2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

ABPolicy: Asynchronous B-Spline Flow Policy for Real-Time and Smooth Robotic Manipulation

Fan Yang[1], Peiguang Jing[1], Kaihua Qu[1], Ningyuan Zhao[1], and Yuting Su[1,*]

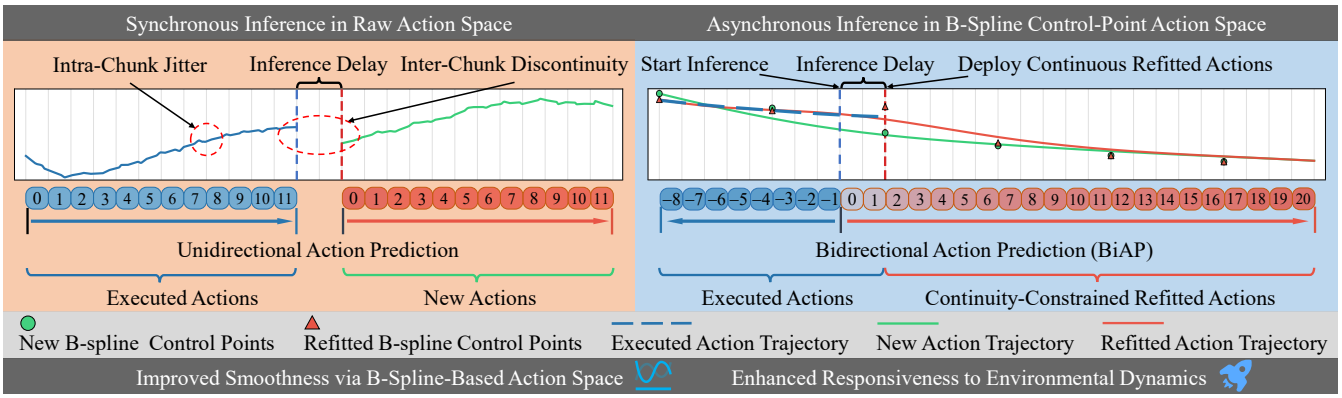

Fig. 1: Overview of ABPolicy. Our method enables real-time and smooth control via: (1) asynchronous inference to prevent execution stalls; (2) a flow-based trajectory generator predicting B-spline control points for intra-chunk smoothness; and (3) bidirectional prediction and refitting to ensure inter-chunk continuity.

***Abstract*— Robotic manipulation requires policies that are smooth and responsive to evolving observations. However, synchronous inference in the raw action space introduces several challenges, including intra-chunk jitter, inter-chunk discontinuities, and stop-and-go execution. These issues undermine a policy's smoothness and its responsiveness to environmental changes. We propose ABPolicy, an asynchronous flow-matching policy that operates in a B-spline control-point action space. First, the B-spline representation ensures intra-chunk smoothness. Second, we introduce bidirectional action prediction coupled with refitting optimization to enforce inter-chunk continuity. Finally, by leveraging asynchronous inference, ABPolicy delivers real-time, continuous updates. We evaluate ABPolicy across seven tasks encompassing both static settings and dynamic settings with moving objects. Empirical results indicate that ABPolicy reduces trajectory jerk, leading to smoother motion and improved performance. Project website: https://teee000.github.io/ABPolicy/.**

I. INTRODUCTION

Robotic manipulation in real-world environments demands control policies that are both temporally smooth and responsive to evolving observations [1-3]. In recent years, there has been growing interest in applying action chunking techniques [2] and diffusion-based models [4, 5] to robot manipulation within imitation learning frameworks. While this paradigm has achieved impressive performance, existing methods, which operate in the raw action space with synchronous inference, face several challenges: (1) intra-chunk jitter that degrades the smoothness of action trajectories; (2) inter-chunk discontinuities that introduce jerks at chunk boundaries and induce distribution shift in subsequent observations; (3) stop-and-go execution caused by synchronous inference, which undermines responsiveness to dynamic environmental changes.

These considerations bring us to the core question: how can we find an action space or representation that guarantees trajectory smoothness while being easy to integrate into asynchronous inference? Recent efforts have explored multiple avenues to mitigate this challenge [6, 7]. ACT [2] and SmolVLA [8] use temporal ensembling to suppress jitter by weighted-averaging multiple rollouts, but the averaged action is not guaranteed to be valid [9]. HumanMAC [10] and FAST [11] parameterize actions using discrete cosine transform (DCT) [12] coefficients, which improves intra-chunk smoothness but does not resolve discontinuities at chunk boundaries. PTP [13] predicts past-action tokens and samples multiple trajectories from the same observation at test time, then selects the one most consistent with prior actions.

*Corresponding author.
[1]The authors are with the School of Electrical and Information Engineering, Tianjin University (e-mail: eeyf@tju.edu.cn; pgjing@tju.edu.cn; 1106153048@qq.com; nyzhao@tju.edu.cn; ytsu@tju.edu.cn).
*This work was supported by the Guangxi Natural Science Foundation Key Project under Grant 2025JJD170006, and the National Natural Science Foundation of China under Grant 62361002.

