RAPID: Reconfigurable, Adaptive Platform for Iterative Design

Zi Yin*, Fanhong Li*, Shurui Zheng, Jia Liu†
Tsinghua University
*Equal Contribution †Corresponding Author

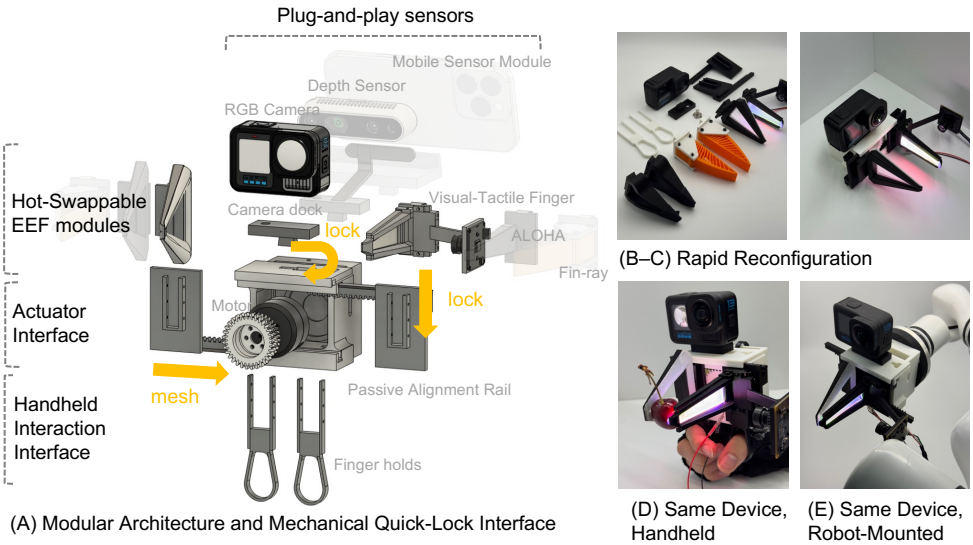

Fig. 1. RAPID system overview: a reconfigurable robotic platform enabling tool-free, plug-and-play integration of sensors, hot-swappable end-effectors, and modular actuation. The same modular device supports both handheld data collection and robot-mounted deployment.

***Abstract*—Developing robotic manipulation policies is iterative and hypothesis-driven: researchers test tactile sensing, gripper geometries, and sensor placements through real-world data collection and training. Yet even minor end-effector changes often require mechanical refitting and system re-integration, slowing iteration. We present RAPID, a full-stack reconfigurable platform designed to reduce this friction. RAPID is built around a tool-free, modular hardware architecture that unifies handheld data collection and robot deployment, and a matching software stack that maintains real-time awareness of the underlying hardware configuration through a driver-level *Physical Mask* derived from USB events. This modular hardware architecture reduces reconfiguration to seconds and makes systematic multi-modal ablation studies practical, allowing researchers to sweep diverse gripper and sensing configurations without repeated system bring-up. The Physical Mask exposes modality presence as an explicit runtime signal, enabling auto-configuration and graceful degradation under sensor hot-plug events, so policies can continue executing when sensors are physically added or removed. System-centric experiments show that RAPID reduces the setup time for multi-modal configurations by two orders of magnitude compared to traditional workflows and preserves policy execution under runtime sensor hot-unplug events. The hardware designs, drivers, and software stack are open-sourced at https://rapid-kit.github.io/.**

I. Introduction

Learning robotic manipulation policies from demonstrations is increasingly an iterative, hypothesis-driven process: researchers test whether tactile sensing, wider fingers, or alternative sensor placements improve performance [13, 53]. Meanwhile, the community has moved beyond visual-only imitation to multi-modal interfaces combining vision, proprioception, and touch, often utilizing portable handheld grippers for in-the-wild data collection [14, 46, 19, 51].

These handheld interfaces, from UMI and its variants [14, 52, 48, 12, 37, 20] to low-cost teleoperation systems [46, 19, 51], have significantly broadened where demonstrations can be collected; however, as sensing and embodiment options proliferate, every new gripper geometry, fingertip material, or sensor combination often means a new round of mechanical refitting and software integration before any data can be collected. Characterizing the benefit of tactile sensing across object geometries, for instance, might require testing N gripper designs against M sensor configurations; under a traditional workflow of screw-based end-effector swaps, manual configuration edits, and repeated restarts of the logging stack, systematically exploring this $N \times M$ space is prohibitively

expensive.

We argue that accelerating multi-modal manipulation research requires transitioning from static interfaces to *full-stack reconfigurable platforms*, which must solve two coupled problems: (i) the *mechanical friction* of swapping components, so that changing end-effectors, fingertips, or sensor modules becomes a seconds-level operation rather than minutes of bespoke rework; and (ii) the *modality observability gap*, namely the fact that the software stack has no principled way to know which sensing and actuation channels are physically present at a given moment, and therefore cannot safely adapt when sensors are added or removed.

We present RAPID (Reconfigurable, Adaptive Platform for Iterative Design), a full-stack platform designed to reduce this iteration gap. RAPID is built around two pillars: (1) a tool-free, modular hardware architecture that unifies handheld data collection and robot-mounted deployment, allowing researchers to sweep across diverse gripper and sensing configurations without repeated system bring-up; and (2) a matching software stack that maintains real-time awareness of the underlying hardware configuration through a driver-level *Physical Mask* derived from USB hot-plug events.

