DexFLEX: Contact-Aware Foundation Controller for Command-Guided Dexterity

Anonymous Author(s)
Affiliation
Address
email

Abstract: Dexterous robot hands can receive useful motion intent from teleoperation or learned policies, but successful execution also depends on contact decisions that these commands rarely specify. We introduce DexFLEX, a contact-aware foundation controller that turns upstream fingertip-motion drafts into contact-consistent joint commands. Instead of treating a draft as a trajectory to copy, DexFLEX treats it as evidence about intent: from the current tactile–proprioceptive state, it proposes feasible short-horizon motion chunks, predicts their contact consequences, and selects the candidate that best follows the command while preserving future contact stability. At inference time, a simple *Draft–Dream–Select* procedure combines pure-prior proposals for recovery with draft-seeded proposals for responsiveness, then decodes the selected motion into executable joint-space targets. Across degraded-command simulation, real-world shared control, and visuomotor policy learning, the same trained controller improves robustness without retraining, increasing real-world teleoperation success from 29.2% to 78.3% and reaching 46.7% policy-learning success, 3.5× direct joint-action prediction, and 1.56× prior-only correction. Videos are available at dex-flex.github.io.

Keywords: Dexterous Manipulation, Shared Autonomy, Sim-to-Real

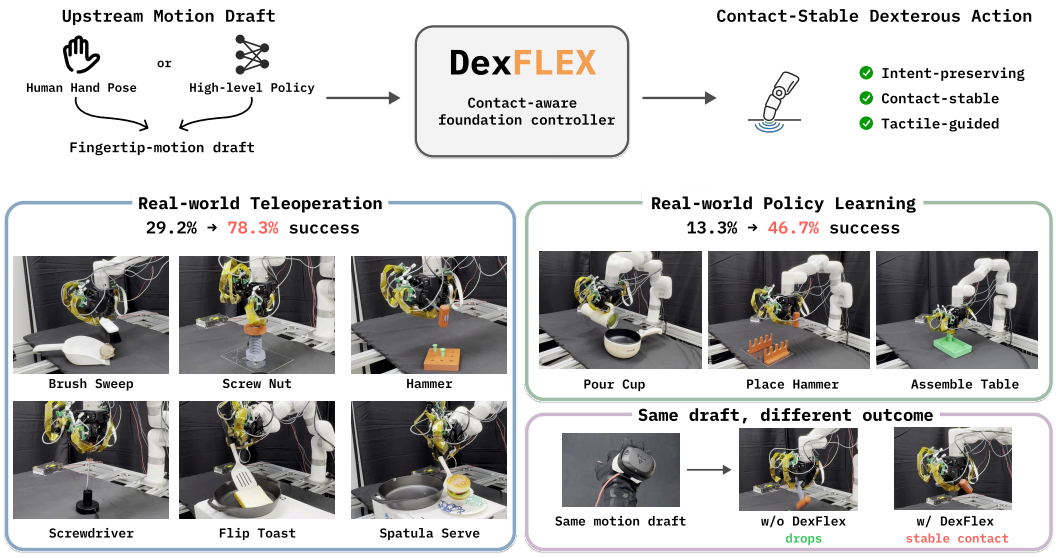

Figure 1: **DexFLEX converts upstream motion drafts into contact-stable dexterous actions.** Human operators or learned policies provide fingertip-motion drafts, and DexFLEX supplies tactile-guided low-level correction for contact-consistent execution. With the same draft, DexFLEX preserves contact where direct execution can fail.

Submitted to the 10th Conference on Robot Learning (CoRL 2026). Do not distribute.

1 Introduction

Dexterous robot hands can receive useful motion intent from teleoperation, retargeting, or learned policies, but successful execution also depends on contacts that are rarely specified by those commands. A short fingertip-motion draft may indicate where the fingertips should move next, yet omit which contacts should remain load-bearing, which fingers should unload, or when support should transfer during regrasping and tool use. As a result, a kinematically reasonable motion can still fail by releasing a support finger, breaking contact too early, or disturbing a stable grasp. We study this problem as *command-guided dexterity*: a low-level controller must preserve useful upstream intent while making execution contact-consistent.

Existing methods address only parts of this problem. Shared-autonomy and generative dexterous-control methods can refine user or policy commands with learned motion priors [1–4], but prior-only correction mainly evaluates motion plausibility or draft agreement, not the contact outcome a motion will produce. Two fingertip motions can be similarly close to the draft, while only one preserves a supporting contact or stable tool grasp. Tactile and visuotactile policies show that touch is crucial for contact-rich dexterity [5–7], yet they are typically trained end-to-end for specific tasks rather than as reusable low-level controllers beneath different command sources. This motivates a contact-aware foundation controller: a reusable dexterous executor guided by motion commands, grounded in tactile state, and compatible with multiple upstream interfaces.

