

FingerEye: Learning Dexterous Manipulation with Continuous Vision-Tactile Sensing

Zhixuan Xu^{1*}, Yichen Li^{1,2*}, Xuanye Wu^{2,3}, Tianyu Qiu^{2,4}, Lin Shao^{1,2†}

¹National University of Singapore ²RoboScience

³Huazhong University of Science and Technology ⁴South China University of Technology

*Equal contribution †Corresponding author

<https://nus-lins-lab.github.io/FingerEyeWeb/>

Abstract—Dexterous robotic manipulation requires perception that remains informative from pre-contact approach to contact initiation and post-contact control. We introduce FingerEye, a sensing and learning framework that strengthens robotic dexterity through continuous vision-tactile feedback throughout interaction. On the sensing side, FingerEye integrates binocular RGB cameras with a compliant contact interface to support perception both before and after contact. Before contact, the fingertip cameras provide close-range visual cues and implicit stereo for precise approach and object localization. After contact, marker-tracked deformation of the compliant ring provides a proxy for contact wrench estimation. On the learning side, we build real-and-sim infrastructure for data collection and evaluation, systematically study policy-interface designs for learning with multiple FingerEye sensors, and use group-structured modality fusion to reduce modality shortcuts and better exploit distributed fingertip feedback. Across seven contact-sensitive task settings, FingerEye improves wrist-only policy by over 30 percentage points in mean success rate in both simulation and the real world.

I. INTRODUCTION

“Manipulation refers to an agent’s control of its environment through selective contact” [1]. In dexterous manipulation, selective contact requires sensing that remains informative before contact, during contact initiation, and after contact. For example, standing a coin upright on a table (Fig. 1) requires the robot to localize the thin coin edge *before contact*, detect fingertip contact onset *during contact initiation*, and regulate contact forces *after contact* to prevent slipping or toppling. Similar seeing-to-touching transitions appear in chip picking, letter retrieving, and syringe manipulation, where the hand must resolve local geometry, choose contact points, and regulate interaction forces to avoid damage or slippage.

Despite the importance of continuous sensing throughout interaction, existing systems mainly rely on two separate sources: external vision and tactile sensing. External vision from environment- or wrist-mounted cameras helps robots localize objects, but provides only indirect observations of contact when fingers occlude small contact patches [2, 3, 4]. Tactile and vision-based tactile sensors capture rich contact information, but are primarily informative after contact is established [5, 6, 7, 8, 9]. This separation leaves the most fragile moment in dexterous manipulation, the transition from seeing to touching, weakly observed, limiting precision and dexterity in contact-rich tasks.

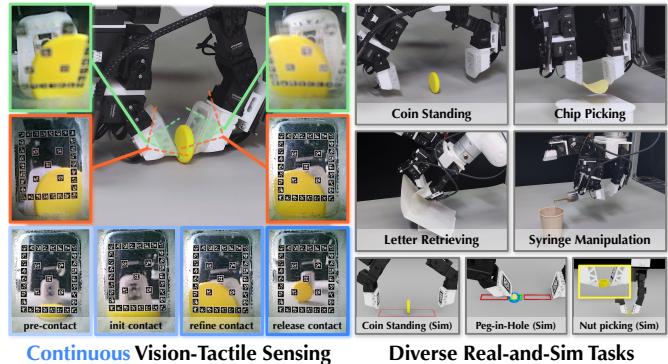


Fig. 1. **FingerEye overview.** FingerEye provides continuous fingertip feedback from pre-contact approach to contact refinement and release, and is evaluated on contact-sensitive real-world and simulation tasks.

Recently, see-through-skin sensors [10, 11, 12, 13] have emerged as a promising middle ground between vision and tactile sensing. Despite their potential, these approaches are still mostly demonstrated on gripper-centric, low-dexterity tasks. What still limits their precision and dexterity on tasks such as coin standing? We identify five subtle yet critical issues: 1) **discontinuous or unnatural visual observations** from semi-transparent optical interfaces [10, 14, 15, 12], active or non-natural self-illumination [16], illumination switching [13, 17], or mechanically reconfigurable components [18, 19]; 2) **fragile contact perception in open environments**, where marker- or keypoint-based tactile perception can be unreliable under changing backgrounds, lighting, occlusions, and subtle contact-induced motion [20, 21]; 3) **limited dexterous-hand compatibility**, where bulky, flat, or front-facing designs can be difficult to mount on compact fingertips and often miss peripheral or non-frontal contacts in confined spaces; 4) **missing real-and-sim learning infrastructure** for learning multisensory policies and evaluating them at scale on dexterous tasks; and 5) **underdeveloped end-to-end policy interfaces**, where naive fusion or conditioning can cause the policy to bypass local FingerEye feedback and rely on easier but less contact-informative cues.

In this paper, we address these issues through careful design of both the sensing and learning interfaces. 1) We design a physical **sensing interface** for dexterous fingertips. FingerEye combines binocular RGB cameras with a transparent contact surface for natural visual perception and implicit stereo depth

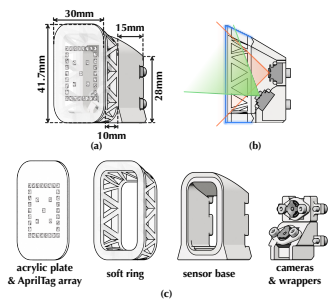


Fig. 2. **FingerEye Sensing Interface Design.** (a) Dimensions. (b) Cross-sectional view: fields of view of the **root** and **tip** cameras, and the **contact sensing region**. (c) Exploded view.

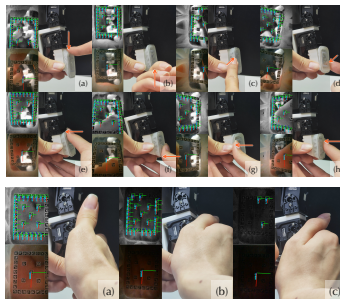


Fig. 3. **Robustness of tag-array pose estimation.** Top: force/occlusion perturbations. Bottom: lighting variation. Multi-tag PnP uses all visible corners, avoiding single-marker dependence.

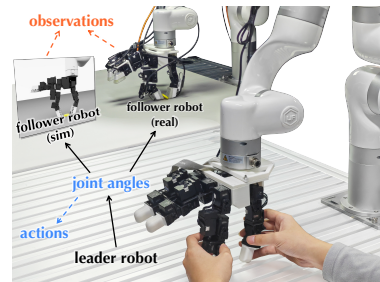


Fig. 4. **Learning-interface collection.** Matched leader-follower setup records synchronized robot, wrist, and fingertip streams.

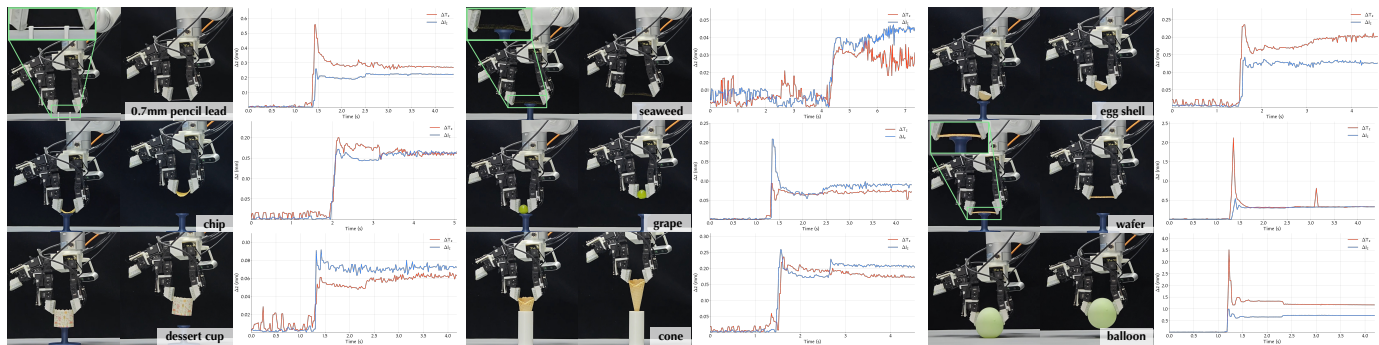


Fig. 5. **Visualization of the setup and fingertip normal deformation in delicate grasping experiments.** Temporal curves of normal displacement Δz measured at the **thumb** and **index** fingertips.

before contact. After contact, external forces and torques deform a surrounding compliant ring, whose deformation is captured through robust tag-array pose estimation as a proxy for contact wrench estimation. This design preserves a single RGB sensing pipeline without active illumination or sensing-mode switching, while remaining compact, low-cost, and capable of sensing both frontal and peripheral contacts. 2) We develop a **learning interface** for multi-sensor FingerEye feedback. We build real-and-sim infrastructure for data collection, policy training, and scalable evaluation on diverse contact-sensitive tasks, and study how multi-fingertip observations should be fused for action prediction. Since FingerEye provides local fingertip feedback while wrist vision and proprioception provide easier global cues, naive fusion can cause policies to fit demonstrations while underusing contact-local information. To address this, we use **group-structured modality fusion**: modality groups are first encoded into structured representations, and action queries then fuse them through group-conditioned decoding to balance cross-modal fusion.