At the software level, the Physical Mask exposes modality presence as an explicit runtime signal: at any time step, the system knows which modalities are physically online, based on USB connect and disconnect events. This hardware-grounded signal complements training-time modality dropout [34, 27]: while dropout encourages robustness to missing inputs at the representation level, the Physical Mask provides the system-level observability needed to auto-configure logging and to gracefully degrade policy execution when sensors are physically added or removed.

We evaluate RAPID through a system-centric lens, focusing on iteration overhead and runtime robustness rather than task-specific success rates: we quantify time savings in a representative multi-modal ablation scenario, and probe runtime behavior by physically unplugging and re-plugging sensors during a tactile identification task. Our contributions are threefold:

- **A full-stack reconfigurable manipulation platform** that unifies handheld data collection and robot-mounted deployment through a tool-free, modular hardware architecture and a unified I/O stack.
- **A driver-level Physical Mask abstraction** that derives modality presence from USB events, enabling auto-configuration and graceful degradation under sensor hot-plug events.
- **A system-centric evaluation** demonstrating that RAPID reduces the setup time for multi-modal configurations by two orders of magnitude and preserves policy execution under runtime sensor hot-unplug.

II. Related Work

Portable manipulation interfaces. The Universal Manipulation Interface (UMI) [14] established handheld, vision-based data collection without requiring the target robot, and has spurred a family of specialized variants: FastUMI [52] (on-board visual-inertial odometry), DexUMI [48] (wearable hand exoskeletons), TacUMI [12] (force-torque and visuotactile sensing), ExUMI [49], MV-UMI [37], and UMI-on-Legs [20], alongside low-cost teleoperation systems such as GELLO [46] and Mobile ALOHA [19, 51]. While each variant addresses real limitations, they exist largely as *siloed designs* [5]: the same UMI concept has spawned at least five incompatible tracking approaches, and adding new modalities often requires complete system redesign. In contrast, RAPID is designed not as a single fixed interface but as a *full-stack reconfigurable platform* covering the spectrum of configurations needed for systematic multi-modal ablation studies, while keeping the iteration friction of moving between them low.

Multi-modal integration and runtime observability. Tactile sensing improves manipulation in contact-rich tasks [25, 50, 32, 40], but current architectures often treat sensor configurations as *static assumptions*: the software expects a fixed set of sensors and fails ungracefully if one is missing [3]. Modality dropout [34, 27] and its refinements [45, 18, 8], as well as multi-modal fusion architectures [11, 21, 43], provide *algorithmic* robustness but assume sensor malfunctions are already detected; runtime monitors [38, 1, 47] and graceful-degradation methods [42] largely assume valid data flow. RAPID targets this system-level *modality observability gap* with a driver-level Physical Mask: unlike timeout heuristics that infer absence from missing data, the mask is updated directly from USB connect and disconnect events, exposing modality presence as an explicit runtime signal that can drive auto-configuration and graceful degradation when sensors are hot-plugged.

Modular design and hardware iteration. Hardware–software co-design is well-motivated theoretically [41, 9, 10], yet is conducted almost exclusively in simulation because real-world iteration is prohibitively expensive [28, 22, 4] and the simulation-reality gap is non-monotonic with respect to morphology [39, 24, 2, 23]. Existing physical-evolution and modular systems [6, 35, 17, 33, 15, 29, 36, 31] still require significant reconfiguration time or are limited to specific actuators. RAPID adopts mortise-and-tenon joinery [16, 26, 44] paired with event-driven software discovery: our goal is not autonomous self-reconfiguration, but minimizing the *human overhead* of hardware iteration, shifting the bottleneck of multi-modal ablation studies from engineering debugging back to scientific inquiry.

III. System Design

RAPID is built around two pillars: a *tool-free, modular hardware architecture* (Section III-A) and a *driver-level Physical Mask* (Sections III-B–III-C). Fig. 2 illustrates the resulting layered architecture.

A. Hardware Layer

The hardware layer is designed to support frequent reconfiguration (Fig. 3). **Base and standardized slots.** The base is a 3D-printed chassis that integrates motor mounting