We introduce DEXFLEX, a *Contact-Aware Foundation Controller for Command-Guided Dexterity*. The key idea is to treat the upstream draft not as a trajectory to copy, but as online evidence about intent. DEXFLEX follows a propose–predict–select principle: a state-conditioned consistency-flow prior proposes short-horizon fingertip-motion chunks from tactile–proprioceptive state, a latent world model predicts each candidate's contact consequence, and a learned score selects the candidate that best preserves the draft while avoiding contact failures. At inference, DEXFLEX implements this through *Draft–Dream–Select*, mixing pure-prior proposals for recovery with draft-seeded proposals for responsiveness before decoding the selected chunk into joint-space targets. This keeps the learned motion prior reusable while allowing the controller to repair contact-critical details without simply damping the command or overriding upstream intent.

We evaluate DEXFLEX in degraded-command simulation, real-world shared control, and visuomotor policy learning. In simulation, we corrupt expert commands with noise, scaling, latency, and support-finger removal to test intent recovery under contact constraints. In the real world, operators complete six long-horizon tool-use tasks, and an image-based flow matching policy uses DEXFLEX as a frozen low-level action interface. DEXFLEX improves average real-world success from 29.2% with raw teleoperation to 78.3%, outperforming DexGen's 52.5% by 25.8 percentage points. As a low-level policy interface, it reaches 46.7% average success, **3.5**$\times$ the success rate of direct joint-action prediction and **1.56**$\times$ that of prior-only correction.

Contributions. Our contributions are: (1) we formulate command-guided dexterity as contact-aware low-level refinement of fingertip-motion drafts; (2) we introduce DEXFLEX, a reusable controller that combines state-conditioned motion proposals, latent consequence prediction, and Draft–Dream–Select execution; and (3) we show that the same trained module improves degraded-command recovery, real-world shared control, and visuomotor policy learning without retraining.

2 Related Work

Foundation controllers and motion priors for dexterous manipulation. Recent progress in robot foundation models has explored large-scale pretraining as a way to obtain generalist policies across tasks and environments [8–10]. Most target end-to-end visuomotor control for broad manipulation, whereas contact-rich dexterous hands require low-level execution that responds to fine-grained motion and contact changes [11–17]. Teleoperation and retargeting provide natural upstream interfaces for such fine-grained dexterous guidance [18–21]. Most closely related, DexGen learns a teleoperation-prompted dexterous foundation controller for generating robot-hand actions [4]. DEXFLEX shares the goal of reusable dexterous control, but separates the state-conditioned motion

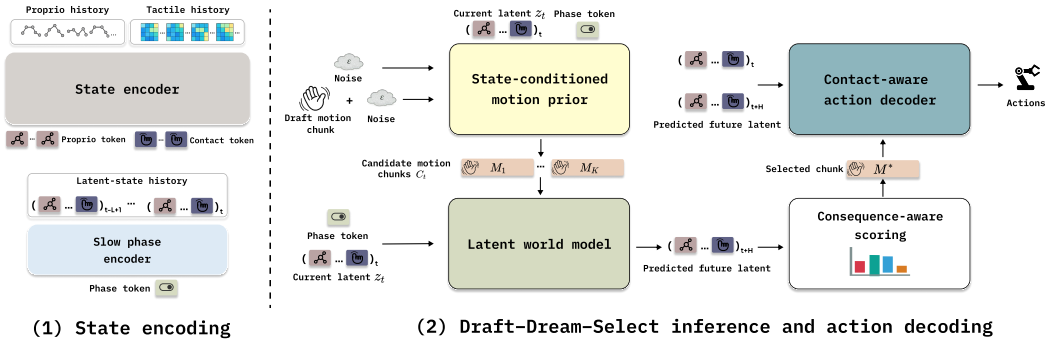

Figure 2: **DEXFLEX pipeline. (1)** A state encoder converts proprioceptive and tactile histories into a contact–proprioceptive latent z_t, whose temporal history is summarized as a slow phase token g_t. **(2)** At inference, Draft–Dream–Select proposes pure-prior and draft-seeded motion chunks, predicts their contact consequences, scores them against the upstream draft, and decodes the selected chunk into joint-space actions.

prior from command-dependent correction: the prior proposes feasible fingertip-motion chunks independent of the command source, while the draft is used to select among candidates. By selecting motions using predicted contact consequences rather than draft agreement alone, DEXFLEX serves as a contact-aware low-level executor beneath teleoperation, retargeting, and learned policies.

Shared autonomy and tactile contact correction. Shared autonomy improves imperfect user commands through intent inference, policy blending, residual correction, safety filtering, or learned command refinement [1–3, 22–27]. Dexterous manipulation adds a contact-level ambiguity: two motions can match the user's intent, while only one preserves supporting contacts or avoids premature release. Tactile and visuotactile methods show that touch is critical for resolving such ambiguities in in-hand rotation, contact-rich control, and dexterous policy learning [5–7, 28–30], but are usually trained as task-specific policies or representation learners. DEXFLEX combines shared-autonomy-style intent preservation with tactile contact reasoning by treating the upstream draft as intent evidence and selecting candidate fingertip motions using predicted contact consequences.

