

AutoDex: An Automated Real-World System for Dexterous Grasping Data Collection

Mingi Choi¹, Gunhee Kim¹, Jisoo Kim¹, Taeksoo Kim¹, Taeyun Ha¹, Jongbin Lim¹, Hanbyul Joo^{1,2}

¹Seoul National University ²RLWRLD

{willi19,gunhee2001,jlogkim,taeksu98,taeyun012,whdqls0534,hbjoo}@snu.ac.kr

Abstract: Learning robust dexterous grasping requires real-world data that records the physical outcomes of grasp attempts. Such data are difficult to obtain at scale: teleoperation provides real physical outcome labels but is slow and operator-biased, while simulation-based generation is cheap and scalable but cannot reliably certify real-world contact validity. A natural solution is to generate candidate grasps and verify them on real hardware, but this scales only if the entire collection loop—perception, execution, labeling, and reset—runs without human intervention. We present **AutoDex**, an automated real-world data-collection system that performs this entire validation cycle without human intervention. Given generated candidate grasps, it estimates the object’s initial 6D pose with dense 20-camera perception, executes selected candidates with collision-monitored robot motions, tracks the object during execution despite severe hand–object occlusion, labels lift-and-hold success or failure, and actively resets the object between trials to continue testing candidates across stable poses. The result is a reusable database of physically labeled grasp trials that downstream systems can retrieve and filter for feasibility. Using AutoDex, we collect 3,593 grasp trials with Allegro and Inspire hands across 100 diverse objects, together with synchronized multi-view observations and robot-state trajectories. In a matched 500-trajectory collection study, AutoDex requires 10.3 h compared with 49.4 h for teleoperation, yielding a 4.8 $\times$ throughput improvement. Grasps retrieved from the AutoDex-validated database achieve 76% real-world success, compared with 34% for grasps selected without real-world validation. Code and data will be released on our project page: <https://willi19.github.io/AutoDex/>.

Keywords: Dexterous Grasping, Autonomous Data Collection, Sim-to-Real

1 Introduction

Learning robust dexterous grasping requires data that records not only proposed hand–object configurations, but also their physical outcomes on real hardware [1, 2, 3, 4]. Unlike parallel-jaw grasps, multi-finger grasps involve high-dimensional hand configurations and multiple simultaneous contacts, so small errors in pose, friction, compliance, or force distribution can determine whether an apparently plausible grasp succeeds or slips [5]. As a result, geometric feasibility alone is not a reliable proxy for grasp validity: a candidate may be collision-free and kinematically feasible while still failing under real contact dynamics. Dexterous grasp datasets should therefore include both successful and failed grasp attempts with ground-truth physical outcome labels across diverse objects, poses, materials, and scene constraints.

Existing collection paradigms for such data trade off scale against physical validity. Human teleoperation [1, 2, 6] produces real contact outcomes, but it is slow, fatigue-limited, and biased by the operator’s choices over approach directions and finger configurations. In contrast, optimization- and simulation-based grasp synthesis methods [7, 8, 9] can generate large sets of candidate hand–object configurations at low cost. Many aspects of candidate validity are model-based and largely geomet-

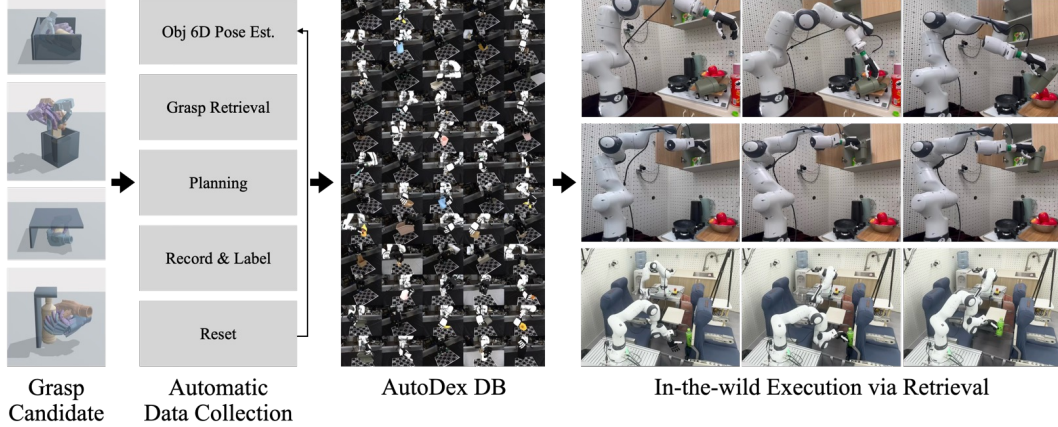


Figure 1: **The AutoDex pipeline.** AutoDex builds a database of physically labeled dexterous-grasp trials by executing generated candidates in a multi-camera workcell, labeling lift-and-hold success or failure, and resetting the object between trials. At deployment, downstream systems retrieve successful grasps, filter them for feasibility in the new scene, and execute the selected grasp.