This improves temporal coherence but demands repeated sampling during inference, hurting real-time responsiveness. BEAST [14] discretizes B-spline [15] control points to represent actions, improving smoothness but sacrificing fitting accuracy due to discretization. In contrast, we use continuous B-spline control points to achieve lower fitting error. We then employ a flow-matching model [16, 17] to predict these control points, yielding more accurate action predictions.

Real-world environments are dynamic, requiring agents to act continuously while perceiving evolving observations [2, 18-20]. Conventional synchronous inference induces action stalls that degrade responsiveness [3, 4, 21]. By deploying asynchronous model inference on cloud GPUs, SmolVLA [8] reduces the computational burden on the local device, but at the cost of introducing additional latency from data transmission. Asynchronous methods like RTC [9] employ an "inpainting" strategy to enforce continuity, which utilizes gradient-based guidance and is further enhanced by a weighted fusion of action chunks. Yet, this combined mechanism perturbs the learned dynamics and introduces sensitive hyperparameters. Our approach is orthogonal, we ensure the continuity of asynchronously predicted trajectories via a simple optimization of the B-spline control points.

To achieve both real-time and smooth robotic manipulation, we propose ABPolicy, an asynchronous B-spline flow policy. ABPolicy operates in a B-spline control-point action space, generating continuous trajectories for action chunks. This representation inherently guarantees intra-chunk smoothness and eliminates jitter. To enforce continuity between chunks, we introduce two key mechanisms: bidirectional action prediction, which jointly models a short window of past and future actions, and a continuity-constrained refitting optimization. This optimization locally adjusts the initial control points of a new trajectory to align with the executed actions. Finally, ABPolicy performs inference asynchronously; the manipulator continues to act while the policy updates in the background, enabling real-time adjustments that enhance responsiveness to environmental dynamics. To validate our approach, we demonstrate its effectiveness on a suite of benchmarks, including three challenging dynamic manipulation tasks and four static ones.

Our main contributions are summarized as follows:

- We propose ABPolicy, a flow matching policy that generates action trajectories in a B-spline control point space for inherent smoothness.
- We introduce a simple yet effective continuity optimization mechanism, combining bidirectional prediction and continuity-constrained refitting, to seamlessly stitch together asynchronously generated trajectories.
- Evaluations on seven manipulation tasks, including three dynamic tasks, confirm that ABPolicy delivers smoother and more reactive control compared to prior methods.

II. RELATED WORKS

A. Action Representation

Action representation serves as the critical bridge between learned models and robotic motion [22-25]. Researchers have proposed a spectrum of action parameterization strategies. One prominent class employs high-level semantic representations. For example, specifying sub-tasks in natural language [26] or encoding behavior via discrete action keypoints [27]. This family of methods integrates naturally with large-scale foundation models [28, 29]. However, these approaches typically rely on dedicated low-level motion controllers for execution, which limits their generality and adaptability across tasks and platforms.

Another paradigm directly predicts low-level control signals from observations and human instructions [1, 30, 31]. A common instantiation converts continuous actions into discrete tokens, enabling generation with autoregressive sequence models [32, 33]. For example, straightforward per-dimension binning can be used to discretize actions [34]. However, such discretization introduces quantization error and information loss, which is detrimental to fine-grained manipulation and precise control.

A complementary line of research eschews discretization by predicting continuous low-level control signals from observations and instructions [2, 35-39]. These methods typically employ U-Net [40] or transformer [41, 42] architectures to model rich, multimodal action distributions within diffusion- or flow-based generative frameworks. By preserving the fidelity of high-frequency motor commands and avoiding quantization artifacts, they deliver superior performance in fine-grained manipulation. Orthogonal to these approaches, our method represents raw action trajectories with a B-spline parameterization and trains a flow-matching model to generate the continuous B-spline control points.

B. Real-Time Execution

Within the imitation learning framework, most methods use synchronous inference [4, 21]: the robot waits for the model to finish inference before executing the next action chunk. This design introduces latency that weakens responsiveness in dynamic environments [43]. As model size increases, computational cost rises, making the delay more pronounced. To mitigate this, prior work follows two main paths: (1) accelerating diffusion-based or flow-based policies with DDIM sampling [44], model distillation, or training one-step or few-step denoisers [45, 46] to accelerate the denoising process; (2) adopting asynchronous inference, which runs policy computation in parallel with control signal dispatch to avoid execution pauses [8, 9, 47]. We adopt an asynchronous inference scheme, running model inference and robot control dispatch in separate threads. This design eliminates idle time and maintains real-time responsiveness to environmental changes.