The final system, **FingerEye**, provides an integrated sensing and learning framework for contact-rich dexterous manipulation. We evaluate FingerEye across diverse simulation and real-world tasks, including coin standing, chip picking, letter retrieving, syringe manipulation, nut picking, and peg-in-hole pushing.

II. METHOD

A. Sensing Interface

The key sensing-interface question is how a fingertip sensor can see clearly before contact, sense contact robustly after contact, and still fit naturally on a dexterous hand (Fig. 2). We summarize the main design and sensing capabilities below, with calibration and fabrication details in Appendix B.1 and Appendix A.2.

Complementary binocular vision for pre-contact alignment. A short-range tip camera observes the transparent contact surface, while a tilted root camera preserves a broader approach view. Together, the two natural-RGB streams provide local appearance and implicit depth cues for object localization and contact alignment without active illumination or sensing-mode switching; empirically, the full binocular input outperforms the monocular variant in both simulation and real-world tasks (Table I).

Peripheral compliance for contact sensing without blocking vision. A clear acrylic window keeps the central optical path unobstructed, while a surrounding silicone ring converts frontal and side loads into plate-pose changes. This expands the effective sensing region for pushing, wedging, and delicate grasping while preserving visual continuity through contact.

Multi-tag deformation tracking for robust plate-pose estimation. A custom AprilTag array converts contact-driven ring deformation into a six-dimensional plate-pose change Δp . Instead of combining single-marker estimates, we aggregate all visible tag corners in one multi-tag PnP problem; under force

perturbations, occlusion, or lighting changes, the remaining visible corners provide redundant constraints for stable pose tracking (Fig. 3).

Sensitive wrench estimation from deformation. The estimated Δp is used as the contact-induced deformation signal for wrench estimation. Across synchronized wrench–pose pairs, it predicts applied forces and torques; sensitivity analysis [22, 23] gives detectable forces of [4.30, 4.22, 9.93] mN and torques of [0.32, 0.13, 8.55] mN·m. The delicate-grasping traces in Fig. 5 further illustrate the resulting contact-onset signal; Appendix B.1 gives the calibration setup and held-out mapping results.

Wedge-shaped, fingertip-scale, and reproducible hardware. The same sensing stack is packaged in a wedge-shaped module of $41.7 \times 30.0 \times 25.0$ mm, making it compatible with compact dexterous fingers and confined object interactions. Off-the-shelf cameras, 3D-printed parts, silicone, and acrylic keep the per-module cost at about \$60.19.

B. Learning Interface

We now study how to design and validate a learning interface that improves dexterous manipulation with FingerEye observations. The interface combines real-and-sim infrastructure for collecting contact-sensitive demonstrations and scalable evaluation with a structured policy design that better leverages local FingerEye feedback.

Robot platform and data collection interface. The real setup uses a fixed-base arm with a LEAP Hand [24], wrist RGB camera, and FingerEye modules on the fingertips. We collect demonstrations with a matched leader–follower hand: the passive leader provides intuitive hand motion, while the follower tracks streamed joint positions. This avoids the embodiment mismatch of direct human-hand retargeting [25] in contact-sensitive tasks. The system records synchronized wrist images, multi-fingertip FingerEye observations, plate poses, robot states, and actions at 10 Hz (Fig. 4); task settings are listed in Appendix E.2.

Simulation digital twin. For scalable evaluation and ablation, we construct a simulation counterpart of the real sensorized hand. Each simulated acrylic plate is modeled as a floating body connected to the sensor base by six calibrated virtual joints, so contact forces produce plate-pose changes consistent with the measured real sensor. This enables controlled comparisons of sensing and policy-interface designs under reproducible settings.

Policy interface design. A key challenge is not merely adding sensory streams, but designing a policy interface that lets local fingertip feedback influence decoded actions. Wrist-view observations and proprioception provide strong global cues, so contact-local FingerEye cues can be underused during fusion. Let $\mathcal{O}_t = (\mathcal{o}_t^1, \dots, \mathcal{o}_t^M)$ denote M observation groups, and let the policy predict an action chunk $\mathcal{A}_t = \pi_\theta(\mathcal{O}_t) = (\mathbf{a}_t, \dots, \mathbf{a}_{t+T_p-1})$, executing the first T_a actions before re-planning. Each group is tokenized as $\mathbf{U}_t^i = \tau_i(\mathcal{o}_t^i)$, and $\mathbf{U}_t = [\mathbf{U}_t^1, \dots, \mathbf{U}_t^M]$. We compare three encoder interfaces:

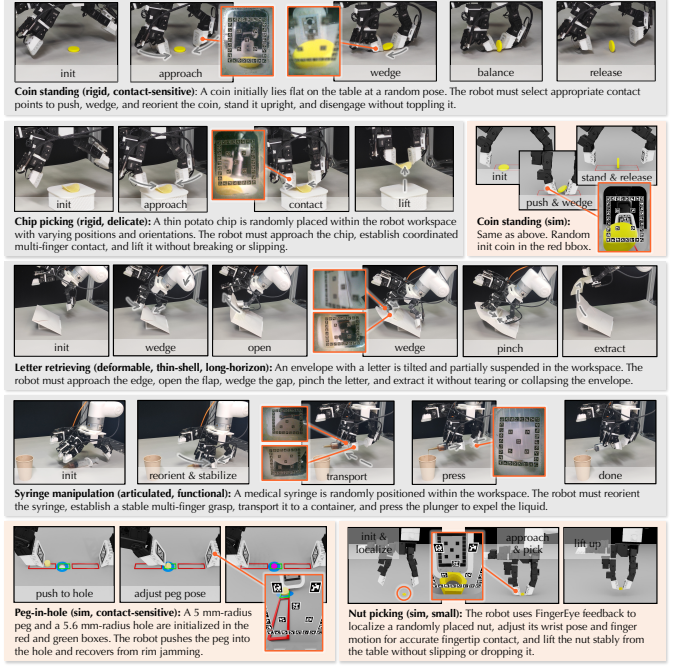


Fig. 6. **Task and rollout suite.** Real and simulated tasks stress precise approach, edge engagement, and post-contact stabilization, making the evaluation sensitive to fingertip-local feedback throughout the seeing-to-touching transition.

$$\begin{aligned} \mathbf{Z}_t^{\text{noenc}} &= \mathbf{U}_t, \\ \mathbf{Z}_t^{\text{flat}} &= E_{\text{flat}}(\mathbf{U}_t), \\ \mathbf{Z}_t^{\text{group}} &= [E_1(\mathbf{U}_t^1), \dots, E_M(\mathbf{U}_t^M)]. \end{aligned}$$

NoEnc provides no pre-decoder interaction, **FlatEnc** immediately mixes all local and global tokens, and **GroupEnc** first forms within-group representations so FingerEye views can integrate local geometry before cross-group fusion.

For decoding, let $\mathbf{Z}_t = [\mathbf{z}_t^1, \dots, \mathbf{z}_t^M]$ be the encoded sensory context and $\mathbf{h}_t^l$ be the action-query state at decoder layer l . The two decoder interfaces differ as

$$\begin{aligned} \tilde{\mathbf{h}}_{t,\text{flat}}^l &= \mathbf{h}_t^l + \text{CrossAttn}^l(\mathbf{h}_t^l, \mathbf{Z}_t), \\ \tilde{\mathbf{h}}_{t,\text{group}}^l &= \mathbf{h}_t^l + \frac{1}{M} \sum_{i=1}^M \text{CrossAttn}_i^l(\mathbf{h}_t^l, \mathbf{z}_t^i). \end{aligned}$$

FlatDec makes all modality tokens compete in one cross-attention normalization. **GroupDec** gives each modality group an independent residual path into the action query. Thus **FlatEnc+FlatDec** mixes observations early and decodes from one shared context, **GroupEnc+FlatDec** protects modality representations before a flat readout, **NoEnc+GroupDec** removes interaction before decoding but preserves grouped decoder access, and **GroupEnc+GroupDec** combines group-local encoding with group-conditioned decoding. We use this final grouped encoder-decoder design in the main FingerEye experiments; Appendix C.1 gives architecture and failure-analysis details.