3 Method

Problem setup. At each control step t, the robot receives recent proprioception o_t and fingertip tactile observations $\mathcal{T}_t$. An upstream interface—such as glove teleoperation, retargeting, or a learned high-level policy—provides a short fingertip-space motion draft $\tilde{M}_t = \{\Delta\tilde{x}_{t+i}\}_{i=1}^{H}$, where x_t denotes the current fingertip keypoints and H is a short horizon. The role of the low-level controller is to convert this draft into a contact-consistent motion chunk $M_t = \{\Delta x_{t+i}\}_{i=1}^{H}$ that preserves the intended fingertip motion while avoiding contact failures such as premature release or loss of support. The selected chunk is then decoded into a joint-space target a_t for execution. We use fingertip-space motion as the shared upstream interface because it avoids exposing embodiment-specific joint coordinates to teleoperation or policy modules, while still providing enough geometric structure for dexterous manipulation.

Algorithm overview. DEXFLEX is a reusable low-level controller for this refinement problem (Fig. 2). Its design follows a simple propose–predict–select principle. First, a state-conditioned motion prior proposes feasible short-horizon fingertip motions from the current tactile–proprioceptive state, independent of where the draft comes from. Second, a latent world model predicts the contact–proprioceptive consequence of each candidate. Third, a learned score selects the candidate that best preserves the upstream draft while avoiding contact failures, and an action decoder converts the selected chunk into a joint-space target. We refer to this procedure as command-conditioned posterior correction: the prior defines the feasible motion support, while command dependence enters through consequence-aware scoring over a finite candidate set. We describe DEXFLEX in three parts: state-conditioned motion proposals (Sec. 3.1), consequence-aware scoring (Sec. 3.2), and Draft–Dream–Select execution with action decoding (Sec. 3.3).

3.1 State-Conditioned Motion Proposals

Contact–proprioceptive state. DEXFLEX encodes recent proprioception and fingertip tactile observations into a controller state $s_t = (z_t, g_t)$. The continuous latent $z_t = [p_t, c_t]$ combines a proprioceptive token p_t with a structured contact token c_t. To preserve finger-level interaction structure, each fingertip taxel field is encoded separately and fused with finger-local proprioceptive features; the resulting per-finger tokens are aggregated with a lightweight attention module and a global contact token. This structure preserves information about which fingers are carrying load, being unloaded, or likely to transfer support, instead of collapsing all tactile readings into a single flattened vector. We also infer a low-rate discrete latent g_t from a longer state window, intended to capture slowly varying manipulation regimes such as holding, pivoting, regrasp preparation, and release. The encoder is trained jointly with the motion prior and then kept fixed for consequence prediction and scoring.

Dexterous motion prior. Given s_t, the motion prior proposes short-horizon fingertip-motion chunks that are consistent with the current contact state, independent of the upstream draft. We instantiate the prior as a state-conditioned consistency-flow model [31], which provides expressive proposals with few function evaluations for online control. The flow network uses a lightweight transformer conditioned on the structured state tokens $\{p_t, c_t, g_t\}$ through cross-attention, allowing proposal generation to depend on both hand configuration and per-finger contact context. The prior is trained on expert motion chunks from the demonstration dataset $\mathcal{D}$ using flow matching with a shortcut-consistency loss; details are given in Appendix. At inference, it is used only as a proposal generator: it defines the feasible motion support from which DEXFLEX later selects using the upstream draft and predicted contact consequence. We apply tactile conditioning dropout during training to improve robustness to missing or noisy tactile readings.

3.2 Consequence-Aware Scoring

The state-conditioned prior can produce many feasible motion chunks, but feasibility alone does not determine which chunk should be executed under the current draft. Two candidates may be similarly plausible and similarly close to $\tilde{M}_t$, while only one preserves a supporting contact or avoids premature release. DEXFLEX resolves this ambiguity by predicting the short-horizon consequence of each candidate and scoring candidates using both draft alignment and predicted contact outcome.

Latent world model. Given the current controller state s_t and a candidate chunk M, a latent world model F_ω predicts the terminal contact–proprioceptive latent $\hat{z}_{t+H} = F_\omega(s_t, M)$ after executing the chunk. Because the horizon is short, we keep the slow discrete latent g_t fixed during this dreamed rollout and predict only the continuous latent. The model is trained by latent regression in simulation using expert chunks, prior samples, and locally perturbed expert chunks. For each candidate, the target latent is obtained by replaying the chunk from the corresponding simulator state and encoding the resulting tactile–proprioceptive observation. Training on these candidate sources keeps the consequence model calibrated for the motions it will evaluate during selection, rather than only for expert demonstrations.