III. METHOD

Our design targets two goals: smooth action trajectories and asynchronous real-time inference. We learn a continuous

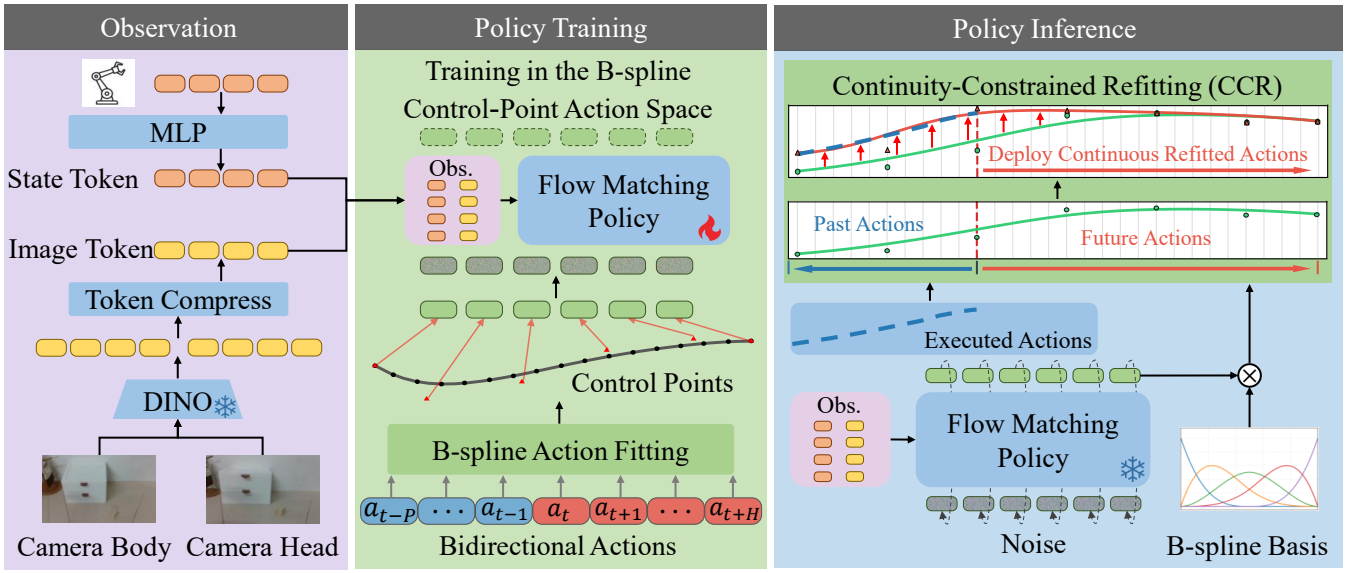

Fig. 2: Overview of ABPolicy. A policy trained for Bidirectional Action Prediction (BiAP) asynchronously generates future B-spline control points at inference time. These are then optimized by our Continuity-Constrained Refitting (CCR) module to guarantee smooth continuity with the executed trajectory.

distribution over B-spline control points using a flow matching model. During inference, we asynchronously update a subset of these points to ensure the forthcoming trajectory remains continuous with recently executed actions.

A. B-Spline Trajectory Parameterization

We employ B-splines [15] to parameterize action trajectories, which ensures smoothness and provides a compact representation. Specifically, we use cubic B-splines (degree $p = 3$). This choice guarantees C^2 continuity, meaning both the velocity (first derivative) and acceleration (second derivative) of the action trajectory are continuous. Such smoothness is crucial for generating physically realistic motions and avoiding the abrupt changes that can arise from predicting raw actions. In practice, this process is applied independently to each dimension of the action vector.

Given a discrete action sequence $\{a_t\}_{t=0}^{T-1}$, where t denotes the time step, the objective is to determine a set of N control points $\{c_i\}_{i=0}^{N-1}$ that optimally approximates the sequence. A B-spline curve $s(t)$ of degree p is defined as follows:

$$s(t) = \sum_{i=0}^{N-1} c_i N_{i,p}(t), \tag{1}$$

where $N_{i,p}(t)$ is the i-th B-spline basis function of degree p, defined over a knot vector. The optimal control points $\{c_i^*\}$ are found by minimizing the sum of squared errors between the original actions a_t and the spline curve $s(t)$ evaluated at the corresponding discrete time steps:

$$\{c_i^*\}_{i=0}^{N-1} = \arg\min_{\{c_i\}} \sum_{t=0}^{T-1} \left(a_t - s(t)\right)^2. \tag{2}$$

This formulation results in a linear least-squares problem, which can be solved efficiently to yield the optimal control points $\{c_i^*\}$.

Once the optimal control points $\{c_i^*\}$ have been obtained from the fitting process, the continuous and smooth action trajectory $\hat{a}(t)$ can be fully reconstructed using the B-spline definition.

B. Bidirectional Action Prediction via Flow Matching

To generate the action trajectories, we introduce a policy network formulated as a conditional model $\pi_\theta(C_t^*|o_t)$. Here, θ represents the network's trainable parameters, o_t is the observation at time t, and $C_t^* = [c_0^*, c_1^*, ..., c_{N-1}^*]$ is the vector of ground-truth control points that parameterize the action chunk. This approach fundamentally shifts the learning objective: instead of predicting raw actions, our policy learns to generate a compact set of control points, inherently ensuring the smoothness of the resulting action trajectory.

To enhance the continuity between past and future actions and explicitly models the temporal structure of actions, we adopt a bidirectional action prediction scheme (BiAP). At each time step t, the policy's target is not a single action a_t, but a full action chunk A_t that spans P previous steps and H future steps:

$$A_t = [a_{t-P}, \ldots, a_{t-1}, a_t, a_{t+1}, \ldots, a_{t+H-1}]. \tag{3}$$

Our approach bypasses the direct regression of raw action chunks A_t by instead working with their B-spline representation. The optimal control points C_t^* corresponding to an action chunk A_t are determined through the least-squares fitting procedure (Eq. (2)). These control points subsequently form the regression target for our policy network.