III. EXPERIMENTS

We evaluate whether FingerEye improves dexterous manipulation by providing continuous fingertip observations that

TABLE I
SENSING MODALITY COMPARISON. SIMULATION REPORTS MEAN $\pm$ STD SR OVER SEEDS; REAL-WORLD TASKS REPORT SR OVER 20 TRIALS.

Input	Simulation SR (%)			Real-world SR (%)					
	Coin	Nut	Peg	Mean	Coin	Chip	Letter	Syringe	Mean
W	25.5 $\pm$ 8.3	38.7 $\pm$ 23.2	16.0 $\pm$ 6.6	26.7	25	35	30	60	37.5
W+M	56.4 $\pm$ 3.7	84.4 $\pm$ 5.2	36.5 $\pm$ 5.5	59.1	50	55	45	75	56.3
W+FE	65.6$\pm$4.8	88.2$\pm$4.2	43.8$\pm$4.2	65.9	65	70	65	85	71.3

TABLE II
FOCUSED SIMULATION ABLATIONS. CONTINUOUS FINGERTIP FEEDBACK AND GROUP-STRUCTURED FUSION BOTH IMPROVE MEAN SR.

Variant	Coin	Nut	Peg	Mean
<i>Continuous vs. post-contact sensing</i>				
W+T	27.3 $\pm$ 6.8	28.6 $\pm$ 5.1	16.5 $\pm$ 7.4	24.1
W+FE	65.6$\pm$4.8	88.2$\pm$4.2	43.8$\pm$4.2	65.9
<i>Policy interface with W+FE input</i>				
FlatEnc+FlatDec	43.6 $\pm$ 8.3	83.7 $\pm$ 6.1	18.2 $\pm$ 7.5	48.5
NoEnc+FlatDec	48.6 $\pm$ 8.3	80.7 $\pm$ 11.3	26.6 $\pm$ 7.6	52.0
NoEnc+GroupDec	54.9 $\pm$ 7.5	88.9$\pm$4.1	33.5 $\pm$ 8.0	59.1
GroupEnc+FlatDec	60.2 $\pm$ 9.1	85.4 $\pm$ 5.5	33.9 $\pm$ 5.3	59.8
GroupEnc+GroupDec	65.6$\pm$4.8	88.2$\pm$4.2	43.8$\pm$4.2	65.9

complement external vision. The experiments answer four questions: **Q1: FingerEye benefits.** Does adding FingerEye improve over wrist-only observation? **Q2: Binocular sensing.** Does the binocular design improve over a monocular fingertip variant? **Q3: Continuous sensing.** Does continuous fingertip feedback improve over post-contact-only tactile maps? **Q4: Policy interface.** Which fusion interface best exploits distributed FingerEye feedback?

Tasks and evaluation protocol. Fig. 6 summarizes seven simulated and real-world tasks spanning precise approach, edge engagement, and post-contact stabilization. Demonstration datasets contain 50/50/200 trajectories for the three simulation tasks and 60/30/60/40 trajectories for the four real-world tasks. Simulation uses 3 training seeds and 3 evaluation seeds with 64 rollouts per evaluation seed; real-world evaluation uses 20 rollouts per method and task. We report task success rate (SR), with **W** for wrist RGB, **M** for monocular FingerEye, **FE** for full binocular FingerEye, and **T** for post-contact tactile maps.

O1: FingerEye improves contact-sensitive manipulation beyond wrist-only observation. Table I shows that adding binocular FingerEye to wrist observations improves mean SR over **W** by 39.1 points in simulation and 33.8 points in real-world tasks. The gains are largest when precise local contact geometry matters: wrist-only policies often miss final edge alignment, lose small objects during engagement, or apply force from an unstable side. The qualitative failures in Fig. 7 show the same pattern.

O2: Binocular fingertip feedback reduces monocular ambiguity near contact. The full binocular **W+FE** input outperforms the monocular **W+M** variant by 6.8 points in simulation mean SR and 15.0 points in real-world mean SR (Table I). The remaining **W+M** failures in Fig. 7 are consistent with limited single-view coverage: the policy can see local contact but still cannot always disambiguate edge side, depth, or hidden

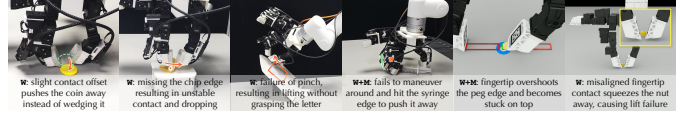


Fig. 7. **Modality failure modes.** Wrist-only policies often fail from missing local contact geometry, while monocular fingertip feedback improves local coverage but can still fail from single-view ambiguity.

engagement state. The ordering **W+FE** > **W+M** > **W** across both simulation and real-world tasks supports the binocular sensing design.

O3: Continuous fingertip sensing matters before contact is stable. Table II compares **W+FE** with **W+T**, where **T** denotes simulated post-contact tactile maps encoded by the same policy. **W+T** remains close to wrist-only performance, while **W+FE** reaches 65.9% mean SR. This gap matches the task structure in Fig. 6: coin standing and peg pushing require pre-contact pose refinement; chip picking and letter retrieving require narrow edge engagement; syringe manipulation and nut picking require stabilization after the object starts sliding or rotating. The delicate-grasping traces in Fig. 5 show the complementary post-contact use: normal deformation changes sharply at first contact, giving a per-finger stop cue.

O4: Group-structured modality fusion better uses FingerEye feedback. With **W+FE** input fixed, **GroupEnc+FlatDec** improves over **FlatEnc+FlatDec** by 11.3 points, showing the value of forming group-level representations before fusion. **NoEnc+GroupDec** improves over **NoEnc+FlatDec** by 7.1 points, showing that grouped decoder access also matters. Combining both gives the best result: **GroupEnc+GroupDec** reaches 65.9% mean SR, 13.9 points above the strongest non-grouped baseline. All modality comparisons use the same datasets and **GroupEnc+GroupDec** policy, while the policy-interface ablation fixes **W+FE** and changes only the fusion path; Fig. 13 gives the architecture variants.

IV. LIMITATIONS AND CONCLUSIONS

FingerEye improves contact-sensitive dexterous manipulation, but several limitations remain. First, the camera placement, tag layout, and ring stiffness are manually designed; changing the hand geometry would require recalibrating the wrench–pose relation and retesting the observation grouping. Automatically optimizing sensor geometry could improve fingertip coverage, reduce blind regions, and better handle diverse contact configurations. Second, our experiments focus on fixed-base, single-arm manipulation at the policy rate, leaving cluttered scenes, fast dynamic manipulation, mobile or bimanual settings, and high-bandwidth force-control objectives

for future work. Third, although the simulation twin supports controlled ablations and scalable evaluation, higher sim-to-real fidelity and reinforcement learning could enable larger-scale data generation, active sensing, and recovery beyond demonstrated behaviors.

Overall, FingerEye advances contact-rich dexterous manipulation by jointly designing a physical sensing interface and a learning interface: binocular fingertip vision and contact-induced deformation provide continuous local feedback, while real-and-sim data collection and group-structured fusion make that feedback effective for action prediction.

