, the number of execution steps increases while the quality improves. When the basic length decreases from $15\,\mathrm{mm}$ to $10\,\mathrm{mm}$, the reduction in chamfer distance is marginal, merely $0.1\,\mathrm{mm}$. Thus, we opt for a trade-off choice of $15\,\mathrm{mm}$, with a balance between efficiency and quality.

Observation Horizon (L). We analyze the effect of different observation horizons, as detailed in Table 5. Policies with 4 or 6 frame horizons outperform that without temporal inputs by approximately 10 points on Recall@1.5. The superiority of our spatial-temporal encoding lies in its robustness to

Horizon	Chamfer(mm)	Recall@1.5	Recall@5.0
w/o multi-frame	1.5	74.4	94.2
L = 2	1.6	70.4	93.1
L = 4	1.1	85.1	98.6
L = 6	1.1	85.4	98.9

Table 5: Ablation study of Temporal encoding and Observation Horizon.

Filter	Data usage	Chamfer(mm)	Recall@1.5	Recall@5.0
w/o filtering	100%	1.7	74.5	92.5
τ = 1.6	$\sim$ 90%	1.4	81.8	97.8
τ = 1.0	$\sim$ 70%	1.1	85.1	98.6
τ = 0.7	$\sim$ 30%	1.1	85.0	98.6

Table 6: Ablation study on the Filtering threshold τ.

occlusion and deformable dynamics. Although increasing the horizon from 4 to 6 yields marginal improvements in cutting quality, e.g., 0.3 for Recall@1.5, we opt for a horizon of 4 in our system to maintain a favorable balance between performance and efficiency.

Filtering Threshold (τ). We train our imitation policy using various demonstration filtering thresholds ranging from 0.7 to 1.6, as well as no filtering, as summarized in Table 6. A stricter threshold yields better performance and a lower data usage rate, necessitating the generation of more demonstrations for distillation. In our system, we select $\tau = 1.0$ to strike a balance between data quality and efficiency.

B Details of PaperCutting-Sim

We consider the occurrence of paper fractures due to the intersection of the scissor blades. As the scissors close, the intersection point of the two blades, denoted as $\mathbf{P}$, moves along the paper surface. The trajectory of this movement forms the fracture line on the paper. At each time step, we detect this moving trajectory on the paper mesh and perform remeshing, which includes vertex insertion, edge connection, and edge splitting.

New Vertex Insertion: For each time step t, the movement of $\mathbf{P}$ can be represented as a segment $E = (\mathbf{P}_t, \mathbf{P}_t - \alpha\mathbf{V}_t)$, where α is the velocity and $\mathbf{V}$ is the cutting direction of the scissors. Performing edge-edge detection between E and the paper mesh M will yield most intersection points. For the endpoints of E, they may not be located at an existing edge, so we also perform vertex-face detection to get the intersection points. We insert all the intersection points as new vertices in order.

New Edge Connection. With the insertion of vertices, we perform edge connection. For a newly inserted vertex on an existing edge, we connect it to the opposite vertices in all triangles containing that edge. For a newly inserted vertex inside a triangle but not on the edge, we connect it with three vertices of the triangle.

Edge Splitting: For each newly connected edge, if its two endpoints are newly inserted vertices, then this edge need to be splitter.

C Details of Demonstration Generation

The action parameters are computed through relative pose between scissors (blade intersection P_t and cutting direction $\mathbf{V}_t$) and target line segment ($\mathbf{Tar}_t^k = (s_t^k, s_t^{k+1})$) for each step t.

When t is the step for *Push*, the pushing distance p_t is computed as :

$$p_t = \frac{(s_t^k - P_t) \cdot \mathbf{V}_t}{\|\mathbf{V}_t\|^2}\mathbf{V}_t \tag{2}$$

When t is the step for *Rotate*, the Rotation $\mathbf{R}_t$ is computed using Rodrigues' rotation formula:

$$\mathbf{w} = \mathbf{V}_t \times \mathbf{Tar}_t^k \tag{3}$$

$$\theta = \cos^{-1}(\mathbf{V}_t \cdot \mathbf{Tar}_t^k) \tag{4}$$

$$\mathbf{K} = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \tag{5}$$

$$\mathbf{R}_t = \mathbf{I} + \sin\theta\mathbf{K} + (1 - \cos\theta)\mathbf{K}^2 \tag{6}$$

where $\mathbf{I}$ is the identity matrix. When t is the step for *Close*, the closed angle c_t is computed as :

$$c_t = \mathrm{Distance2Angle}(\frac{(s_t^{k+1} - P_t) \cdot \mathbf{V}_t}{\|\mathbf{V}_t\|^2}\mathbf{V}_t) \tag{7}$$

where $\mathrm{Distance2Angle}$ is a function to map the cutting distance to the closed angle. The function is depended on the mechanical structure of scissors and we obtain it via real-world calibration.

D Details of the Benchmark

We consider the generation of curves in the uv space of the paper. In each track, curves are generated using Bézier curves parameterized by four control points, which are equidistant along the u-axis. By connecting these four points with three line segments, we can compute the gradients of these lines. We approximate the second-order derivatives of the curve by calculating the differences in gradients between these lines. For the *Easy* track, the gradient differences are always either positive or negative, ensuring a consistent curve direction. For the *Middle* track, the gradient differences include both positive and negative values, creating more complex curves. Considering the robotic arm workspace, scissors cannot undergo significant rotations, i.e., exceeding $90°$, relative to the initial orientation along the moving trajectory. Empirically, we constrain the gradient of the first line to be within $[-\tan(40°), \tan(40°)]$ and the gradient of the last line within $[-\tan(60°), \tan(60°)]$.

E Implementation of Policy Learning

The sequential point cloud features from the PointNet++ encoder are concatenated and fed into three action heads. Each action head is a shallow MLP with the shape of $[L \times 512 - 256 - 64 - a]$, where L is the length of the observation horizon and a is the dimension of the action parameter. We train the model from scratch for 120,000 iterations. The learning rate is initialized at 1×10^{-4} and decays by a factor of 0.1 at 60,000 and 110,000 iterations, respectively.

For Action Chunking, we follow the implementation of [7], which predicts k future actions using an exponential weighting scheme where $w_i = e^{-m \times i}$, with w_0 representing the weight for the most recent action. In our experiments, we set $k = 4$ and $m = 0.01$.