Command- and consequence-aware score. The score $S_\eta(s_t, \tilde{M}_t, M, F_\omega(s_t, M))$ assigns a scalar value to each candidate. It uses s_t to interpret the current contact context, $\tilde{M}_t$ as evidence of upstream intent, M as the candidate motion, and $F_\omega(s_t, M)$ as the candidate's predicted consequence. This allows the controller to prefer a candidate that slightly modifies the draft to preserve support over one that matches the draft more closely but is predicted to break contact.

Since the dataset contains expert state–motion–action tuples but no paired examples of imperfect drafts and desired corrections, we train the score with synthetic proxy drafts. For an expert chunk M_t, we construct $\tilde{M}_t = \mathcal{A}_{\mathrm{cmd}}(M_t)$ using mild masking, small perturbations, and coarse temporal corruption. The score is trained as a contrastive ranker: the expert chunk is the positive candidate, while negatives include prior samples, hard negatives from other states with similar proxy drafts, and proposal-matched negatives from the draft-seeded branch. This training uses proxy drafts only as

noisy intent cues; at deployment, the candidate set is still generated from the state-conditioned prior, and the real upstream draft enters only through scoring. Implementation details and negative-mix ablations are provided in Appendix.

3.3 Draft–Dream–Select Execution

At deployment, DEXFLEX executes a simple Draft–Dream–Select loop. Given the current state s_t and upstream draft $\tilde{M}_t$, it forms a finite candidate set from two proposal branches, predicts each candidate's consequence with F_ω, and selects the highest-scoring chunk:

$$\mathcal{C}_t = \mathcal{C}_t^{\text{pure}} \cup \mathcal{C}_t^{\text{draft}}, \qquad M_t^\star = \arg\max_{M \in \mathcal{C}_t} S_\eta\big(s_t, \tilde{M}_t, M, F_\omega(s_t, M)\big). \tag{1}$$

The pure-prior branch samples from the state-conditioned prior and provides recovery options when the draft is unsafe or incomplete. The draft-seeded branch initializes the same flow from a noised version of $\tilde{M}_t$ at an intermediate noise level, providing local refinements that remain responsive to the upstream command. Importantly, the flow model itself is still conditioned only on s_t; the draft changes the proposal initialization used for finite-candidate search, not the learned motion prior.

Action decoding. The selected fingertip-motion chunk is converted into an executable joint-space target by an inverse-dynamics decoder, $a_t = I_\phi(s_t, M_t^\star, F_\omega(s_t, M_t^\star))$. Conditioning the decoder on the dreamed consequence helps disambiguate cases where similar fingertip displacements require different joint realizations, such as consolidating a grasp, unloading a support finger, or entering a release transition. At runtime, selection is performed in a receding-horizon manner: DEXFLEX executes a short prefix of the selected chunk, updates the tactile–proprioceptive state, and then repeats Draft–Dream–Select. Timing, proposal counts, and control-frequency details are provided in Appendix.

Training and reuse. We train the components sequentially to keep each interface stationary: the encoder and motion prior are trained first, followed by the consequence model, the score, and finally the action decoder. After training, the same low-level module can be reused with any upstream source that provides fingertip-motion drafts in the same command space. In shared control, the draft comes from glove teleoperation or retargeting; in policy learning, a high-level policy predicts the draft and DEXFLEX executes it as a frozen contact-aware low-level controller.

4 Experiments

We evaluate DEXFLEX as a reusable contact-aware low-level controller through three questions. **Q1:** Can it correct degraded upstream commands while preserving intended motion? **Q2:** Can it assist human users in real-world long-horizon tool use? **Q3:** Can it serve as a frozen low-level interface for visuomotor policy learning? Together, these settings test whether the same learned controller can preserve upstream intent while repairing contact-critical execution details. Unless otherwise noted, DEXFLEX is trained once and reused across settings without retraining.

4.1 Experimental Setup

Training data. We train DEXFLEX from simulated tactile–proprioceptive rollouts in Isaac Gym. The data is collected with a 16-DoF LEAP four-fingered hand or a 22-DoF SHARPA five-fingered hand, each mounted on a 7-DoF xArm, over large-scale procedurally generated objects. RL experts trained on an Any-Goal-Pose Reaching task [32] provide expert fingertip-motion chunks and joint targets. This single dataset trains all learned components of DEXFLEX; rollout scale, filtering, and training details are provided in Appendix.