REFERENCES

- [1] Matthew T Mason. Toward robotic manipulation. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):1–28, 2018.
- [2] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.
- [3] Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.
- [4] Toru Lin, Kartik Sachdev, Linxi Fan, Jitendra Malik, and Yuke Zhu. Sim-to-real reinforcement learning for vision-based dexterous manipulation on humanoids. *arXiv preprint arXiv:2502.20396*, 2025.
- [5] Wenzhen Yuan, Siyuan Dong, and Edward H Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017.
- [6] Changyi Lin, Han Zhang, Jikai Xu, Lei Wu, and Huazhe Xu. 9dtact: A compact vision-based tactile sensor for accurate 3d shape reconstruction and generalizable 6d force estimation. *IEEE Robotics and Automation Letters*, 2023.
- [7] Toru Lin, Yu Zhang, Qiyang Li, Haozhi Qi, Brent Yi, Sergey Levine, and Jitendra Malik. Learning visuotactile skills with two multifingered hands. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5637–5643. IEEE, 2025.
- [8] Haozhi Qi, Brent Yi, Sudharshan Suresh, Mike Lambeta, Yi Ma, Roberto Calandra, and Jitendra Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023.
- [9] Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, et al. Neuralfeels with neural fields: Visuotactile perception for in-hand manipulation. *Science Robotics*, 9(96):eadl0628, 2024.
- [10] Akihiko Yamaguchi and Christopher G Atkeson. Implementing tactile behaviors using fingervision. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 241–248. IEEE, 2017.
- [11] Sheeraz Athar, Gaurav Patel, Zhengtong Xu, Qiang Qiu, and Yu She. Vistac toward a unified multimodal sensing finger for robotic manipulation. *IEEE Sensors Journal*, 23(20):25440–25450, 2023.
- [12] Francois R Hogan, Jean-François Tremblay, Bobak H Baghi, Michael Jenkin, Kaleem Siddiqi, and Gregory Dudek. Finger-sts: Combined proximity and tactile sensing for robotic manipulation. *IEEE Robotics and Automation Letters*, 7(4):10865–10872, 2022. doi: 10.1109/LRA.2022.3191812.
- [13] Etienne Roberge, Guillaume Fornes, and Jean-Philippe Roberge. Stereotac: A novel visuotactile sensor that combines tactile sensing with 3d vision. *IEEE Robotics and Automation Letters*, 8(10):6291–6298, 2023. doi: 10.1109/LRA.2023.3304560.
- [14] Patrick Lancaster, Pratik Gyawali, Christoforos Mavrogiannis, Siddhartha S Srinivasa, and Joshua R Smith. Optical proximity sensing for pose estimation during in-hand manipulation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11818–11825. IEEE, 2022.
- [15] Quan Khanh Luu, Dinh Quang Nguyen, Nhan Huu Nguyen, Nam Phuong Dam, and Van Anh Ho. Vision-based proximity and tactile sensing for robot arms: Design, perception, and control. *IEEE Transactions on Robotics*, 41:5000–5019, 2025. doi: 10.1109/TRO.2025.3593087.
- [16] Qi Wang, Yipai Du, and Michael Yu Wang. Spectac: A visual-tactile dual-modality sensor using uv illumination. In *2022 international conference on robotics and automation (ICRA)*, pages 10844–10850. IEEE, 2022.
- [17] Trevor Ablett, Oliver Limoyo, Adam Sigal, Affan Jilani, Jonathan Kelly, Kaleem Siddiqi, Francois Hogan, and Gregory Dudek. Multimodal and force-matched imitation learning with a see-through visuotactile sensor. *IEEE Transactions on Robotics*, 41:946–959, 2025. doi: 10.1109/TRO.2024.3521864.
- [18] Jikai Xu, Lei Wu, Changyi Lin, Ding Zhao, and Huazhe Xu. Dtactive: A vision-based tactile sensor with active surface. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 21664–21670. IEEE, 2025.
- [19] Yueshi Dong, Jieji Ren, Zhenle Liu, Zhanxuan Peng, Zihao Yuan, Ningbin Zhang, and Guoying Gu. Look-to-touch: a vision-enhanced proximity and tactile sensor for distance and geometry perception in robotic manipulation. *IEEE/ASME Transactions on Mechatronics*, 2026.
- [20] Akihiko Yamaguchi and Christopher G Atkeson. Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables. In *2016 IEEE-RAS 16th international conference on humanoid robots (humanoids)*, pages 1045–1051. IEEE, 2016.
- [21] Yuyang Li, Yinghan Chen, Zihang Zhao, Puhao Li, Tengyu Liu, Siyuan Huang, and Yixin Zhu. Simultaneous tactile-visual perception for learning multimodal robot manipulation. *IEEE Robotics and Automation Letters*, 2026.
- [22] Rui Ouyang and Robert Howe. Low-cost fiducial-based 6-axis force-torque sensor. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1653–1659. IEEE, 2020.
- [23] Jinxuan Zhu, Zihao Yan, Yangyu Xiao, Jingxiang Guo, Chenrui Tie, Xinyi Cao, Yuhang Zheng, and Lin Shao. Shapeforce: Low-cost soft robotic wrist for contact-rich manipulation. *arXiv preprint arXiv:2511.19955*, 2025.
- [24] Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. Leap hand: Low-cost, efficient, and anthropomor-

- phic hand for robot learning. *arXiv preprint arXiv:2309.06440*, 2023.
- [25] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems*, 2023.
 - [26] Eric T Chang, Peter Ballentine, Zhanpeng He, Do-Gon Kim, Kai Jiang, Hua-Hsuan Liang, Joaquin Palacios, William Wang, Pedro Piacenza, Ioannis Kyriassis, and Matei Ciocarlie. SpikeATac: A multimodal tactile finger with taxelized dynamic sensing for dexterous manipulation. *arXiv preprint arXiv:2510.27048*, 2025.
 - [27] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.
 - [28] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.
 - [29] Mike Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Am-radio: Agglomerative vision foundation model reduce all domains into one. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12490–12500, 2024.
 - [30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Appendix Contents

A	Additional Sensing Interface Design Details	8
A.1	Hardware Design Details	8
A.2	Fabrication Details and Bill of Materials	9
A.3	Multi-Tag Deformation Estimation	9
A.4	Comparison with GelSight and Keyline Tracking	10
A.5	Design Trade-offs	10
B	Sensing Interface Capability and Experiment Details	11
B.1	Wrench–Deformation Calibration Setup	11
B.2	Sensitivity Analysis Details	12
B.3	Delicate Grasping Setup and Hyperparameters	13
C	Learning Interface Details	13
C.1	Policy Architecture Details	13
D	Decoder Attention Diagnostics	15
E	Experiment Details	15
E.1	Default Policy-Training Settings	15
E.2	Task-Specific Training Settings	16
E.3	Post-Contact Tactile Map Baseline	16

A. Additional Sensing Interface Design Details

1) *Hardware Design Details*: This section expands the implementation details behind the sensing-interface design highlights in the main paper. Each FingerEye module integrates a binocular RGB camera system, a transparent acrylic plate with an AprilTag array, and a compliant silicone ring that converts contact into measurable plate-pose changes. The main paper shows the overall geometry and exploded view; here we specify the mechanical and optical choices that are shortened there.

Wedge-shaped fingertip module. The assembled module has a compact wedge-shaped form factor with an overall size of approximately $41.7 \times 30.0 \times 25.0$ mm. This geometry is chosen so the sensor can be mounted at dexterous fingertips while still entering narrow spaces, approaching object edges, and wedging under thin objects such as coins or letters. The wedge shape is therefore not only a packaging choice: it determines whether the fingertip can physically reach the contact configurations used in edge engagement and contact initiation tasks, as also illustrated by the GelSight wedging failure in Fig. 9.

Complementary binocular cameras. Each module uses two HBVCAM-2307-FPC82 V11 USB camera modules, each approximately $8.6 \times 8.6 \times 5.4$ mm with a 120° field of view. The tip camera is mounted approximately perpendicular to the acrylic plate at a short working distance of 10 mm, giving a sharp view of the AprilTag array and nearby contact surface. The root camera is mounted closer to the finger root and tilted by 21.48° relative to the fingertip horizontal direction, with a longer working distance of roughly 80 mm. Together, the two views cover different interaction phases: the tip camera emphasizes local deformation and close-range geometry, while the root camera preserves a wider scene view during approach and manipulation. Their spatially offset viewpoints also provide implicit depth cues that help disambiguate close-range object position and contact alignment.

Compliant peripheral ring. The soft ring is fabricated by silicone molding and mechanically bonded around the transparent acrylic cover. Its internal triangular elements provide directional compliance: the ring can deform under external forces and torques while preserving enough shape stability for repeatable pose estimation. Because the compliant region surrounds the plate rather than occupying the central visual path, frontal contact, lateral contact, and peripheral loading can all induce measurable acrylic-plate motion without blocking the RGB view. This is the mechanical basis for using plate-pose change as a contact-wrench proxy.

Acrylic cover and AprilTag array. A 1 mm-thick transparent acrylic plate protects the optical path and carries the fiducial pattern on its inner surface. The current tag layout uses 35 tags from the tag25h9 family arranged as a compact central cluster surrounded by a ring of tags for broad spatial coverage. Each tag is 2 mm wide and includes a 0.2 mm white border for reliable detection. This distributed layout is designed for partial visibility: when force, occlusion, or local deformation hides some tags, the remaining visible corners still constrain one global plate-pose estimate.

Mechanical assembly. The camera wrappers, sensor base, soft-ring molds, and supporting fixtures are 3D printed, while M2×4 screws fix the camera wrappers and help preserve optical alignment. The fully assembled module connects to a host PC through USB-C camera interfaces. The fabrication procedure and bill of materials are given in Sec. A.2.

2) *Fabrication Details and Bill of Materials:* These details support the main-paper claim that FingerEye can be built from low-cost off-the-shelf and 3D-printed components.

The fabrication process consists of five steps.