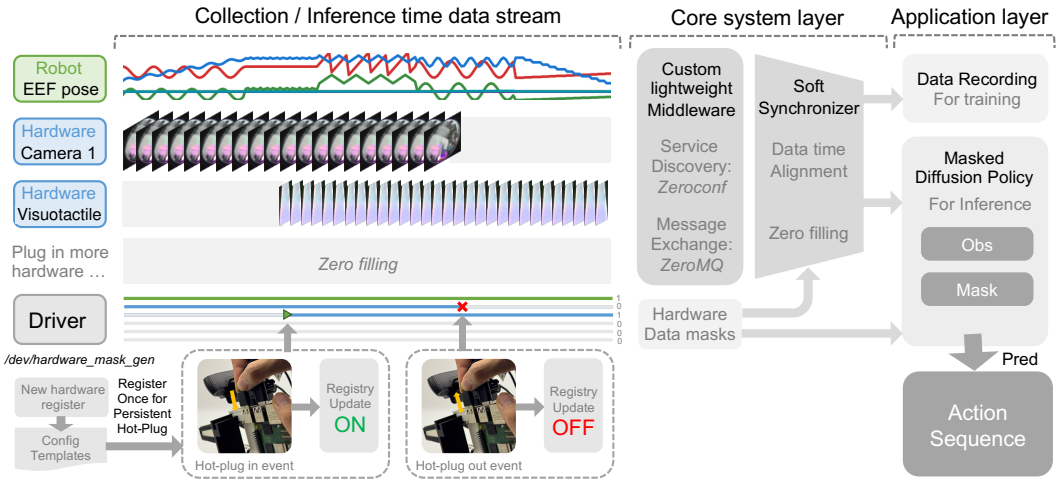

Fig. 2. Runtime architecture of the RAPID system. Top: asynchronous multimodal data streams during collection and inference. Bottom: the event-driven pathway from hardware hot-plug events to the driver-generated Physical Mask, which enables time alignment and zero filling of absent modalities, ensuring fixed-dimensional observations for data recording and mask-aware policy inference.

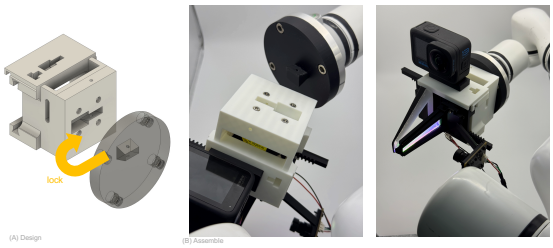

Fig. 3. Hardware layer details. (A) Tool-free connector design. (B) Rapid deployment to robot arm as end-effector.

and cable routing; it remains fixed across configurations. The base exposes four standardized attachment slots: front (finger bodies), tip (fingertips), top (wrist camera mounts), and bottom (handheld or robot-mounting adapters). Only modules change; the base does not. Swapping the bottom adapter switches the same base between handheld data collection and robot-mounted deployment. **Mechanical attachment.** Modules connect via mortise-and-tenon joints [51, 7], a geometry well-suited to FDM-printed PLA: it tolerates typical printing variation and survives repeated insertion cycles better than snap-fit alternatives. For extended operation, a single set screw can be added for additional retention. **Electrical interface.** We adopt USB as a unified interconnect layer, for its mature OS-level hot-plug support and wide ecosystem of converter boards. The system aggregates heterogeneous devices through a USB hub: USB2CAN (motors, robot arm), USB2Serial (ToF sensors), USB2LAN (LiDAR, GoPro), and native USB devices (RealSense, webcam, tactile sensors), as shown in Fig. 4 (Appendix A). Devices with direct LAN connections are integrated through a parallel registration mechanism. When a module is physically inserted or removed, the operating system generates a USB event that propagates to the driver layer.

B. Driver Layer

The driver layer translates USB events into runtime state information for the software stack. **Event detection and device registry.** A daemon process monitors USB plug/unplug events from the operating system; if the device matches a registered module, the corresponding data node is started or stopped. To add a new module type, users maintain a single configuration file, from which a registration script generates all derived artifacts (details in Appendix B and C).

Physical Mask Generator. The central contribution of this layer is a virtual device that reads the current registry state and exposes the **Physical Mask** at 500 Hz via a device file (e.g., `/dev/hardware_mask_gen`): a vector-valued signal mapping each modality name to a boolean indicating its current physical presence. Unlike data-timeout heuristics, this mask reflects hardware state directly: a modality is marked absent the moment its USB disconnect event fires, not after a timeout window (Fig. 2, bottom). This real-time, hardware-grounded signal is what enables downstream layers to distinguish transient packet loss from physical removal.

C. Middleware and Application Layer

Middleware. We implement a lightweight publish-subscribe layer based on ZeroMQ and Zeroconf, chosen over ROS2 [30] for ease of customization. The Physical Mask device file is published as a topic at this layer, alongside sensor data streams.

Synchronized Subscriber. A synchronized subscriber aggregates data topics and the mask topic, aligning them within a configurable time window (default 25 ms, similar to ROS ApproximateSync; Fig. 2, top). When the Physical Mask indicates a modality is offline, the subscriber zero-fills that channel, ensuring downstream consumers always receive fixed-dimension input.

Dual-mode operation. The application layer supports two modes sharing the same infrastructure. In *inference mode*,

TABLE I
RECONFIGURATION TIME: TRADITIONAL WORKFLOW VS. RAPID (N×M=9).

Design Choice	Traditional	Δ Saved	Ours
EEF swap	Screw (5±1 min)	−295 sec	Quick-release (5 sec)
Modality config	Manual YAML (2±0.5 min)	−120 sec	Auto-discovery (0 sec)
Collection system	Ctrl-C + restart (30±10 sec)	−30 sec	Hot-plug support (0 sec)
Sensor deployment	×2 (handheld ≠ exec)	−50%	×1 (unified)
Per-config total	∼480 sec	−475 sec	∼5 sec
Full ablation (N×M=9)	∼72 min	−71 min	∼45 sec

TABLE II
INFERENCE UNDER RUNTIME MODALITY CHANGES: IDENTIFICATION ACCURACY OVER 25 TRIALS, AND SYSTEM STATUS.