ric: with accurate object meshes, robot calibration, and collision models, one can check joint-limit satisfaction, IK solvability, rigid-body collision avoidance, and motion-planning feasibility reasonably reliably. However, these checks only establish that a candidate is geometrically executable. They do not certify whether the resulting multi-finger contact will stably support the object. Real grasp success depends on contact physics—friction, slip, compliance, finger-pad deformation, actuator behavior, and the distribution of contact forces—which are difficult to simulate accurately. Thus, a candidate can be generator-valid and planner-executable, yet still fail during physical execution [10].

A natural solution is to combine scalable candidate synthesis and model-based feasibility checking with real-world verification: use synthesis and planning tools to propose executable candidate grasps, then test whether those candidates succeed under real physical interaction. However, hardware validation scales only if the full loop is automated. Any manual intervention in initial object pose estimation, execution-time tracking under hand-object occlusion, candidate selection, safety monitoring, success/failure labeling, or scene reset reintroduces the human bottleneck of teleoperation. The challenge is therefore not merely grasp generation, but end-to-end autonomous data collection.

To this end, we present AutoDex, an automated real-world data-collection system that turns generated grasp candidates into physically labeled robot trials. Given an object, AutoDex obtains candidate grasps from a modular candidate generator, filters and selects executable candidates, executes selected grasps on a physical multi-finger robot hand, records synchronized robot-state and multi-view visual data with success/failure labels, and resets the object for the next trial. AutoDex realizes this automation through four system components: (1) dense 20-camera perception for initial object pose estimation and execution-time tracking under hand-object occlusion, (2) collision-monitored motion execution for safe unattended grasp attempts, (3) physical success/failure labeling from lift-and-hold trials, and (4) active object reset for continued testing across stable poses. Together, these components enable fully automated real-world dexterous-grasp data collection at scale.

Using AutoDex, we collect 3,593 automatically labeled real-world grasp trials with Allegro and Inspire hands across 100 diverse objects. In a matched 500-trajectory collection study, AutoDex requires 10.3 h, whereas teleoperation requires 49.4 h, yielding a $4.8\times$ throughput improvement. Across scene types, grasps retrieved from the AutoDex-validated database achieve 76% real-world success, compared with 34% for grasps selected without real-world validation. Our contributions are threefold. First, we introduce AutoDex, an end-to-end automated system for real-world execution and labeling of generated dexterous-grasp candidates. Second, we collect a large real-world grasp-trial database containing both successes and failures, synchronized multi-view observations, and robot-state trajectories across 100 objects. Third, we provide a system-level evaluation of the full

collection loop, including throughput against teleoperation, real-world success against a model-screened baseline, autonomous reset across difficult pose transitions, and the effect of dense multi-view perception on collection reliability.

2 Related Work

Dexterous-grasp data: teleoperation and simulation synthesis. Existing real-world dexterous-grasp datasets are largely collected through human teleoperation [1, 2, 6, 11], where an operator controls a multi-finger hand through a glove, mocap, or vision interface. These datasets capture real contact outcomes on hardware, but throughput is bounded by sustained operator attention for a high-DoF hand, and the recorded distribution reflects the operator’s choices rather than an explicit specification over stable poses, scene constraints, or contact regions. A complementary line of work generates dexterous grasps in simulation through optimization or learned synthesis [7, 8, 9, 12, 13, 14, 15, 16, 17, 18, 4, 19, 20, 21]. These pipelines produce large candidate sets at low per-sample cost and can incorporate user-specified scene geometry, but the predicted validity of each candidate depends on the simulator’s contact model. Domain randomization [22, 23] can improve robustness to modeled variations, but it does not address effects that are absent or poorly represented in the simulator, such as finger-pad deformation, backlash, and micro-slip. AutoDex builds on these generators—we use BODex [8] in our implementation—and adds a real-world execution stage that supplies success/failure labels for contact outcomes that simulation cannot reliably predict.