Real-world platform. For shared-control experiments, we use a 16-DoF LEAP hand mounted on a 7-DoF xArm7. Human finger motion is captured by a Rokoko Smart Glove, and wrist pose is tracked by a Vive Ultimate tracker. The teleoperation stack retargets human motion to fingertip and wrist targets and solves inverse kinematics with Mink [33] at 50 Hz. Each fingertip is equipped with a 10×6 FlexiTac tactile sensor [34, 35], whose spatial taxel layout is used by the structured contact encoder

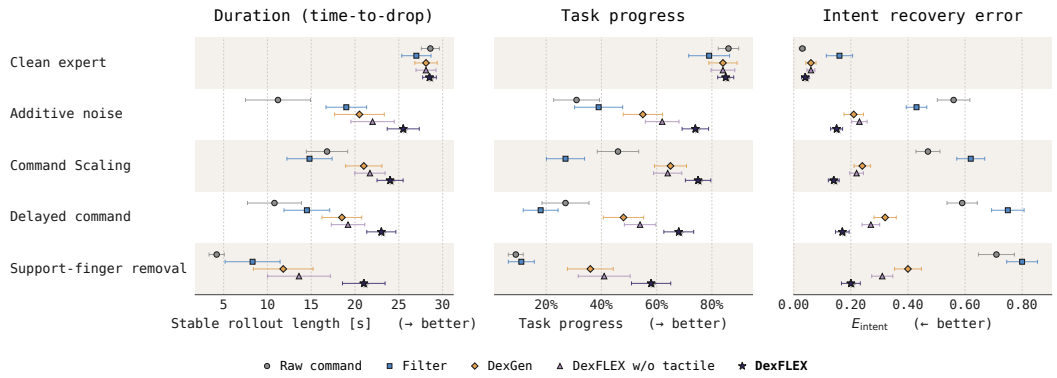

Figure 3: **Robustness under command degradation.** Expert commands are corrupted with additive noise, command scaling, delay, and support-finger removal. We evaluate whether each controller preserves contact stability while recovering the expert intent rather than merely damping motion.

in Sec. 3.1. Sensor calibration, synchronization, and inference-time proposal hyperparameters are provided in Appendix.

Baselines and metrics. For command-correction experiments, we compare against RAW, which directly executes the upstream draft; FILTER, which applies low-pass filtering and command clipping; and DEXGEN, a diffusion-based prior-only correction baseline [4] without latent consequence prediction or consequence-aware scoring. We also include DEXFLEX W/O TACTILE, which removes tactile/contact tokens while keeping the rest of the pipeline unchanged. In simulation, we report normalized rollout duration, task progress, and intent recovery error. In real-world teleoperation, we report success rate (SR), completion time over successful trials (CT), and normalized holding time (NHT), the fraction of the task horizon with a stable functional grasp. Each real-world teleoperation task is evaluated over 20 trials per method across two operators after a 30-minute per-interface familiarization period. For policy learning, all methods use 50 demonstrations per task and the same image-based flow matching policy backbone [36], varying only the low-level action interface.

4.2 Recovering Intent from Degraded Commands

A useful low-level controller should not simply suppress unsafe commands; it should recover the intended motion when the draft is noisy, delayed, or incomplete, while intervening when the draft would destabilize contact. Simulation provides clean expert commands $u_t^\star$, allowing us to measure recovery against an external reference. Starting from clean expert commands $u_t^\star$, we construct four degraded command streams: ADDITIVE NOISE, $\tilde{u}_t = u_t^\star + \xi_t$ with $\xi_t \sim \mathcal{U}(-\alpha, \alpha)$; COMMAND SCALING, $\tilde{u}_t = \beta_t u_t^\star$; DELAYED COMMAND, $\tilde{u}_t = u_{t-\Delta}^\star$; and SUPPORT-FINGER REMOVAL, which replaces the commands of load-bearing fingertips with release-like motions. We report normalized rollout duration, task progress, and intent recovery error $E_{\text{intent}} = \frac{1}{T}\sum_t \frac{\|\hat{u}_t - u_t^\star\|_2}{\|u_t^\star\|_2 + \epsilon}$, where $\hat{u}_t$ is the executed command after low-level correction. Lower intent error means the controller recovers the clean expert command rather than merely staying close to the corrupted draft. Fig. 3 shows that DEXFLEX improves all three metrics across command degradations. The largest gains occur under SUPPORT-FINGER REMOVAL, where RAW often drops the object by removing stabilizing contacts, while DEXFLEX selects motions that preserve support and still follow the command direction. The gains are not explained by conservative damping: FILTER can improve duration by suppress-

Variant	Dur. ↑	Prog. ↑	Intent ↓
DEXFLEX	**0.86**	**0.78**	**0.18**
w/o predictive score	0.52	0.46	0.45
w/o tactile/contact	0.62	0.55	0.36
w/o consequence pred.	0.64	0.57	0.34
w/o pure-prior prop.	0.66	0.61	0.25
w/o draft-seeded prop.	0.74	0.66	0.38
w/o slow phase token	0.81	0.74	0.22

Table 1: **Component ablations.** Metrics are averaged over the four command-degradation settings. *w/o predictive score* replaces the learned score with nearest-to-draft selection over the same proposal set.