Condition	RAPID		Static Config	
	Score	Status	Score	Status
Full	0.92	Normal	0.92	Normal
No-tactile	0.52	Degraded	0.52	Degraded
Hot-unplug	0.56	Degraded	—	Crash
Hot-replug	0.84	Normal	—	Crash

at each control step the program fetches synchronized data with the current Physical Mask and feeds both to a mask-aware multimodal Diffusion Policy [13]: absent modalities are zero-filled to maintain fixed input dimensions, and the policy uses the mask to determine which channels carry valid observations, so sensor hot-plug does not crash the pipeline. In *collection mode*, the program displays real-time modality status, records synchronized data streams, and logs the Physical Mask per frame; since the mask updates at 500 Hz while data is recorded at 30–60 Hz, it captures transient dropouts at finer granularity than the data itself. Unlike dropout masks derived through post-processing, the Physical Mask is recorded as ground-truth hardware state at capture time.

IV. EVALUATION

Policy training. We use the standard CNN-based Diffusion Policy [13] with two observation modalities: a wrist camera image and a visuotactile fingertip image. During training, each modality is independently dropped with probability p=0.3 (with at least one modality retained), and the corresponding feature embedding is zeroed. At inference time, the Physical Mask replaces random dropout: physically absent sensors receive the same embedding zeroing, exactly matching the training-time condition.

A. Reconfiguration Efficiency

We evaluate time savings in a representative ablation scenario: a researcher iterating through N=3 gripper materials (rigid, Finray, silicone) × M=3 sensor modalities (vision-only, vision+motor, vision+tactile), i.e., 9 configurations.

Table I compares the time cost under a traditional workflow versus RAPID. The traditional workflow is defined as RAPID with each design choice removed: screw-based end-effector swaps instead of quick-release, manual YAML configuration editing instead of auto-discovery, full system restarts instead of hot-swap, and separate devices for handheld collection and robot execution instead of unified hardware.

In our prototype implementation and the configuration set used in this paper, RAPID reduces per-configuration setup time from approximately 480 s to 5 s; for the full N×M=9 ablation, this translates to on the order of 100× speedup.

B. Inference with Runtime Modality Change

We evaluate whether the Physical Mask mechanism enables graceful degradation during inference when sensors are physically added or removed (Fig. 5, Appendix D); this experiment stress-tests system behavior under runtime modality changes rather than benchmarking manipulation performance.

Task and metric. The gripper is equipped with a wrist camera and a visuotactile fingertip. Two visually identical blocks are presented, differing only in side-face surface texture (triangular vs. hexagonal), perceivable through tactile but not visual sensing. The policy must grasp each block and place it at the texture-specific target location. We report identification accuracy (correct placement / total trials) over 25 trials, and system status: Normal, Degraded (reduced performance), or Crash (pipeline failure).

Conditions. We compare RAPID against a baseline with the same hardware and policy but a static configuration file in place of the Physical Mask; changing the sensor set then requires manually editing this file and restarting the pipeline. We test four conditions: *Full* (wrist camera + tactile online throughout), *No-tactile* (tactile absent from the start), *Hot-unplug* (tactile physically removed mid-episode), and *Hot-replug* (removed and reinserted mid-episode). Table II reports results per condition.

V. DISCUSSION AND CONCLUSION

Trade-offs of tool-free modularity. A primary concern is the loss of mechanical rigidity relative to screw-based assemblies. Our mortise-and-tenon joints prioritize exchange speed and 3D printing tolerance over absolute stiffness: sufficient for handheld data collection and pick-and-place tasks, but potentially subject to deflection under high-load manipulation. A single retention screw can lock the joint for such regimes; systematic characterization of this trade-off is left to future work.

Limitations. USB hubs introduce bandwidth constraints with multiple high-resolution cameras, and our software synchronization lacks the microsecond-level precision required for high-speed dynamic manipulation. Our evaluation was conducted in a single laboratory setting, and we do not claim state-of-the-art manipulation performance. Newly inserted cameras still require manual extrinsic calibration, and the long-term durability of the PLA printed interfaces under thousands of insertion cycles remains to be quantified.

Conclusion. By mirroring mechanical modularity with software modality observability, RAPID reduces hardware reconfiguration time by two orders of magnitude and enables robust inference under physical sensor removal; the Physical Mask further enables dataset-level data curation (Appendix H).

References

[1] Christopher Agia, Rohan Sinha, Jingyun Yang, Zi-ang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. *arXiv preprint arXiv:2410.04640*, 2024.

[2] Elie Aljalbout, Jiaxu Xing, Angel Romero, Iretiayo Akinola, Caelan Reed Garrett, Eric Heiden, Abhishek Gupta, Tucker Hermans, Yashraj Narang, Dieter Fox, et al. The reality gap in robotics: Challenges, solutions, and best practices. *Annual Review of Control, Robotics, and Autonomous Systems*, 9, 2025.

[3] Michal Bednarek, Piotr Kicki, and Krzysztof Walas. On robustness of multi-modal fusion—robotics perspective. *Electronics*, 9(7):1152, 2020.

[4] Jagdeep Bhatia, Holly Jackson, Yunsheng Tian, Jie Xu, and Wojciech Matusik. Evolution gym: A large-scale benchmark for evolving soft robots. *Advances in Neural Information Processing Systems*, 34:2201–2214, 2021.