Autonomous real-world data collection. Self-supervised and autonomous collection has scaled real-world grasping and manipulation data [24, 25, 26, 27, 28, 29, 30, 31], but largely for parallel-jaw grippers. Because the action space is high-dimensional, proposing useful multi-finger candidates directly through on-robot exploration is inefficient; practical systems often rely on a separate synthesis pipeline. Success labeling is also harder under multi-finger contact: the hand can occlude much of the object during execution, so reliable lift-and-hold labels require dense execution-time tracking rather than a sparse final observation. Finally, inter-trial reset cannot rely on passive settling alone, since dexterous grasp candidates are indexed by the object’s stable pose and collection must reorient the object to poses with remaining candidates. Recent autonomous pipelines that approach this direction still restrict themselves to parallel-jaw grippers to avoid the contact-modeling cost [32]. AutoDex addresses these challenges by combining generated multi-finger candidates with lift-and-hold execution under dense 20-camera tracking, success/failure labeling, and an explicit grasp-based reset module for pose-to-pose transitions between trials.

3 Method

AutoDex is an automated system for collecting real-world dexterous-grasp trials with physical outcome labels. Given an object and generated candidate grasps, AutoDex estimates the object pose, filters and selects executable candidates, executes the selected grasp on a physical multi-finger hand, labels the trial as success or failure, records synchronized robot and visual data, and resets the object to continue collection, all without human intervention.

3.1 Grasp Candidate Generation

AutoDex begins by obtaining candidate grasps from a modular candidate generator. The generator takes an object model and a scene specification as input and returns a set of candidate grasps, each represented by a wrist pose in the object frame and a hand configuration. The scene specification describes the manipulation environment, such as a tabletop with surrounding obstacles. The generator is responsible for proposing kinematically and geometrically feasible candidates, such as wrist poses and hand configurations that are collision-free with respect to the object and scene. It is not assumed to certify stability under real contact dynamics. AutoDex instead tests selected candidates through real-world execution and labels each trial by lift-and-hold success or failure.

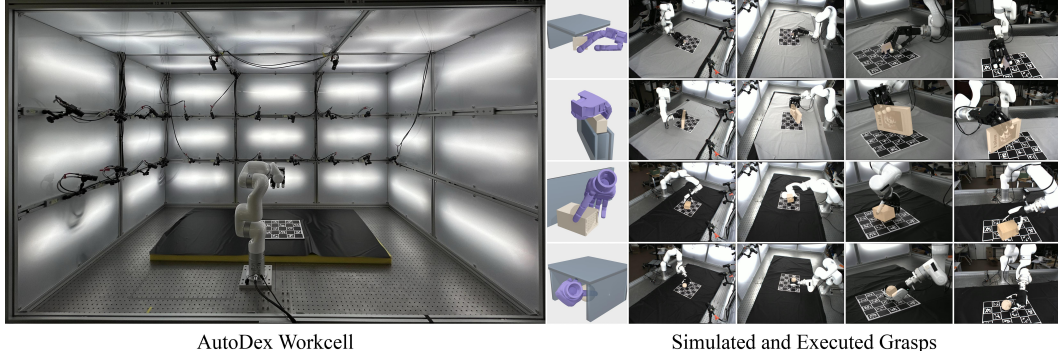


Figure 2: **AutoDex workcell and execution examples.** **Left:** A multi-camera workcell with a 6-DoF xArm, a swappable multi-finger hand, and 20 synchronized RGB cameras. **Middle and right:** Each row pairs a candidate grasp generated under a wall, shelf, or box scene constraint with its corresponding real-world execution, shown with synchronized views and overlaid tracked 6D object poses.

In this work, we use BODex [8] as the candidate generator and instantiate the scene specification with three planar constraint primitives: wall, shelf, and box. These primitives cover common tabletop constraints in which one or more approach directions are blocked, while excluding fully enclosed configurations that are unreachable by the hand. Let $\mathcal{P}$ denote the set of stable tabletop poses of the object. For each stable pose $P_i \in \mathcal{P}$, we aggregate candidates generated over scene primitives into a pose-indexed set $\mathcal{G}_{\text{cand}}(P_i)$. At runtime, AutoDex considers candidates for the current stable pose, invokes reset when no feasible unattempted candidates remain at that pose, and continues across stable poses until no further feasible candidates are available or the target dataset size is reached. The leftmost column in Fig. 1 shows representative candidates.

3.2 AutoDex Workcell

The AutoDex workcell is a calibrated robot-capture system designed to execute dexterous grasp trials without human intervention while recording synchronized visual and robot-state data. It consists of a 6-DoF xArm equipped with a swappable multi-finger hand, mounted inside a $1.5 \times 1.5 \times 2.5$ m LED-lit capture cell with 20 synchronized RGB cameras, as shown in Fig. 2. Each camera records at 30 FPS with 2048×1536 resolution and is synchronized by sub-millisecond hardware triggers. The dense camera arrangement provides redundant views for object pose estimation during grasp execution, when the hand can severely occlude the object. Camera intrinsics are calibrated before mounting using a ChArUco board, and per-session extrinsics are recovered with COLMAP and hand-eye calibration, yielding sub-millimeter multi-view pose self-consistency in our 20-camera setting. The robot controller operates at 120 Hz, streaming joint positions, velocities, torques, motor currents, and commanded targets from both the arm and hand. In our experiments, we use either a 16-DoF Allegro Hand or a 6-DoF Inspire Hand, attached through a standardized arm-hand mounting interface. A calibrated robot-camera timestamp offset aligns robot-state measurements with the multi-view RGB observations.