Method	Brush	Nut	Hammer	Screwdrv.	Spatula	Toast	Avg SR ↑	Avg CT ↓	Avg NHT ↑
Raw Teleop	50	30	40	30	15	10	29.2	27.5	0.42
Filter	60	40	50	40	25	20	39.2	33.8	0.55
DexGen	70	55	60	55	40	35	52.5	31.2	0.65
DexFLEX w/o Tactile	75	60	65	60	50	45	59.2	30.8	0.71
DexFLEX	**90**	**80**	**85**	**80**	**70**	**65**	**78.3**	32.0	**0.85**

Table 2: **Real-world teleoperation results.** We report per-task success rate (SR, %) and average SR / CT / NHT across tasks. CT is completion time in seconds over successful trials. NHT is normalized holding time. Each method is evaluated for 20 trials per task across two operators. Averages are computed over task-level means.

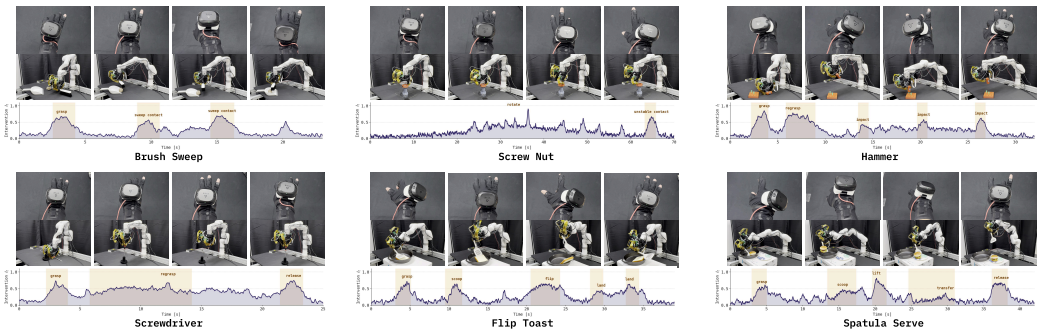

Figure 4: **Selective intervention during real-world tool use.** We visualize intervention magnitude $I_t = \|\hat{u}_t - u_t\|_2/(\|u_t\|_2 + \epsilon)$ along representative rollouts. DexFLEX keeps intervention low during benign user motion and produces peaks near contact-critical moments such as release, impact, unstable tool contact, and regrasp.

ing motion, but increases intent error and reduces task progress. DexGen improves over simple baselines through prior-based correction, and DexFLEX w/o Tactile improves further through consequence-aware selection, but the full model performs best under contact-critical corruptions by using tactile state to identify load-bearing fingertips.

Component ablations. Table 1 isolates the main design choices. Removing consequence-aware scoring causes the largest drop, showing that selection over prior samples is not reducible to nearest-to-draft matching. Removing tactile input or consequence prediction also degrades performance, especially under contact-critical corruptions, confirming that both current contact state and predicted contact outcome are needed for robust correction. The proposal branches are complementary: pure-prior proposals provide recovery options when the draft is unsafe, while draft-seeded proposals preserve responsiveness to upstream intent.

4.3 Real-World Shared Control for Tool Use

We next evaluate whether the same controller assists human operators in real-world long-horizon tool use teleoperation. Unlike the simulated corruptions, these tasks contain natural user errors, tool–environment contact, in-hand adjustment, and functional grasp transitions. We evaluate six tasks: Brush Sweep (maintain a brush grasp while sweeping), Screw Nut (align and rotate a nut), Hammer (grasp and strike with impact), Screwdriver (grasp a screwdriver, insert it into a base, and rotate), Spatula Serve (scoop a hamburger with a spatula and place it on a plate), and Flip Toast (flip a toast slice with a spatula).

Reliability across tool-use tasks. Table 2 shows that DexFLEX more than doubles raw teleoperation's average SR (29.2 to 78.3) and roughly doubles NHT (0.42 to 0.85), with a 25.8-point SR margin over DexGen. DexFLEX w/o Tactile also improves over DexGen (52.5 to 59.2 SR), suggesting that consequence-aware selection helps beyond prior-only correction, while the full model gains another 19.1 points by using tactile state. The gains are largest on contact-critical tasks such as Spatula Serve and Flip Toast, where premature release or unstable support frequently causes failure. The improvement is not explained by uniformly slowing down execution: DexFLEX's av-

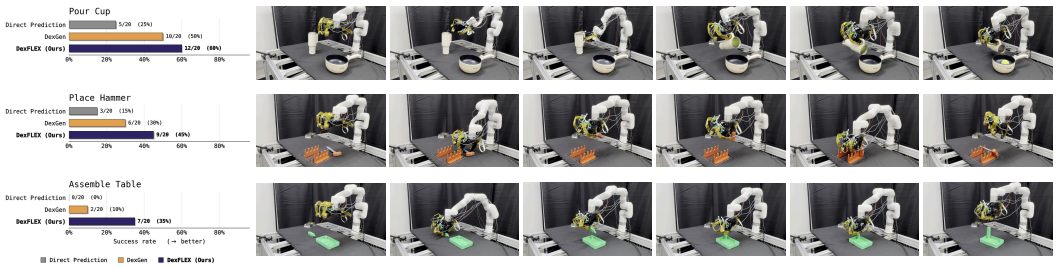

Figure 5: **Policy learning with different action interfaces.** All methods use the same image-based Flow-matching policy and 50 demos per task. Direct joint-action prediction is compared with fingertip-motion prediction executed by a prior-only DexGen controller and by DEXFLEX.

erage CT (32.0 s) lies within the inter-method range of 27.5 to 33.8 s, while substantially improving success and holding stability.