[5] Raunaq Mahesh Bhirangi. *Tactile sensing for Robot Learning: Development to Deployment*. PhD thesis, Carnegie Mellon University, 2024.

[6] Luzius Brodbeck, Simon Hauser, and Fumiya Iida. Morphological evolution of physical robots through model-free phenotype development. *PloS one*, 10(6):e0128444, 2015.

[7] Michael Burgess and Edward H Adelson. Grasp everything (get): 1-dof, 3-fingered gripper with tactile sensing for robust grasping. *arXiv preprint arXiv:2505.09771*, 2025.

[8] Jiahang Cao, Qiang Zhang, Hanzhong Guo, Jiaxu Wang, Hao Cheng, and Renjing Xu. Modality-composable diffusion policy via inference-time distribution-level composition. *arXiv preprint arXiv:2503.12466*, 2025.

[9] Tianjian Chen, Zhanpeng He, and Matei Ciocarlie. Hardware as policy: Mechanical and computational co-optimization using deep reinforcement learning. *arXiv preprint arXiv:2008.04460*, 2020.

[10] Tianjian Chen, Zhanpeng He, and Matei Ciocarlie. Co-designing hardware and control for robot hands. *Science Robotics*, 6(54):eabg2133, 2021.

[11] Yizhou Chen, Andrea Sipos, Mark Van der Merwe, and Nima Fazeli. Visuo-tactile transformers for manipulation. *arXiv preprint arXiv:2210.00121*, 2022.

[12] Tailai Cheng, Kejia Chen, Lingyun Chen, Liding Zhang, Yue Zhang, Yao Ling, Mahdi Hamad, Zhenshan Bing, Fan Wu, Karan Sharma, et al. Tacumi: A multi-modal universal manipulation interface for contact-rich tasks. *arXiv preprint arXiv:2601.14550*, 2026.

[13] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[14] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[15] HeeSun Choi, Cindy Crump, Christian Duriez, Asher Elmquist, Gregory Hager, David Han, Frank Hearl, Jessica Hodgins, Abhinandan Jain, Frederick Leve, et al. On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proceedings of the National Academy of Sciences*, 118(1): e1907856118, 2021.

[16] Alexandre Colle, Karen Donaldson, and Mauro Dragone. Co-design methodology for rapid prototyping of modular robots in care settings. *Frontiers in Robotics and AI*, 12: 1581506, 2025.

[17] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

[18] Yuqing Du, Daniel Ho, Alex Alemi, Eric Jang, and Mohi Khansari. Bayesian imitation learning for end-to-end mobile manipulation. In *International Conference on Machine Learning*, pages 5531–5546. PMLR, 2022.

[19] Zipeng Fu, Tony Z Zhao, and Chelsea Finn. Mobile ALOHA: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In *Conference on Robot Learning (CoRL)*, 2024.

[20] Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. *arXiv preprint arXiv:2407.10353*, 2024.

[21] Zihao He, Hongjie Fang, Jingjing Chen, Hao-Shu Fang, and Cewu Lu. Foar: Force-aware reactive policy for contact-rich robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

[22] David Howard, Agoston E Eiben, Danielle Frances Kennedy, Jean-Baptiste Mouret, Philip Valencia, and Dave Winkler. Evolving embodied intelligence from materials to machines. *Nature Machine Intelligence*, 1 (1):12–19, 2019.

[23] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. The transferability approach: Crossing the reality gap in evolutionary robotics. *IEEE Transactions on Evolutionary Computation*, 17(1):122–145, 2012.

[24] Sam Kriegman, Amir Mohammadi Nasab, Dylan Shah, Hannah Steele, Gabrielle Branin, Michael Levin, Josh Bongard, and Rebecca Kramer-Bottiglio. Scalable sim-to-real transfer of soft robot designs. *arXiv preprint arXiv:1911.10290*, 2019.

[25] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, et al. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5 (3):3838–3845, 2020.

[26] Maria Larsson, Hironori Yoshida, Nobuyuki Umetani, and Takeo Igarashi. Tsugite: Interactive design and fabrication of wood joints. In *UIST*, volume 20, pages 317–327, 2020.

[27] Guan-Horng Liu, Avinash Siravuru, Sai Prabhakar, Manuela Veloso, and George Kantor. Learning end-to-end multimodal sensor policies for autonomous navigation. In *Conference on robot learning*, pages 249–261. PMLR, 2017.

[28] Kevin Sebastian Luck, Heni Ben Amor, and Roberto Calandra. Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning. In *Conference on Robot Learning*, pages 854–869. PMLR, 2020.

[29] Raymond Ma and Aaron Dollar. Yale openhand project: Optimizing open-source hand designs for ease of fabrication and adoption. *IEEE Robotics & Automation Magazine*, 24(1):32–40, 2017.

[30] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66):eabm6074, 2022. doi: 10.1126/scirobotics.abm6074. URL https://www.science.org/doi/abs/10.1126/scirobotics.abm6074.

[31] MSS Malik, Nabila Naz, Mubashir Gulzar, and Soumya K Manna. A scalable pneumatic soft gripper with modular silicone fingers for adaptive grasping. *Journal of Robotics*, 2025(1):9148351, 2025.