- 1) **3D printing.** Print the sensor base, camera wrappers, and mold components, including the lower mold, upper mold, and core insert.
- 2) **Silicone preparation and casting.** Mix the silicone base with curing agent at a mass ratio of base : curing agent = 100 : (1 ~ 3). Pour the mixture into the lower mold, assemble the upper mold and core insert, press the assembly onto the lower mold, and cure for more than 4 hours. After curing, remove the mold components and demold the soft silicone ring.
- 3) **Acrylic plate and tag preparation.** Apply the AprilTag sticker to the inner surface of the acrylic plate, bond the soft silicone ring to the sticker using silicone adhesive (JL-401), and let the adhesive fully cure.
- 4) **Camera assembly.** Secure the camera wrappers to the sensor base using M2×4 screws and install the camera modules into the wrappers.
- 5) **Wiring and connection.** Connect the assembled module to a host PC using a USB-C cable.

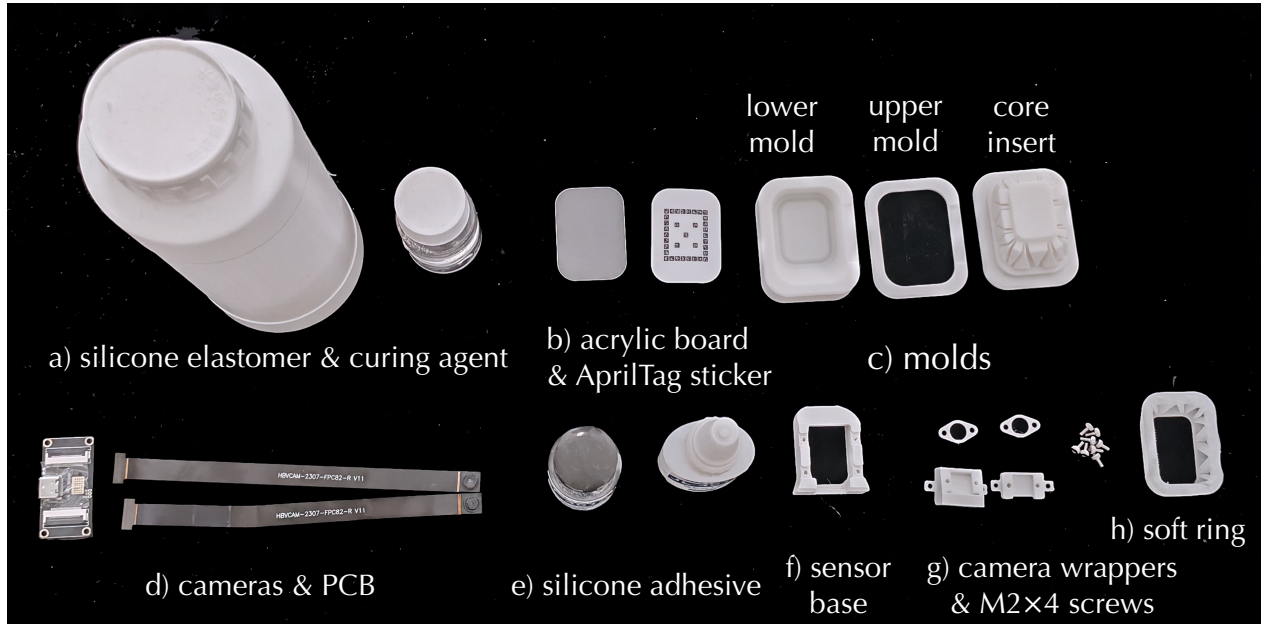


Fig. 8. **Materials used for FingerEye fabrication.** The module uses silicone elastomer and curing agent, camera modules, an acrylic plate with AprilTag sticker, silicone adhesive, and 3D-printed mold and fixture components.

TABLE III
BILL OF MATERIALS FOR ONE FINGEREYE MODULE.

Item	Cost (USD)
Acrylic plate	0.28
AprilTag sticker	0.42
Silicone elastomer & curing agent (20 g)	0.10
Silicone adhesive (JL-401, ~0.5 ml)	0.17
Soft silicone ring	0.02
Sensor base (3D-printed)	0.44
Camera wrappers (3D-printed)	0.45
Soft-ring mold (3D-printed)	1.78
M2×4 screws (8 pcs)	0.04
Camera modules (×2)	56.49
Total	60.19

3) *Multi-Tag Deformation Estimation:* This section gives the deformation estimator used by the multi-tag pose tracking design. The AprilTag array is rigidly attached to the acrylic plate, so the estimated plate pose provides a compact proxy for deformation of the surrounding compliant ring. For each detected tag, we extract four image corners and aggregate all visible

corners into one multi-tag correspondence set,

$$\{(\mathbf{X}_{i,j}^{\text{ref}}, \mathbf{u}_{i,j})\},$$

where $\mathbf{X}_{i,j}^{\text{ref}}$ is the 3D corner in the reference-tag frame and $\mathbf{u}_{i,j}$ is the observed image corner. We estimate the acrylic-plate pose by solving

$$\mathbf{T}_{\text{cam} \leftarrow \text{ref}} = \arg \min_{\mathbf{T} \in SE(3)} \sum_{i,j} \|\mathbf{u}_{i,j} - \pi(\mathbf{T}\mathbf{X}_{i,j}^{\text{ref}})\|_2^2, \quad (1)$$

using EPnP followed by Levenberg–Marquardt refinement. An alternative is to solve a pose for each visible tag independently and average the resulting tag poses. We found this averaging strategy less stable because each tag estimate has its own local corner noise, occlusion pattern, and degeneracy, so a single poorly localized tag can perturb the averaged pose. The joint multi-tag solve is more stable because all visible corners are optimized under one shared rigid-plate transform, making the estimate overdetermined and reducing jitter when individual tags are occluded, blurred, or locally distorted. During interaction, we subtract the no-contact reference pose from the current plate pose to obtain the six-dimensional deformation vector used in the wrench calibration in Sec. B.1.

4) *Comparison with GelSight and Keyline Tracking:* This section clarifies how the design differs from central-contact gel sensors and keyline-marker STS tracking.



Fig. 9. **GelSight failure in coin standing.** The square sensor geometry makes edge wedging difficult in this setup.

We attempted to use GelSight for the *coin standing* task, but encountered two practical issues during teleoperation data collection. First, the sensor did not provide a useful signal for table-edge interaction because the relevant contact occurred near the side of the fingertip rather than on the central gel surface. Second, the square sensor geometry made it difficult to wedge under the coin smoothly. In contrast, FingerEye uses a wedge-shaped fingertip with a peripheral soft ring, allowing the robot to engage the coin edge while still obtaining visual and deformation cues from the fingertip.

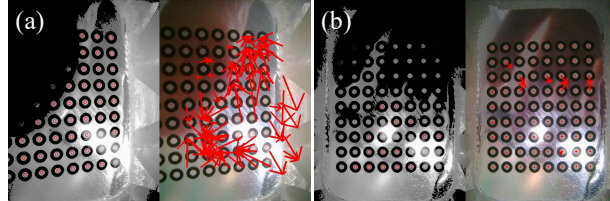


Fig. 10. **Qualitative comparison of marker-tracking pipelines.** Keyline tracking is brittle under deformation and lighting, while AprilTags preserve tag identity and multi-corner pose constraints.

We also compared the AprilTag-based pipeline with a keyline-marker pipeline inspired by TacThru [21]. The keyline pipeline detects circular markers with blob detection, then relies on nearest-neighbor association and temporal filtering. In our setup, it was less stable when deformation between frames was large or when ambient illumination varied, because both blob responses and marker association became brittle. By contrast, explicit tag identities and multi-corner constraints provide better conditioning for the unified PnP solve.

5) *Design Trade-offs: Complementary binocular views vs. a single compact camera.* A single fingertip camera would reduce wiring and calibration complexity, but it would force one optical configuration to cover both the close-range tag array and the wider pre-contact scene. FingerEye instead separates these roles across root and tip cameras. This increases hardware complexity slightly, but it provides the local coverage and close-range depth cues evaluated by the monocular-vs.-binocular ablations in the main paper.

Transparency–friction trade-off. The acrylic cover improves optical clarity and keeps the RGB stream stable across pre-contact and post-contact phases, but a rigid surface can reduce conformity and friction compared with soft gel interfaces. We mitigate this by placing high-friction gripping material on regions not visible to the cameras, such as perimeter regions and areas hidden behind tags, so that grasp reliability improves without blocking the visual sensing path.

Passive lighting vs. active self-illumination. We avoid active or colored self-illumination because the same RGB observations are used for both sensing and policy learning. Passive lighting keeps image appearance close to ordinary visual observations, while contrast enhancement improves AprilTag detection under low-light conditions without introducing illumination-specific color artifacts. The lighting-variation robustness result in the main paper further supports this choice: stable tag-array pose estimation does not require switching to active colored illumination.