Selective intervention. Fig. 4 shows that DEXFLEX does not continuously override the operator. Intervention remains small during benign alignment and transport, but rises near contact-critical events such as release, impact, scoop, and regrasp. Thus, DEXFLEX improves reliability by selectively repairing low-level contact failures while leaving high-level task execution to the user.

4.4 Low-Level Interface for Policy Learning

Finally, we test whether DEXFLEX can serve as a frozen low-level interface for visuomotor policy learning. We evaluate three contact-rich tasks: POUR CUP (maintain a handle grasp while pouring), PLACE HAMMER (grasp, reorient, and place into a holder), and ASSEMBLE TABLE (grasp, reorient, insert, and screw until lit). For each task, we collect 50 demos and train the same Flow-matching policy [36]. The methods differ only in the action interface: direct joint-command prediction, fingertip-motion prediction executed by a DexGen-style prior-only controller, or fingertip-motion prediction routed through the frozen DEXFLEX module. The DexGen-style baseline follows [4]; reproduction details are provided in Appendix. Each policy is evaluated for 20 rollouts per task.

Fig. 5 shows that DEXFLEX achieves the highest SR on all three tasks, reaching 46.7% average SR compared with 13.3% for direct joint actions and 30.0% for DexGen. With only 50 demonstrations per task, direct joint-action policies are brittle because the policy must learn both task intent and contact-sensitive execution in a high-dimensional joint space. Predicting fingertip-motion drafts reduces this burden by letting the image policy express local task intent, while the low-level controller maps the draft back onto a contact-consistent dexterous motion distribution. A prior-only controller provides this projection only through motion plausibility, whereas DEXFLEX additionally uses predicted contact consequence to choose among plausible executions. This improves policy success by 3.5× over direct joint actions and 1.56× over DexGen.

5 Conclusions

We presented DEXFLEX, a contact-aware foundation controller for command-guided dexterity. DEXFLEX refines upstream fingertip-motion drafts through a propose–predict–select procedure that combines state-conditioned motion proposals, latent contact-consequence prediction, and consequence-aware selection. Across degraded-command simulation, real-world shared control, and visuomotor policy learning, the same trained controller improves contact-consistent execution without retraining, making it a reusable low-level interface beneath different command sources.

Limitations and future work. DEXFLEX currently assumes a fixed robot embodiment, tactile sensing, and a shared fingertip-motion command space. It can still fail under tactile sim-to-real gaps or when the upstream draft lies outside the support of the learned prior. Future work will study cross-embodiment transfer, tactile adaptation, longer-horizon contact prediction, and tighter coupling between high-level policy learning and low-level contact-consistent execution.

References

[1] S. Reddy, A. D. Dragan, and S. Levine. Shared autonomy via deep reinforcement learning. *arXiv preprint arXiv:1802.01744*, 2018.

[2] C. Schaff and M. R. Walter. Residual policy learning for shared autonomy. *arXiv preprint arXiv:2004.05097*, 2020.

[3] T. Yoneda, L. Sun, G. Yang, B. Stadie, and M. Walter. To the noise and back: Diffusion for shared autonomy. *arXiv preprint arXiv:2302.12244*, 2023.

[4] Z.-H. Yin, C. Wang, L. Pineda, F. Hogan, K. Bodduluri, A. Sharma, P. Lancaster, I. Prasad, M. Kalakrishnan, J. Malik, et al. Dexteritygen: Foundation controller for unprecedented dexterity. *arXiv preprint arXiv:2502.04307*, 2025.

[5] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023.

[6] H. Qi, B. Yi, S. Suresh, M. Lambeta, Y. Ma, R. Calandra, and J. Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023.

[7] I. Guzey, B. Evans, S. Chintala, and L. Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play. *arXiv preprint arXiv:2303.12076*, 2023.

[8] 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.

[9] 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*, pages 2165–2183. PMLR, 2023.

[10] K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. R. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, b. ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In J. Lim, S. Song, and H.-W. Park, editors, *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 17–40. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/black25a.html.

[11] M. A. Lee, Y. Zhu, P. Zachares, M. Tan, K. Srinivasan, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg. Making sense of vision and touch: Learning multimodal representations for contact-rich tasks. *IEEE Transactions on Robotics*, 36(3):582–596, 2020.