[32] Andrew Melnik, Luca Lach, Matthias Plappert, Timo Korthals, Robert Haschke, and Helge Ritter. Using tactile sensing to improve the sample efficiency and performance of deep deterministic policy gradients for simulated in-hand manipulation tasks. *Frontiers in Robotics and AI*, 8:538773, 2021.

[33] Rodrigo Moreno and Andres Faiña. Emerge modular robot: A tool for fast deployment of evolved robots. *Frontiers in Robotics and AI*, 8:699814, 2021.

[34] Natalia Neverova, Christian Wolf, Graham Taylor, and Florian Nebout. Moddrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(8):1692–1706, 2015.

[35] Tønnes F Nygaard, Charles P Martin, Eivind Samuelsen, Jim Torresen, and Kyrre Glette. Real-world evolution adapts robot morphology and control to hardware limitations. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 125–132, 2018.

[36] Moises Pacheco, Rune Fogh, Henrik Hautop Lund, and David Johan Christensen. Fable ii: Design of a modular robot for creative learning. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 6134–6139. IEEE, 2015.

[37] Omar Rayyan, John Abanes, Mahmoud Hafez, Anthony Tzes, and Fares Abu-Dakka. Mv-umi: A scalable multi-view interface for cross-embodiment learning. *arXiv preprint arXiv:2509.18757*, 2025.

[38] Ralf Römer, Adrian Kobras, Luca Worbis, and Angela P Schoellig. Failure prediction at runtime for generative robot policies. *arXiv preprint arXiv:2510.09459*, 2025.

[39] Kent Rosser, Jia Kok, Javaan Chahl, and Josh Bongard. Sim2real gap is non-monotonic with robot complexity for morphology-in-the-loop flapping wing design. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7001–7007. IEEE, 2020.

[40] Siqi Shang, Mingyo Seo, Yuke Zhu, and Lillian Chin. Forte: Tactile force and slip sensing on compliant fingers for delicate manipulation. *arXiv preprint arXiv:2506.18960*, 2025.

[41] Karl Sims. Evolving virtual creatures. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 699–706. 2023.

[42] Taku Sugiyama, Kyo Kutsuzawa, Dai Owaki, Elijah Almanzor, Fumiya Iida, and Mitsuhiro Hayashibe. Versatile graceful degradation framework for bio-inspired proprioception with redundant soft sensors. *Frontiers in Robotics and AI*, 11:1504651, 2025.

[43] Yikai Tang, Haoran Geng, Sheng Zang, Pieter Abbeel, and Jitendra Malik. Visual-geometry diffusion policy: Robust generalization via complementarity-aware multimodal fusion. *arXiv preprint arXiv:2511.22445*, 2025.

[44] Tarik Tosun, Jay Davey, Chao Liu, and Mark Yim. Design and characterization of the ep-face connector. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 45–51. IEEE, 2016.

[45] Pankhuri Vanjani, Paul Mattes, Xiaogang Jia, Vedant Dave, and Rudolf Lioutikov. Disdp: Robust imitation learning via disentangled diffusion policies. In *Reinforcement Learning Conference*, 2025.

[46] Philipp Wu, Fred Shentu, Xingyu Lin, and Pieter Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition @ CoRL2023*, 2023.

[47] Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Miller, Robert Lee, Paarth Shah, Rares Ambrus, Haruki Nishimura, and Masha Itkina. Can we detect failures without failure data? uncertainty-aware runtime failure detection for imitation learning policies. *arXiv preprint arXiv:2503.08558*, 2025.

[48] Mengda Xu, Han Zhang, Yifan Hou, Zhenjia Xu, Linxi Fan, Manuela Veloso, and Shuran Song. Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation. *arXiv preprint arXiv:2505.21864*, 2025.

[49] Yue Xu, Litao Wei, Pengyu An, Qingyu Zhang, and Yong-Lu Li. exumi: Extensible robot teaching system with action-aware task-agnostic tactile representation. *arXiv preprint arXiv:2509.14688*, 2025.

[50] Wenzhen Yuan, Siyuan Dong, and Edward H Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017.

[51] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with

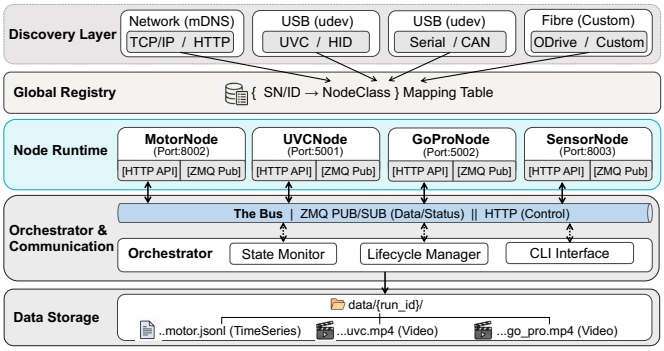

Fig. 4. System implementation of RAPID. The discovery layer supports heterogeneous device interfaces (USB-CAN, USB-Serial, USB-LAN, and native USB) through a unified hub. The driver layer performs event-driven device registration and generates the Physical Mask via a virtual device file. The middleware layer publishes sensor streams and the mask over a lightweight publish-subscribe transport (ZeroMQ + Zeroconf). The synchronisation layer aligns multimodal observations within a configurable time window and zero-fills absent channels. The application layer consumes fixed-dimension observations in either collection mode (logging with per-frame mask) or inference mode (mask-aware policy execution).

low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[52] Zhaxizhuom Zhaxizhuoma, Kehui Liu, Chuyue Guan, Zhongjie Jia, Ziniu Wu, Xin Liu, Tianyu Wang, Shuai Liang, Pengan CHEN, Pingrui Zhang, et al. Fastumi: A scalable and hardware-independent universal manipulation interface with dataset. In *Conference on Robot Learning*, pages 3069–3093. PMLR, 2025.