Peripheral compliance for unobstructed contact sensing. Unlike designs that place a deformable gel in the central optical path, FingerEye keeps the central view clear and places compliance around the plate perimeter. This peripheral compliant interface allows frontal contact, edge engagement, and side loading to induce measurable plate-pose changes. The expanded deformation region is important for behaviors such as wedging, sliding, and reorientation, where useful contact often occurs near the fingertip edge.

Continuous RGB sensing instead of explicit contact-map reconstruction. FingerEye does not explicitly reconstruct dense contact shape or contact location from the real sensor stream. This trades explicit contact-map output for visual clarity, easy fabrication, continuous pre-contact perception, implicit depth from binocular RGB, and marker-tracked post-contact deformation in one sensing stream. The resulting output stays simple for policy learning while preserving the contact-sensitive cues needed by the evaluated tasks.

B. Sensing Interface Capability and Experiment Details

1) Wrench–Deformation Calibration Setup: This section gives the calibration protocol behind the main-paper claim that plate-pose change is a contact-wrench proxy.



Fig. 11. **Calibration setup.** A scale and 3D-printed fixtures provide axis-specific force and torque loading while the tag-array pose is recorded.

Let ΔL denote the six-dimensional pose change of the acrylic plate relative to a no-contact reference,

$$\Delta L = [\delta l_x, \delta l_y, \delta l_z, \delta \theta_x, \delta \theta_y, \delta \theta_z]^\top,$$

with translations in millimeters and rotations in radians. Let $\mathbf{F}$ denote the applied force and torque,

$$\mathbf{F} = [f_x, f_y, f_z, \tau_x, \tau_y, \tau_z]^\top.$$

We fit a mapping $g(\cdot)$ such that $\mathbf{F} = g(\Delta L)$. For force measurements, a digital scale measures the applied load while 3D-printed fixtures constrain the loading direction and contact location. For torque measurements, a force is applied at a known lever arm, and torque is computed from the measured force and lever-arm length. During each trial, forces or torques are applied along one axis at a time while the acrylic-plate pose is estimated from AprilTag tracking. We collect over 1,000 synchronized wrench–pose pairs spanning all six translational and rotational degrees of freedom. For each axis, 80% of the samples are used for fitting and 20% are held out for testing.

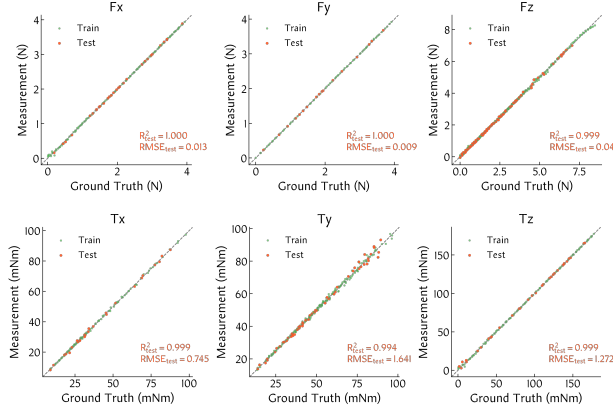


Fig. 12. **Wrench–deformation calibration.** Predicted forces and torques closely match held-out measurements.

Fig. 12 shows that the learned wrench–deformation mapping is close to linear and generalizes consistently across held-out samples. The three force components achieve $R^2_{\text{test}} \geq 0.999$, with held-out RMSE values of 0.013, 0.009, and 0.047 N for F_x , F_y , and F_z , respectively. The torque components also remain well predicted, with $R^2_{\text{test}} = 0.999$, 0.994, and 0.999 for τ_x , τ_y , and τ_z . These results support the use of acrylic-plate pose change as a compact contact-wrench proxy for the sensitivity analysis in Sec. B.2.

2) *Sensitivity Analysis Details:* This section derives the minimum detectable wrench reported in the main paper. Following fiducial-based sensitivity analysis in prior work [22, 23], we estimate the minimum detectable pose change of FingerEye from pixel-level localization accuracy and propagate it through the calibrated wrench–deformation model.

Subpixel localization assumption. We assume a subpixel localization accuracy of $d_R = 0.25$ px for AprilTag detection.

Translational sensitivity. Let w_{tag} denote the physical width of the AprilTag and w_{img} denote its observed image width. A displacement of d_R pixels corresponds to a minimum detectable translation

$$\Delta l_{\min} = \frac{w_{\text{tag}}}{w_{\text{img}}} d_R. \quad (2)$$

Using $w_{\text{tag}} = 2.0$ mm and $w_{\text{img}} = 37$ px gives

$$\Delta l_{\min} \approx 0.0135 \text{ mm}.$$

Rotational sensitivity. For a reference rotation angle θ , the corresponding pixel displacement at radius r , the half-width of the observed tag image patch, is given by the chord length $2r \sin(\theta/2)$. A d_R -pixel displacement therefore corresponds to a minimum detectable rotation

$$\Delta \theta_{\min} = \frac{\theta}{2r \sin(\theta/2)} d_R. \quad (3)$$

Using $\theta = \pi/12$ and $r = 18.5$ px gives

$$\Delta \theta_{\min} \approx 0.0136 \text{ rad}.$$

Pose sensitivity summary. The resulting minimum detectable 6-DoF pose change is

$$\Delta \mathbf{L}_{\min} = [0.0135, 0.0135, 0.0135, 0.0136, 0.0136, 0.0136]^T, \quad (4)$$

where the first three entries are in millimeters and the last three are in radians.

Propagation to wrench sensitivity. Propagating $\Delta \mathbf{L}_{\min}$ through the calibrated linear wrench–deformation model yields

$$\mathbf{F}_{\min} = [4.30, 4.22, 9.93, 0.32, 0.13, 8.55]^T, \quad (5)$$

where the first three entries correspond to force sensitivity in mN and the last three entries correspond to torque sensitivity in mNm.

3) *Delicate Grasping Setup and Hyperparameters*: This section gives the control details behind the contact-onset utility experiment in the main paper. Inspired by contact-sensitive grasping studies such as SpikeAtac [26], we evaluate whether FingerEye can trigger gentle stopping and lifting behavior on fragile or deformable objects. All grasping motions follow predefined joint-space trajectories generated by linear interpolation between an initial pose and a target pose. Joint commands are issued at a fixed control interval of 0.02 s per frame. To detect contact, we monitor the acrylic-plate normal displacement Δz estimated from AprilTag tracking. When Δz exceeds the object-specific threshold in Table IV, the approach motion stops and the robot lifts the object. Most objects are placed on a custom 3D-printed support platform so that thin or deformable objects can be contacted without scraping the table.

TABLE IV
HYPERPARAMETERS FOR DELICATE GRASPING EXPERIMENTS.

Object	Contact threshold (mm)	Total frames
0.7 mm pencil lead	0.02	15
Chip	0.10	30
Dessert cup	0.05	10
Seaweed	0.01	150
Grape	0.06	10
Cone	0.10	20
Eggshell	0.05	20
Wafer	0.05	5
Balloon	0.05	3

C. Learning Interface Details

1) *Policy Architecture Details*: This section gives the tensor-level policy architecture used for the current policy-interface comparisons. It specifies the temporal structure, tokenization, self-attention encoder variants, decoder cross-attention variants, and action projection used in the reported ablations. Fig. 13 visualizes representative encoder-decoder variants before the tensor-level specification below.

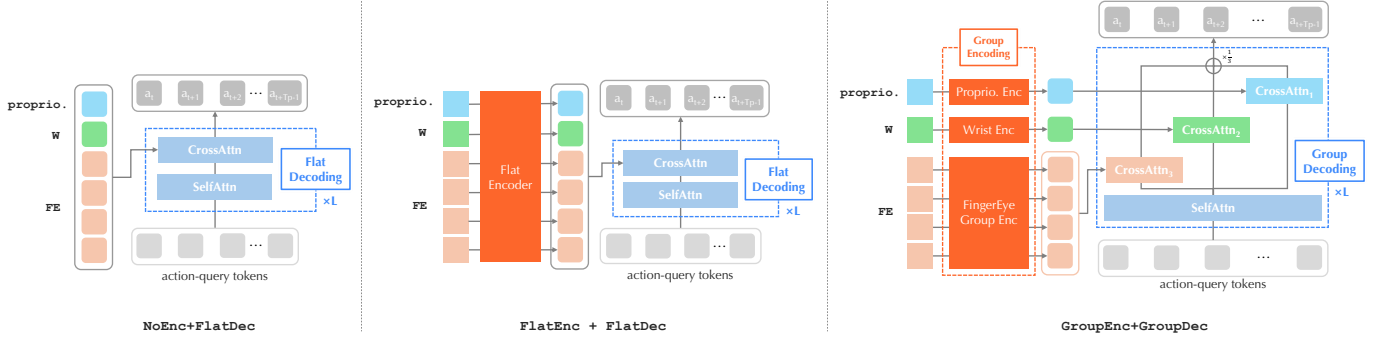


Fig. 13. **Policy architecture comparison.** The ablation varies how observation tokens are encoded and how action-query tokens attend to modality groups; the diagram uses the full **FlatEnc**, **GroupEnc**, **FlatDec**, and **GroupDec** labels. Left: **NoEnc+FlatDec** exposes tokenized observations directly to a flat decoder. Middle: **FlatEnc+FlatDec** applies flat self-attention across all observation tokens before flat decoding. Right: **GroupEnc+GroupDec** encodes proprioception, wrist, and FingerEye groups separately, then gives each group an independent decoder cross-attention path.