We formulate the prediction of B-spline control points as a conditional generative modeling problem, which we address using flow matching [16, 17]. Rather than regressing the control points directly, the policy network is trained to predict

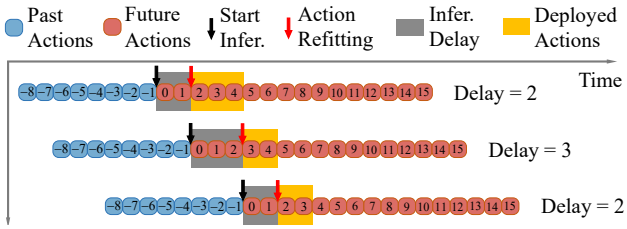

Fig. 3: Asynchronous inference overview. During inference delay, the robot executes the prior cycle's actions. Gray shaded regions denote the inference-delay window, whereas orange shaded regions indicate the actions being executed.

a conditional vector field that governs a dynamic transformation. This transformation maps samples from a simple prior (e.g., a standard Gaussian) to the target distribution of valid control points, conditioned on the observation. A key advantage of this generative formulation is its inherent ability to model multi-modal trajectory distributions [4, 5]. The network parameters θ are optimized via the loss function:

$$\mathcal{L}^{\tau}_{\text{FM}}(\theta) = \mathbb{E}_{\tau, o_t, z, C^*_t}\left[\left\|\pi_\theta(C^\tau_t|o_t) - (C^*_t - z)\right\|^2\right], \quad (4)$$

where C^*_t denotes the ground-truth control points for observation o_t, $z \sim \mathcal{N}(0, \mathbf{I})$ is sampled from the prior, $\tau \sim U[0,1]$, and $C^\tau_t = (1-\tau)z + \tau C^*_t$ is a point on the linear path between the noise and data. Minimizing this objective compels the network's output to align with the target vector $(C^*_t - z)$, thus learning the vector field that transforms noise into the desired data, conditioned on o_t.

C. Continuity-Constrained Refitting

Asynchronous inference design enhances the system's real-time responsiveness to dynamic changes [8, 9, 47]. However, this architecture introduces an unavoidable latency between when an observation is captured and when the corresponding new action trajectory is available. As shown in Fig. 1 and Fig. 3, during the model inference period, the robot continues to execute actions from the previously computed trajectory. As a result, a direct application of the newly predicted trajectory would cause action discontinuity, as it fails to account for the actions executed during the delay.

To address this challenge, we introduce the Continuity-Constrained Refitting (CCR) mechanism, illustrated in Fig. 2. The core idea is to locally adjust the initial segment of the newly predicted trajectory to enforce continuity with the sequence of executed actions.

Let $\{c_{\text{pred},i}\}_{i=0}^{N-1}$ be the set of control points predicted by the policy network. Let $\{a_t^{\text{exec}}\}_{t=0}^{P-1}$ be the sequence of P actions that were executed. Our goal is to derive a new set of control points, $\{c_{\text{new},i}\}_{i=0}^{N-1}$, that forms a smooth continuation from this history.

To this end, we exploit the local support property of B-splines. The refitting procedure is constrained to adjust only the initial N_{free} control points, while the subsequent points from the policy's prediction, $\{c_{\text{pred},i}\}_{i=N_{\text{free}}}^{N-1}$, remain unaltered. The optimal values for this initial "free" subset are determined by solving the following least-squares problem:

$$\{c_{\text{new},i}\}_{i=0}^{N_{\text{free}}-1} = \underset{\{c_i\}_{i=0}^{N_{\text{free}}-1}}{\arg\min} \sum_{t=0}^{P-1} \left(a_t^{\text{exec}} - \hat{s}_{\text{new}}(u_t)\right)^2, \quad (5)$$

where u_t maps the step t to the B-spline domain, and $\hat{s}_{\text{new}}(u)$ is the partially updated trajectory defined as:

$$\hat{s}_{\text{new}}(u) = \underbrace{\sum_{i=0}^{N_{\text{free}}-1} c_i N_{i,p}(u)}_{\text{Free points (to be optimized)}} + \underbrace{\sum_{i=N_{\text{free}}}^{N-1} c_{\text{pred},i} N_{i,p}(u)}_{\text{Fixed points (from policy)}}. \quad (6)$$

This optimization minimizes the error between the start of the new trajectory and the executed action history. By solving for the initial control points that best fit this history, CCR effectively "anchors" the new trajectory to the immediate past, guaranteeing continuity and smoothness.

D. Asynchronous Inference

To achieve real-time reactivity in dynamic environments, we introduce an asynchronous inference framework that decouples model inference from action execution. As depicted in Fig. 3, this architecture runs inference and control in two parallel threads. While the model computes the next action sequence, the robot executes the trajectory from the prior cycle, effectively hiding inference latency.