[53] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan 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.

Appendix

A. Physical Mask Data Format

The main text describes the Physical Mask as a "JSON dictionary." For clarity, we provide detailed specifications of both the actual runtime format and the debug format:

Binary Format (Runtime). The production system publishes the Physical Mask at 500 Hz via shared memory (`/dev/shm/rapid_hardware_mask`) for minimal latency. The binary layout is cache-line aligned (32 bytes):

TABLE III
Physical Mask binary format (32 bytes, cache-line aligned).

Offset	Size	Field	Description
0	4	magic	0x52415044 ("RAPD")
4	1	version	Protocol version (1)
5	1	device_count	Number of registered devices
6	2	_padding	Alignment padding
8	8	mask	Online status bitmask (u64)
16	8	timestamp_ns	Nanosecond timestamp
24	8	sequence	Monotonic sequence number

JSON Format (Debug). For debugging and visualization, a separate process outputs a human-readable JSON file at 1 Hz to `/dev/shm/rapid_hardware_mask.json`:

```
{
  "timestamp": "2026-02-05T10:30:00Z",
  "device_count": 3,
  "online_count": 2,
  "mask": "0x05",
  "mask_binary": "00000101",
  "sequence": 123456,
  "devices": [
    {"name": "cam_wrist",
     "bit": 0, "online": true},
    {"name": "tac_left",
     "bit": 1, "online": false},
    {"name": "motor_grip",
     "bit": 2, "online": true}
  ]
}
```

B. Device Discovery Pipeline

The discovery pipeline employs a two-tier matching strategy:

1) **Exact match:** VID + PID + Serial number uniquely identifies a specific device instance (e.g., a particular tactile sensor).
2) **Model match:** VID + PID identifies the device type when serial numbers are unavailable or for interchangeable devices.

Device registration is specified via TOML configuration files:

```
[device.tactile_left]
vid = "0x1234"
pid = "0x5678"
serial = "TACL001"
node = "tactile_publisher"
topic = "/rapid/tactile/left"
```

The registration script automatically generates udev rules (e.g., `/etc/udev/rules.d/99-rapid.rules`) to create stable symlinks such as `/dev/rapid/tactile_left`.

C. Driver Implementation

Stable device paths are established via udev symlinks (e.g., `/dev/rapid/tactile_left`), using USB serial numbers when available or port topology as fallback. An orchestrator monitors node health via periodic heartbeats; failed nodes are automatically restarted with backoff.

Single-point registration. To add a new module type, users maintain a single configuration file specifying device identifiers and data publishing logic. A registration script validates the configuration and generates all derived artifacts (udev rules, descriptor files, node entries). This keeps the user-facing configuration minimal while ensuring system-wide consistency.

D. System Latency Analysis

Table IV summarizes the end-to-end latency from hardware event to software response.

TABLE IV
SYSTEM LATENCY BREAKDOWN FOR DEVICE HOT-PLUG EVENTS.

Stage	Latency
USB insertion → OS udev event	50–100 ms
udev event → process startup	100–200 ms
Process startup → data stream ready	800 ms–1.5 s
End-to-end (USB → visualization)	**1.0–1.8 s**

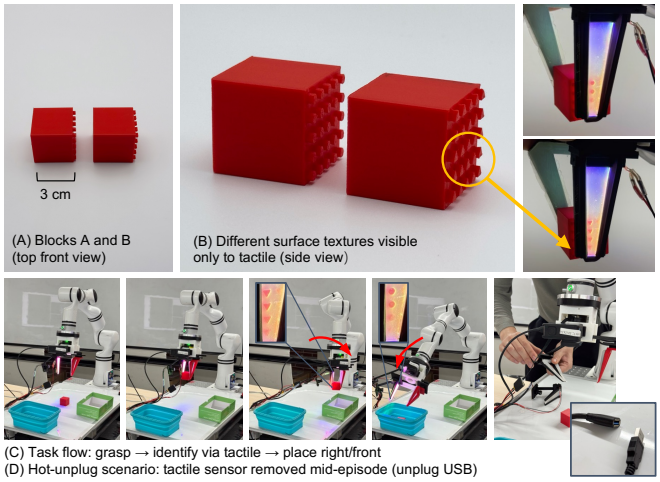

Fig. 5. Experiment task for evaluating runtime modality change. Blocks A and B appear identical to the wrist camera but differ in surface texture (triangular vs. hexagonal) perceivable through tactile sensing. The policy must grasp, identify, and place the triangular block to the right and the hexagonal block to the front. We test system behavior when the tactile sensor is unplugged during execution.