Receding-horizon action chunking. At each control step t , the policy conditions on the most recent $T_o = 1$ observation and predicts an action chunk of length T_p . The robot executes the first T_a actions and then replans in a receding-horizon manner, following action-chunking policies [27, 28]. The predicted action chunk is

$$\mathbf{A}_t = \pi_\theta(\mathbf{O}_t) = (\mathbf{a}_t, \mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+T_p-1}), \quad (6)$$

where each $\mathbf{a}_t \in \mathbb{R}^{d_a}$ lies in the task-specific action space listed in Table VII.

Observation groups. We write the observation as a tuple of modality groups

$$\mathbf{O}_t = (\mathbf{o}_t^1, \dots, \mathbf{o}_t^M), \quad (7)$$

where groups include wrist RGB, robot proprioception, FingerEye RGB, and optional auxiliary groups such as plate-pose histories or post-contact tactile-map tokens. All non-action inputs are represented as tokens before encoding; optional plate-pose histories are therefore treated as additional observation tokens, not as query modulation terms. RGB images are resized to 192×256 , and the proprioceptive state is projected from $\mathbf{J}_t \in \mathbb{R}^{d_s}$.

Visual tokenization with cached RADIO summaries. Let $i \in \{1, \dots, N_{\text{cam}}\}$ index the wrist and fingertip cameras. Each RGB image $I_t^{(i)}$ is passed independently through a frozen RADIO image backbone [29]. The backbone outputs one global summary vector

$$\mathbf{s}_t^{(i)} \in \mathbb{R}^{2304}. \quad (8)$$

For efficient training, these summary vectors are precomputed and cached offline. During policy training, each camera summary is projected to the policy dimension $d = 512$ by a camera-specific MLP

$$\phi_i : \mathbb{R}^{2304} \rightarrow \mathbb{R}^{512}, \quad \mathbf{v}_t^{(i)} = \phi_i(\mathbf{s}_t^{(i)}). \quad (9)$$

Projection MLPs are implemented as two-layer MLPs with GELU activations. We add a learnable camera-identity embedding $\mathbf{e}_{\text{cam}}^{(i)} \in \mathbb{R}^{512}$ to form the final visual token

$$\mathbf{u}_t^{(i)} = \mathbf{v}_t^{(i)} + \mathbf{e}_{\text{cam}}^{(i)}. \quad (10)$$

Vector observations, such as proprioception and optional plate-pose histories, are flattened and projected by group-specific MLPs into the same 512-dimensional token space. The FingerEye RGB group contains the projected tokens from the fingertip-mounted cameras, with camera-identity embeddings preserving finger and viewpoint identity. The resulting token sets are grouped as

$$\mathbf{U}_t^m = \tau_m(\boldsymbol{\sigma}_t^m), \quad \mathbf{U}_t = [\mathbf{U}_t^1, \dots, \mathbf{U}_t^M], \quad (11)$$

where τ_m denotes the tokenizer for modality group m .

Encoder-interface variants. The encoder ablation changes the self-attention range before decoding. **NoEnc** directly exposes tokenized observations to the decoder without pre-decoder self-attention. **FlatEnc** applies layers of Transformer self-attention [30] over all tokens together, so wrist, proprioception, FingerEye, and optional auxiliary tokens can attend to each other immediately. **GroupEnc** applies self-attention within each modality group only when the group contains at least two tokens; in particular, all FingerEye tokens can attend to other FingerEye tokens, but they do not attend to wrist or proprioceptive tokens until decoder fusion. Formally,

$$\begin{aligned} \mathbf{NoEnc} : \quad \mathbf{Z}_t^{\text{noenc}} &= \mathbf{U}_t, \\ \mathbf{FlatEnc} : \quad \mathbf{Z}_t^{\text{flat}} &= E_{\text{flat}}(\mathbf{U}_t), \\ \mathbf{GroupEnc} : \quad \mathbf{Z}_t^{\text{group}} &= [E_1(\mathbf{U}_t^1), \dots, E_M(\mathbf{U}_t^M)]. \end{aligned} \quad (12)$$

where E_{flat} is a stack of self-attention and feed-forward layers over all tokens, and E_m is a group-local stack when $|\mathbf{U}_t^m| \geq 2$ and the identity mapping when $|\mathbf{U}_t^m| < 2$. The default Transformer stacks use 4 layers, 16 attention heads, hidden dimension 512, and feed-forward dimension 2048; size-matched architecture ablations adjust hidden or feed-forward widths only when needed. For FingerEye, **GroupEnc** lets fingertip-local views integrate contact geometry before competing with global wrist or proprioceptive cues.

Action-query initialization. The decoder starts from clean learnable action-query embeddings:

$$\mathbf{h}_{t,k}^0 = \mathbf{q}_k, \quad \mathbf{q}_k \in \mathbb{R}^{512}, \quad k = 1, \dots, T_p. \quad (13)$$

Decoder-interface variants. Let $\mathbf{Z}_t = [\mathbf{z}_t^1, \dots, \mathbf{z}_t^M]$ denote the encoded sensory context grouped by modality. A decoder layer follows the standard Transformer structure: self-attention among action queries, cross-attention from action queries to sensory tokens, and a feed-forward update. The self-attention update is shared by both variants,

$$\bar{\mathbf{h}}_t^l = \mathbf{h}_t^l + \text{SelfAttn}^l(\mathbf{h}_t^l). \quad (14)$$

The flat and group-conditioned decoders differ in the cross-attention update:

$$\begin{aligned} \mathbf{FlatDec} : \quad \tilde{\mathbf{h}}_t^l &= \bar{\mathbf{h}}_t^l + \text{CrossAttn}^l(\bar{\mathbf{h}}_t^l, \mathbf{Z}_t), \\ \mathbf{GroupDec} : \quad \tilde{\mathbf{h}}_t^l &= \bar{\mathbf{h}}_t^l + \frac{1}{M} \sum_{m=1}^M \text{CrossAttn}_m^l(\bar{\mathbf{h}}_t^l, \mathbf{z}_t^m). \end{aligned} \quad (15)$$

After the cross-attention update, each decoder layer applies the standard Transformer feed-forward update,

$$\mathbf{h}_t^{l+1} = \tilde{\mathbf{h}}_t^l + \text{FFN}^l(\tilde{\mathbf{h}}_t^l). \quad (16)$$

FlatDec lets all modality tokens compete in one cross-attention normalization. **GroupDec** gives each modality group a separate cross-attention path and averages the residual updates, so every group has a direct route into the action queries. The decoded action features are mapped to joint commands by an MLP head,

$$\hat{\mathbf{a}}_{t+k-1} = f_{\text{act}}(\mathbf{h}_{t,k}^L), \quad k = 1, \dots, T_p. \quad (17)$$

Architecture-ablation notation. The architecture names in the main paper combine the encoder and decoder choices above. For example, **GroupEnc+GroupDec** uses group-local observation encoding followed by group-conditioned decoder cross-attention, while **FlatEnc+FlatDec** uses flat self-attention and flat cross-attention. Across these ablations, the tokenizer, action-query initialization, prediction horizon, action head, and training data are held fixed; when needed, hidden or feed-forward widths are adjusted to keep model sizes comparable.

D. Decoder Attention Diagnostics

To diagnose modality competition at the action interface, we inspect decoder cross-attention from action queries to context tokens in flat-decoder policies. We group context tokens into FingerEye (**FE**), **Wrist**, and **Proprio** groups, and aggregate attention mass over test-time rollouts on three representative simulation tasks and three evaluation seeds. We compare **NoEnc+FlatDec**, which directly exposes observation tokens to a flat decoder, with **FlatEnc+FlatDec**, an encoded-token flat-decoder baseline that first applies flat self-attention over observation tokens but still uses the same flat decoder readout.