Once a new sequence is inferred, we instantly update the action queue by applying our refitting method. This guarantees a smooth transition from the previous action plan, prevents jitter, and immediately triggers the next inference cycle. This asynchronous loop enables the robot to remain responsive to environmental changes without being stalled by computation delays.

IV. EXPERIMENTS

A. Tasks

As illustrated in Fig. 4, we evaluate seven manipulation tasks comprising dynamic and static settings. The dynamic tasks are: stacking a block on a rotating platform, pushing a block on a rotating platform, and hanging a cup onto a rotating rack. The rotary fixture maintains a constant angular velocity of one revolution every 10 s. The static tasks are: folding a towel, stacking a block on a stationary base, hanging a cup onto a fixed hook, and putting a block into a drawer. For dynamic tasks, we collect 100 demonstrations per task. For static tasks, we collect 200 demonstrations per task with the manipulated object randomly initialized within a $20cm \times 20cm$ workspace region. Success rates for each task are calculated based on 20 evaluation runs.

B. Implementation Details

We employ a 6-DoF AgileX Piper manipulator equipped with a parallel gripper and two fixed third-person Intel RealSense D435 RGB cameras. The action space comprises arm joint angles and a continuous gripper aperture command, executed at a 30 Hz control rate. We fit an independent cubic B-spline (degree $p = 3$) for each action dimension,

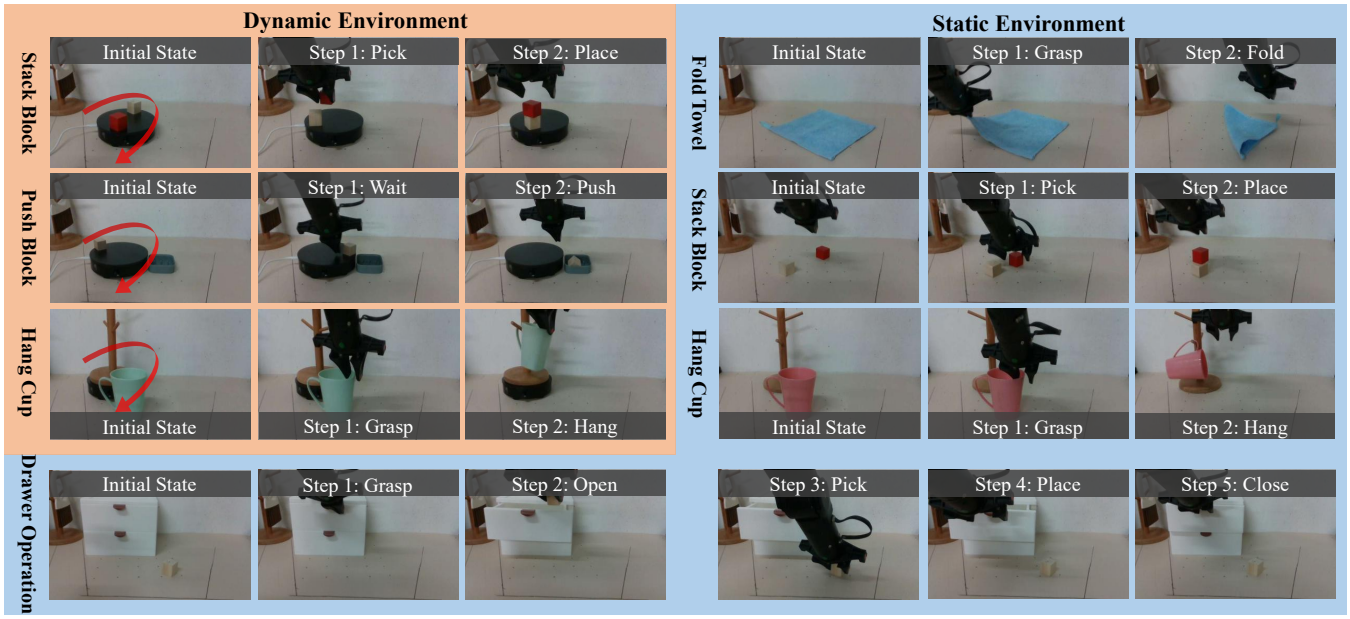

Fig. 4: Manipulation tasks in static and dynamic settings, where dynamic tasks involve an object on a platform rotating at a constant velocity (approximately 10 seconds per revolution).

TABLE I: Comparison of synchronous and asynchronous inference performance across dynamic and static tasks.

	Dynamic Tasks			Static Tasks							
	Stack Block	Push Block	Hang Cup	Stack Block		Fold Towel		Hang Cup		Drawer Operation	
	Success ↑	Success ↑	Success ↑	Success ↑	Time ↓	Success ↑	Time ↓	Success ↑	Time ↓	Success ↑	Time ↓
DP [4]	40	75	35	25	10.2	25	12.2	85	10.8	**100**	18.9
Sync	30	75	40	**85**	8.7	55	10.2	**90**	10.1	**100**	17.5
Async	**55**	**85**	**60**	80	**7.5**	**60**	**8.4**	85	**8.2**	**100**	**15.8**

employing an open-uniform (clamped) knot vector. For BiAP, we set $H = 32$ (future) and $P = 8$ (history). Inference is performed on an NVIDIA RTX 4070 Ti Super GPU with 10 denoising steps for the flow-matching model. The policy network employs a DiT architecture [42], which fuses observations via cross-attention [3]. The observation encoder has two streams: a frozen, pre-trained DINO-V2 model [48] processes the current image frame, while an MLP encodes the robot state from the past 8 frames. More information can be found on our project website.