The "process startup to data stream ready" latency is device-dependent: USB cameras typically initialize within 800 ms, while network-connected devices (e.g., GoPro via USB-LAN adapter) may require up to 1.5 s. The Physical Mask itself is updated at 500 Hz (2 ms interval), ensuring that modality presence information is available with minimal delay after the OS-level event.

E. Dataset Statistics

TABLE V
DATASET STATISTICS FOR THE TACTILE IDENTIFICATION TASK.

Parameter	Value
Total episodes	72
Episodes per configuration	36
Frames per episode	∼100–170

The dataset comprises demonstrations collected with two block types (triangular and hexagonal texture), with 36 episodes per type. Episodes vary in length depending on grasp timing and placement trajectory.

TABLE VI
DIFFUSION POLICY HYPERPARAMETERS.

Parameter	Value
Architecture	CNN-based Diffusion Policy
Observation horizon	2
Action horizon	8
Prediction horizon	16
Training epochs	200
Modality dropout rate	0.3

F. Diffusion Policy Configuration

G. Ablation Study: Modality Configurations

Table VII presents ablation experiments investigating the contribution of different input modalities.

TABLE VII
ABLATION STUDY ON MODALITY CONFIGURATIONS.

#	Config	Data	Drop	Result
1	Vision only	mix	–	Biased (0.50)
2	Vision + mask	mix	0.3	Biased (0.50)
3	Vis + mask + diff	mix	0.3	**Success (1.00)**
4	Vision + diff	big	–	Biased (0.50)
5	Vision + diff	mix	–	**Success (1.00)**

Data: mix = mixture, big = bigcube. All diff image experiments use hand_view. "Biased" = policy places all objects to the same side; "Success" = correct texture-based placement.

Key findings:

- **Experiments 1–2:** Vision alone cannot discriminate objects that appear identical but differ in tactile texture.
- **Experiments 3, 5:** The diff image effectively captures tactile-induced visual changes, enabling successful texture identification.
- **Experiment 4 vs. 5:** Dataset diversity (mixture vs. single configuration) affects policy generalization; training on a single cube size leads to biased behavior.

H. Diff Image Computation

The diff image captures visual changes induced by tactile contact:

$$\text{diff} = \frac{\text{current_frame} - \text{reference_frame}}{2} + 0.5 \qquad (1)$$

where `reference_frame` is the first frame of the episode (before contact). The output is normalized to $[0, 1]$, with 0.5 representing no change. This encoding preserves both positive and negative intensity changes while maintaining a consistent input range for the policy network.

Beyond preventing runtime crashes, the Physical Mask has broader implications for data efficiency. In traditional pipelines, episodes with missing sensor data are often discarded entirely to maintain tensor shape consistency. RAPID enables the collection and utilization of heterogeneous datasets where episodes with varying sensor availability coexist. By logging the mask alongside the trajectory, researchers can train policies that explicitly learn robustness to sensor failure or

leverage expensive tactile sensors only for a subset of the data while using cheaper visual-only setups for the bulk. This capability transforms hardware failure from a fatal error into a manageable data attribute. In this sense, the Physical Mask is not only a runtime safety mechanism but also a dataset-level abstraction that supports more realistic system-centric evaluation of multimodal policies.

To verify that tactile images contain discriminative texture information, we trained a standalone classifier on tactile frames.

TABLE VIII
TACTILE TEXTURE CLASSIFICATION RESULTS.

Metric	Value
Architecture	ResNet-18 (ImageNet pretrained)
Dataset split	Episode-level (80/10/10)
Test accuracy	**99.73%**
Precision (triangular / hexagonal)	1.000 / 0.994
Recall (triangular / hexagonal)	0.995 / 1.000

Confusion Matrix:

	Pred: Triangular	**Pred: Hexagonal**
True: Triangular	586	3
True: Hexagonal	0	536

The near-perfect classification accuracy confirms that tactile images capture sufficient information to discriminate the two textures. This validates that the policy's inability to distinguish textures in vision-only conditions (Table VII, Experiments 1–2) stems from the visual similarity of the blocks rather than insufficient tactile information in the diff images.

Table IX lists the hardware modules used in this work.

TABLE IX
RAPID MODULE CATALOG.

Module	Slot	Sensors	Bring-up
Finger (parallel-jaw)	front	–	–
Finger (pinch)	front	–	–
Rigid tip (PLA)	tip	–	–
Soft tip (TPU)	tip	–	–
Tactile tip (GET)	tip	cam+LED	USB
Wrist mount (std)	top	GoPro	Net
Wrist mount (angled)	top	GoPro	Net
Handheld adapter	bottom	–	–
Robot adapter	bottom	–	–

Slots: front (finger bodies), tip (fingertips), top (wrist mounts), bottom (adapters). Sensor-bearing modules provide YAML descriptors for automatic discovery; mechanical-only modules require no software configuration.

Table X lists the specific hardware components used in our experimental setup.

TABLE X
HARDWARE COMPONENTS USED IN EXPERIMENTS.

Component	Model
Gripper motor	Dynamixel XL330
Wrist camera	Intel RealSense D405 / GoPro
Tactile sensor	GET (visuotactile)
USB Hub	4-port USB 3.0
Robot arm	Franka Emika Panda