TABLE V

DECODER CROSS-ATTENTION DIAGNOSTIC FOR FLAT DECODERS. WE AGGREGATE TEST-TIME ACTION-QUERY CROSS-ATTENTION OVER CONTEXT TOKENS AND MAP TOKENS TO FINGEREYE (**FE**), **WRIST**, AND **PROPRIO** GROUPS. PROP. WIN RATE DENOTES THE FRACTION OF ACTION-QUERY SAMPLES FOR WHICH THE **PROPRIO** GROUP RECEIVES THE LARGEST GROUP-LEVEL ATTENTION MASS. SR IS REPORTED FOR CONTEXT FROM THE CORRESPONDING MAIN-PAPER POLICY-INTERFACE EVALUATION; ATTENTION VALUES ARE COMPUTED FOR TRAIN SEED 42 AND AVERAGED OVER THREE EVALUATION SEEDS. ALL ENTRIES ARE PERCENTAGES.

Task	Policy	SR	FE mass	Wrist mass	Prop. mass	Prop. win rate
Coin	NoEnc+FlatDec	48.6±8.3	51.6±0.4	12.9±0.1	35.6±0.4	18.9±0.8
Coin	FlatEnc+FlatDec	43.6±8.3	46.0±0.1	10.5±0.0	43.4±0.1	37.6±0.4
Nut Pick	NoEnc+FlatDec	80.7±11.3	31.3±0.2	9.8±0.2	59.0±0.1	77.2±0.4
Nut Pick	FlatEnc+FlatDec	83.7±6.1	48.8±0.0	11.8±0.0	39.5±0.0	34.1±0.1
Peg-in-Hole	NoEnc+FlatDec	26.6±7.6	52.0±0.3	10.1±0.1	38.0±0.3	22.8±0.7
Peg-in-Hole	FlatEnc+FlatDec	18.2±7.5	42.2±0.0	7.8±0.0	50.0±0.1	61.0±1.1

Table V shows two complementary patterns in the flat-decoder action readout. First, the single **Proprio** token receives 35.6%–59.0% of the attention mass, far above the 16.7% mass it would receive under a token-uniform reference over four **FE** tokens, one **Wrist** token, and one **Proprio** token. This attention imbalance is consistent with decoder-level modality competition and a proprioceptive shortcut. Second, the variants with stronger overall SR in the corresponding policy-interface evaluation also show higher **FE** attention in this diagnostic, suggesting that preserving action-query access to **FE** observations is important for performance. On Coin, **NoEnc+FlatDec** has higher SR than **FlatEnc+FlatDec** (48.6% vs. 43.6%) and higher **FE** mass (51.6% vs. 46.0%). On Nut Pick, **FlatEnc+FlatDec** has higher SR (83.7% vs. 80.7%) and higher **FE** mass (48.8% vs. 31.3%). On Peg-in-Hole, **NoEnc+FlatDec** again has higher SR (26.6% vs. 18.2%) and higher **FE** mass (52.0% vs. 42.2%). Wrist attention remains low across all rows (7.8%–12.9%), indicating that the main flat-decoder competition is between distributed **FE** tokens and the **Proprio** token.

Although attention weights are a diagnostic rather than a causal measure of modality importance, the table suggests that the action readout interface matters: flat decoding can leave **FE** tokens and the low-dimensional proprioceptive token competing in one cross-attention distribution. Encoder-side processing can improve token representations, but it does not by itself guarantee that action queries preserve strong **FE** access during decoding. This motivates our group-conditioned decoder, which lets action queries read from each modality group separately before combining group-conditioned action features, avoiding a single softmax competition across all modality groups while still fusing **FE**, **Wrist**, and **Proprio** cues for action prediction.

E. Experiment Details

1) *Default Policy-Training Settings:* Unless otherwise stated, all policy experiments use the shared training defaults in Table VI. The number of training epochs and the prediction/execution horizon T_p/T_a vary by task and are listed separately in Table VII.

TABLE VI
DEFAULT POLICY-TRAINING SETTINGS.

Category	Default setting
Temporal context	$T_o = 1$ observation step; task-specific T_p/T_a in Table VII.
Policy backbone	4 encoder layers, 4 decoder layers, hidden dimension 512, 16 attention heads, feed-forward dimension 2048, GELU activation, dropout 0.0.
Modality fusion	Group encoding and group decoding enabled by default, corresponding to GroupEnc+GroupDec .
Optimizer	AdamW with learning rate 1.0×10^{-4} , $\beta = (0.95, 0.999)$, $\epsilon = 1.0 \times 10^{-8}$, and weight decay 1.0×10^{-6} .
Schedule and batch	Cosine learning-rate schedule, 500 warmup steps, batch size 256.
EMA and evaluation	Exponential moving average enabled with maximum decay 0.9999; evaluation uses the final checkpoint.

2) *Task-Specific Training Settings*: Table VII summarizes the task-specific policy settings used to track the experiment configuration. For each task, all modality ablations and policy-architecture ablations use the same demonstration dataset and the same task-specific train/test configuration split. Simulation training and testing initial states are sampled from the same randomized ranges. For real-world tasks, both training and testing randomize the object pose within each listed object, size, angle, or scale condition. For policy-architecture comparisons, we also size-match the model variants by adjusting hidden or feed-forward widths when needed, so that the comparison reflects the interface design rather than parameter count.

TABLE VII
TASK-SPECIFIC POLICY AND DATASET SETTINGS.

Setting	Task	Cams in use	Demos	Epochs	T_p/T_a	Action space	Train config.	Test config.
Simulation	Coin standing	4FE+1W	50	500	16/16	Abs. hand joints (8)	Randomized coin pose	Same randomized range
Simulation	Nut picking	4FE+1W	50	500	16/8	Abs. wrist position (3) + abs. hand joints (8)	Randomized nut/support + wrist init	Same randomized range
Simulation	Peg-in-Hole pushing	4FE+1W	200	300	16/16	Abs. hand joints (8)	Randomized peg/hole offsets	Same randomized range
Real-world	Coin standing	4FE+1W	60	500	16/8	Abs. hand joints (8)	Rand. pose; coin $D = 10/30/50$ mm	Rand. pose; coin $D = 20/40$ mm
Real-world	Chip picking	4FE+1W	30	500	16/8	Abs. hand joints (8) + rel. z (1)	Rand. pose; chip full/half/quarter	Rand. pose; chip full/irregular
Real-world	Letter retrieving	4FE+1W	60	500	16/8	Abs. hand joints (8) + arm joints (7)	Rand. pose; letter $A = 60^\circ/30^\circ$	Rand. pose; letter $A = 45^\circ$
Real-world	Syringe manipulation	6FE+1W	40	500	16/8	Abs. hand joints (12) + binary arm go-to pose command (1)	Rand. pose; syringe 10/30 mL	Rand. pose; syringe 20 mL

Note. Here, 4FE denotes four FingerEye camera views, i.e., two FingerEye modules with two cameras each; 6FE denotes three FingerEye modules with two cameras each. 1W denotes one wrist RGB camera.

3) *Post-Contact Tactile Map Baseline*: This section specifies the **W+T** simulation baseline used in Q3. The baseline tests whether post-contact tactile maps can replace continuous FingerEye observations under the same policy interface. Unlike FingerEye, which observes the scene throughout approach, alignment, and contact, **W+T** builds a tactile proxy from known scene geometry available only in simulation; it is not a PhysX contact-force sensor or a deformable tactile simulation.

For the index and thumb fingertips, we rasterize the local contact surface, treated as a tactile plane, into 64×96 grids. At each grid point g , the simulator computes the nearest distance d_g to nearby collision geometry and converts it to a contact-depth response,

$$c_g = \max(0, d_{\text{contact}} - d_g), \quad d_{\text{contact}} = 1 \text{ mm}. \quad (18)$$

Only geometry within 1 mm of the tactile plane produces a nonzero response, so the signal is intentionally silent before near contact. Each tactile map is encoded by the same frozen RADIO backbone used for RGB observations. The **W+T** policy then receives wrist RGB, proprioception, and the two per-finger tactile tokens through the same **GroupEnc+GroupDec** interface as **W+FE**.

W+T is intentionally a post-contact-only baseline. After contact, it provides local proximity and contact-geometry cues, but it does not provide the continuous fingertip RGB cues available to FingerEye during approach, alignment, and edge engagement. This is consistent with the main-text Q3 result: **W+T** remains close to wrist-only performance, while **W+FE** achieves much higher mean success by providing contact-centric observations throughout the full interaction.