C. Comparison of Synchronous and Asynchronous Inference

To assess how inference mode affects robotic manipulation, we compared synchronous (sync) and asynchronous (async) approaches across seven tasks, categorized as dynamic or static (Tab. I).

For dynamic tasks, the advantage of async inference is more pronounced due to its ability to respond to environmental changes. Success rates improved by an average of 18.3% across the three dynamic tasks. Under synchronous inference, the robot idles during inference, limiting reactivity to changes and increasing failure rates. In contrast, asynchronous inference yields lower action latency, enabling timely and effective interaction with moving objects.

For static tasks, the principal benefit of async inference is improved efficiency. It reduces completion time by 14.2% on average, with an inference delay of roughly 90 ms. This efficiency gain is beneficial for real-world deployment, where throughput is a key metric.

D. Analysis of Action Representation Accuracy

The fidelity of action representation is a critical factor that profoundly influences the performance of robotic manipulation tasks; inaccurate or coarse representations can introduce fitting errors that manifest as suboptimal execution. To quantify this effect, we compare four prevalent techniques by evaluating reconstruction accuracy on 40-timestep action chunks, averaged over 200 random samples.

- **Discrete Bins (256):** Quantizes the action space into 256 bins [1, 30, 34, 49].
- **DCT Coefficients (8):** Uses the 8 lowest-frequency DCT coefficients to encode the trajectory [10].
- **B-spline (Discrete):** Represents each action chunk with 8 B-spline control points quantized into 256 bins [14].
- **B-spline (Continuous):** Represents each action chunk with 8 continuous B-spline control points.

We assessed performance using two metrics. Mean Error is the average discrepancy between the original and recon-

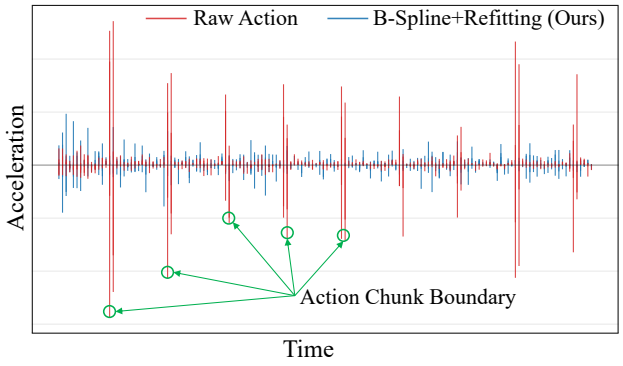

Fig. 5: Acceleration comparison between raw actions and our proposed method. The raw action representation produces large accelerations at the boundaries of action chunks, which leads to jitter. In contrast, our B-spline representation with refitting reduces boundary spikes and improves smoothness.

structed trajectories. SNR measures fidelity by comparing the power of the original signal to that of the fitting error.

TABLE II: Comparison of reconstruction accuracy for different action representation methods.

Action Representation Method	Mean Error ↓	SNR (dB) ↑
256 Discrete Bins [34]	0.0020	41.9
DCT Low-Frequency Coeffs [10]	0.0010	44.6
B-spline (Discrete) [14]	0.0025	37.7
B-spline (Continuous) (Ours)	**0.00031**	**50.7**

As shown in Tab. II, the continuous B-spline representation attains a mean error of 0.00031 and an SNR of 50.7 dB, outperforming other methods. This indicates the most faithful, low-noise reconstruction of the original action trajectory, enabling smoother and more effective manipulation.

Notably, the DCT-based approach provides a reasonable approximation but remains less accurate than the continuous B-spline. Standard 256-bin quantization performs worse, exposing the limits of direct discretization. Quantizing B-spline control points does not outperform alternatives in reconstruction error. This underscores the need to maintain continuity in the control-point space; discretization there markedly degrades precision.

E. Analysis of Action Smoothness

To evaluate the performance of the B-spline representation in reducing jitter, we conducted comparative experiments against raw actions. We used two key metrics:

- **Zero-Crossing Rate (ZCR) of velocity:** the frequency with which a joint's velocity changes sign. Lower ZCR indicates fewer oscillations and smoother motion.
- **95th percentile of acceleration (Acc p95):** captures high-frequency changes in motion; lower values indicate less aggressive acceleration and smoother control.

As shown in Tab. III, the B-spline method reduces the average velocity ZCR by 29.2% compared to raw actions. Furthermore, it achieves a 57.1% reduction in Acc p95

TABLE III: Comparison of action representations with trajectory jitter.