[12] G. Ye, Z. Zhang, X. Zhao, S. Wu, H. Lu, S. Lu, and H. Liu. Learning to feel the future: Dreamtacvla for contact-rich manipulation. *arXiv preprint arXiv:2512.23864*, 2025.

[13] W. Wan, J. Fu, X. Yuan, Y. Zhu, and H. Su. Lodestar: long-horizon dexterity via synthetic data augmentation from human demonstrations. In *9th Annual Conference on Robot Learning*, 2025.

[14] Y. Zheng, S. Gu, W. Li, Y. Zheng, Y. Zang, S. Tian, X. Li, C. Hao, C. Gao, S. Liu, et al. Omnivta: Visuo-tactile world modeling for contact-rich robotic manipulation. *arXiv preprint arXiv:2603.19201*, 2026.

[15] C. Higuera, S. Arnaud, B. Boots, M. Mukadam, F. R. Hogan, and F. Meier. Visuo-tactile world models. *arXiv preprint arXiv:2602.06001*, 2026.

[16] H. Yuan, W. Yi, Z. Zhang, W. Chen, Y. Mo, J. Yin, X. Li, X. Zeng, C. Wen, C. Lu, et al. Vtam: Video-tactile-action models for complex physical interaction beyond vlas. *arXiv preprint arXiv:2603.23481*, 2026.

[17] Y. Niu, Z. Fang, B. Chen, S. Zhou, R. Senthilkumaran, H. Zhang, B. Chen, C. Qiu, H. E. Tseng, J. Francis, et al. Learning versatile humanoid manipulation with touch dreaming. *arXiv preprint arXiv:2604.13015*, 2026.

[18] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox. Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170. IEEE, 2020.

[19] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *arXiv preprint arXiv:2307.04577*, 2023.

[20] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

[21] C. Pan, C. Wang, H. Qi, Z. Liu, H. Bharadhwaj, A. Sharma, T. Wu, G. Shi, J. Malik, and F. Hogan. Spider: Scalable physics-informed dexterous retargeting. *arXiv preprint arXiv:2511.09484*, 2025.

[22] D. Aarno, S. Ekvall, and D. Kragic. Adaptive virtual fixtures for machine-assisted teleoperation tasks. In *Proceedings of the 2005 IEEE international conference on robotics and automation*, pages 1139–1144. IEEE, 2005.

[23] H. K. Kim, J. Biggs, W. Schloerb, M. Carmena, M. A. Lebedev, M. A. Nicolelis, and M. A. Srinivasan. Continuous shared control for stabilizing reaching and grasping with brain-machine interfaces. *IEEE Transactions on Biomedical Engineering*, 53(6):1164–1173, 2006.

[24] S. Schröer, I. Killmann, B. Frank, M. Völker, L. Fiederer, T. Ball, and W. Burgard. An autonomous robotic assistant for drinking. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 6482–6487. IEEE, 2015.

[25] W. Schwarting, J. Alonso-Mora, L. Pauli, S. Karaman, and D. Rus. Parallel autonomy in automated vehicles: Safe motion generation with minimal intervention. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1928–1935. IEEE, 2017.

[26] A. D. Dragan and S. S. Srinivasa. A policy-blending formalism for shared control. *The International Journal of Robotics Research*, 32(7):790–805, 2013.

[27] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell. Shared autonomy via hindsight optimization for teleoperation and teaming. *The International Journal of Robotics Research*, 37(7):717–742, 2018.

[28] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik. Learning visuotactile skills with two multifingered hands. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5637–5643. IEEE, 2025.

[29] S. Suresh, H. Qi, T. Wu, T. Fan, L. Pineda, M. Lambeta, J. Malik, M. Kalakrishnan, R. Calandra, M. Kaess, et al. Neuralfeels with neural fields: Visuotactile perception for in-hand manipulation. *Science Robotics*, 9(96):eadl0628, 2024.

[30] Z. Xu, Y. Li, X. Wu, T. Qiu, and L. Shao. Fingereye: Continuous and unified vision-tactile sensing for dexterous manipulation. *arXiv preprint arXiv:2604.20689*, 2026.

[31] 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.

[32] K. Kedia, T. G. W. Lum, J. Bohg, and C. K. Liu. Simtoolreal: An object-centric policy for zero-shot dexterous tool manipulation. *arXiv preprint arXiv:2602.16863*, 2026.

[33] K. Zakka. Mink: Python inverse kinematics based on MuJoCo, Feb. 2026. URL https://github.com/kevinzakka/mink.

[34] B. Huang, Y. Wang, X. Yang, Y. Luo, and Y. Li. 3d-vitac: Learning fine-grained manipulation with visuo-tactile sensing. *arXiv preprint arXiv:2410.24091*, 2024.

[35] B. Huang and Y. Li. Flexitac: A low-cost, open-source, scalable tactile sensing solution for robotic systems. *arXiv preprint arXiv:2604.28156*, 2026.

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