Joint	Average ZCR of Velocity ↓		Acc p95 ↓	
	B-spline	Raw	B-spline	Raw
j0	0.1535	0.2748	2.34	8.73
j1	0.1063	0.1615	3.63	8.04
j2	0.1321	0.1334	3.06	7.44
j3	0.1422	0.1356	3.09	6.29
j4	0.1387	0.2201	2.35	4.23
j5	0.1075	0.1769	3.88	8.03
Average	**0.1301**	**0.1837**	**3.06**	**7.13**

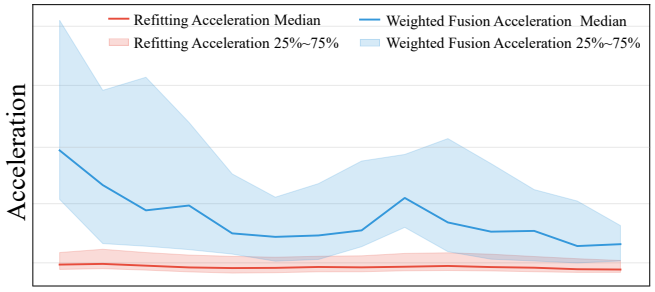

Fig. 6: Comparison of action-chunk boundary smoothing methods: our B-spline refitting versus the weighted fusion. The y-axis shows acceleration magnitude, and the x-axis represents the smoothing time window.

relative to raw actions. These results confirm that our method produces smoother actions with lower velocity and acceleration variations and is highly effective at reducing jitter.

To validate our continuity-constrained refitting method for reducing discontinuities at action chunk boundaries, we benchmarked it against two baselines: raw actions and the weighted fusion technique used in SmolVLA [8] and RTC [9]. As illustrated in Fig. 6 and Fig. 5, our refitting method yields a markedly smoother action profile. We model the action trajectory using a cubic B-spline of degree $p = 3$ [15]. This approach provides a mathematical guarantee of C^2 continuity, ensuring the resulting velocity and acceleration profiles are inherently smooth.

F. Analysis of Bidirectional Action Prediction

TABLE IV: Ablation study on the effectiveness of BiAP, comparing success rate against the discontinuity metric at action boundaries, measured before and after refitting.

	Success Rate (%) ↑	Initial Jitter ↓	Refitted Jitter ↓
w/o BiAP	60	0.0220	0.0180
w/ BiAP	**85**	**0.0170**	**0.0097**

Our ablation study on the static block stacking task (Tab. IV) confirms the efficacy of BiAP. Its inclusion boosts the success rate from 60% to 85%, a performance gain stemming from enhanced motion continuity. To quantify this, we measure the transitional jitter between action segments.

BiAP reduces the initial jitter by nearly 23% and enables our refitting module to cut the final jitter by 46% relative to the baseline (0.0097 vs. 0.018). These findings demonstrate that BiAP is crucial for generating smooth, low-jitter trajectories.

V. CONCLUSION

This paper introduced ABPolicy, an asynchronous B-spline flow policy that resolves critical smoothness and latency issues in robotic manipulation. By combining a B-spline representation with bidirectional prediction and refitting, our method ensures smooth, continuous trajectories, while its asynchronous design enables the real-time responsiveness crucial for dynamic tasks. Experimental results confirm ABPolicy reduces jerk and improves manipulation performance, offering a powerful framework for developing more agile and capable robots for real-world environments.

REFERENCES

[1] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[2] 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.

[3] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, "Rdt-1b: a diffusion foundation model for bimanual manipulation," *arXiv preprint arXiv:2410.07864*, 2024.

[4] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, 2023.

[5] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "π0: A vision-language-action flow model for general robot control. corr, abs/2410.24164, 2024. doi: 10.48550," *arXiv preprint ARXIV.2410.24164*, 2024.

[6] J. Studer, D. Agrawal, D. Borer, S. Sadat, R. W. Sumner, M. Guay, and J. Buhmann, "Factorized motion diffusion for precise and character-agnostic motion inbetweening," in *Proceedings of the 17th ACM SIGGRAPH conference on motion, interaction, and games*, 2024, pp. 1–10.

[7] J.-G. Habekost, C. Gäde, P. Allgeuer, and S. Wermter, "Inverse kinematics for neuro-robotic grasping with humanoid embodied agents," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 7315–7322.

[8] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti *et al.*, "Smolvla: A vision-language-action model for affordable and efficient robotics," *arXiv preprint arXiv:2506.01844*, 2025.

[9] K. Black, M. Y. Galliker, and S. Levine, "Real-time execution of action chunking flow policies," *arXiv preprint arXiv:2506.07339*, 2025.

[10] L.-H. Chen, J. Zhang, Y. Li, Y. Pang, X. Xia, and T. Liu, "Humanmac: Masked motion completion for human motion prediction," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 9544–9555.

[11] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine, "Fast: Efficient action tokenization for vision-language-action models," *arXiv preprint arXiv:2501.09747*, 2025.

[12] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. 100, no. 1, pp. 90–93, 2006.

[13] M. Torne, A. Tang, Y. Liu, and C. Finn, "Learning long-context diffusion policies via past-token prediction," *arXiv preprint arXiv:2505.09561*, 2025.

[14] H. Zhou, W. Liao, X. Huang, Y. Tang, F. Otto, X. Jia, X. Jiang, S. Hilber, G. Li, Q. Wang *et al.*, "Beast: Efficient tokenization of b-splines encoded action sequences for imitation learning," *arXiv preprint arXiv:2506.06072*, 2025.

[15] W. J. Gordon and R. F. Riesenfeld, "B-spline curves and surfaces," in *Computer Aided Geometric Design*. Elsevier, 1974, pp. 95–126.

[16] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[17] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," *arXiv preprint arXiv:2209.03003*, 2022.

[18] M. J. Kim, C. Finn, and P. Liang, "Fine-tuning vision-language-action models: Optimizing speed and success," *arXiv preprint arXiv:2502.19645*, 2025.

[19] J. Wen, Y. Zhu, J. Li, M. Zhu, Z. Tang, K. Wu, Z. Xu, N. Liu, R. Cheng, C. Shen *et al.*, "Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation," *IEEE Robotics and Automation Letters*, 2025.

[20] Y. Yang, Y. Wang, Z. Wen, L. Zhongwei, C. Zou, Z. Zhang, C. Wen, and L. Zhang, "Efficientvla: Training-free acceleration and compression for vision-language-action models," *arXiv preprint arXiv:2506.10100*, 2025.

[21] S. Xia, H. Fang, C. Lu, and H.-S. Fang, "Cage: Causal attention enables data-efficient generalizable robotic manipulation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 13 242–13 249.

[22] X. Li, P. Li, M. Liu, D. Wang, J. Liu, B. Kang, X. Ma, T. Kong, H. Zhang, and H. Liu, "Towards generalist robot policies: What matters in building vision-language-action models," *arXiv preprint arXiv:2412.14058*, 2024.

[23] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh, "Rt-h: Action hierarchies using language," *arXiv preprint arXiv:2403.01823*, 2024.

[24] J. Liu, H. Chen, P. An, Z. Liu, R. Zhang, C. Gu, X. Li, Z. Guo, S. Chen, M. Liu *et al.*, "Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model," *arXiv preprint arXiv:2503.10631*, 2025.

[25] J. O. von Hartz, T. Welschehold, A. Valada, and J. Boedecker, "The art of imitation: Learning long-horizon manipulation tasks from few demonstrations," *IEEE Robotics and Automation Letters*, 2024.

[26] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[27] N. Di Palo and E. Johns, "Keypoint action tokens enable in-context imitation learning in robotics," in *Robotics: Science and Systems (RSS)*, 2024.

[28] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge *et al.*, "Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution," *arXiv preprint arXiv:2409.12191*, 2024.

[29] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello *et al.*, "Paligemma: A versatile 3b vlm for transfer," *arXiv preprint arXiv:2407.07726*, 2024.

[30] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Conference on Robot Learning*. PMLR, 2023, pp. 2165–2183.

[31] S. Lee, Y. Wang, H. Etukuru, H. J. Kim, N. M. M. Shafiullah, and L. Pinto, "Behavior generation with latent actions," *arXiv preprint arXiv:2403.03181*, 2024.

[32] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo, "Embodiedgpt: Vision-language pre-training via embodied chain of thought," *Advances in Neural Information Processing Systems*, vol. 36, pp. 25 081–25 094, 2023.

[33] X. Wang, X. Zhang, Z. Luo, Q. Sun, Y. Cui, J. Wang, F. Zhang, Y. Wang, Z. Li, Q. Yu *et al.*, "Emu3: Next-token prediction is all you need," *arXiv preprint arXiv:2409.18869*, 2024.

[34] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[35] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu *et al.*, "Octo: An open-source generalist robot policy," *arXiv preprint arXiv:2405.12213*, 2024.

[36] J. Wen, Y. Zhu, J. Li, Z. Tang, C. Shen, and F. Feng, "Dexvla: Vision-language model with plug-in diffusion expert for general robot control," *arXiv preprint arXiv:2502.05855*, 2025.

[37] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3d diffuser actor: Policy diffusion with 3d scene representations," *arXiv preprint arXiv:2402.10885*, 2024.
[38] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.
[39] G. Yan, J. Zhu, Y. Deng, S. Yang, R.-Z. Qiu, X. Cheng, M. Memmel, R. Krishna, A. Goyal, X. Wang *et al.*, "Maniflow: A general robot manipulation policy via consistency flow training," *arXiv preprint arXiv:2509.01819*, 2025.
[40] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[42] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4195–4205.
[43] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.
[44] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.
[45] Z. Geng, M. Deng, X. Bai, J. Z. Kolter, and K. He, "Mean flows for one-step generative modeling," *arXiv preprint arXiv:2505.13447*, 2025.
[46] K. Frans, D. Hafner, S. Levine, and P. Abbeel, "One step diffusion via shortcut models," *arXiv preprint arXiv:2410.12557*, 2024.
[47] G. Gao, J. Wang, J. Zuo, J. Jiang, J. Zhang, X. Zeng, Y. Zhu, L. Ma, K. Chen, M. Sheng *et al.*, "Towards human-level intelligence via human-like whole-body manipulation," *arXiv preprint arXiv:2507.17141*, 2025.
[48] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.
[49] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan, "3d-vla: A 3d vision-language-action generative world model," *arXiv preprint arXiv:2403.09631*, 2024.