Since AutoDex runs without human intervention, the workcell requires an automatic safety mechanism for unexpected contacts during execution. The robot arm’s built-in collision detector was unreliable in our setting because grasped objects appear as persistent external loads and the mounted dexterous hand changes the end-effector inertia, both of which can trigger false positives. We therefore use a learned residual-torque monitor trained on collision-free motions of the deployed arm-hand assembly. Given joint position q , joint velocity $\dot{q}$, and motor-current-derived torque τ_{motor} , the monitor predicts the nominal free-space torque with an MLP and computes the residual:

$$\hat{\tau}_\theta = \text{MLP}_\theta(q, \dot{q}), \quad \tau_{\text{res}} = \hat{\tau}_\theta - \tau_{\text{motor}}.$$

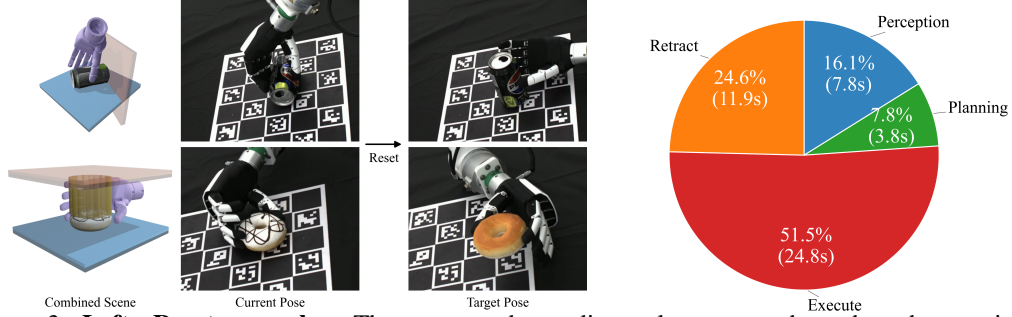


Figure 3: **Left: Reset examples.** The top row shows direct placement, where the robot carries the object to the target pose and releases it at the tabletop. The bottom row shows height-relaxed placement for a flat object, where virtual support pillars prevent finger intrusion into the object's descent region after release. In each row, the first panel shows the generated reset grasp, and the next two panels show the corresponding real-world pickup and placement motions. **Right: Runtime analysis.** Runtime composition across 500 autonomous trials, showing that physical robot motion dominates the cycle.

A sustained residual above a preset safety threshold is treated as unexpected contact. To avoid unnecessary aborts, we enable the monitor only during contact-critical segments, such as downward motions near the tabletop or surrounding scene. When such contact is detected, execution is halted, and the robot recovers by planning a trajectory back to a predefined home configuration while monotonically increasing the end-effector height. Together, dense multi-view perception and residual-torque safety monitoring provide precise object-state estimation and execution safeguards needed for autonomous dataset collection without human supervision.

3.3 Autonomous Data Collection Loop

We now describe how AutoDex uses the workcell to validate grasp candidates. At the start of each trial, AutoDex estimates the object's 6D pose from multi-view observations, identifies the current stable pose $P_i \in \mathcal{P}$, and transforms each object-relative candidate in $\mathcal{G}_{\text{cand}}(P_i)$ into the robot frame. We discard candidates whose wrist pose has no arm IK solution or whose arm or hand geometry collides with the tabletop or object. The remaining candidates form the feasible set for the current trial. AutoDex selects an unattempted feasible candidate and plans a collision-free arm trajectory that approaches the candidate wrist pose with an optimized open-hand configuration maximizing clearance from the object and tabletop. The hand then moves through the candidate pre-grasp and grasp configurations, after which the robot lifts the object and holds for a fixed duration. Each trial is labeled automatically from the captured multi-view stream. We run 6D object pose tracking across the stream and apply the lift-and-hold criterion: a trial is labeled successful if the tracked object remains at least 5 cm above its initial height throughout the 3 s hold phase, and as a failure otherwise.

After labeling, AutoDex saves the trial record, including the executed plan, multi-view recordings, tracked object poses, camera calibration, timing metadata, scene information, and the final success/failure label. The executed candidate is then marked as attempted in the collection state for the current stable pose. This feedback prevents repeated evaluation of the same candidate and makes candidate selection stateful: each subsequent trial selects from the remaining unattempted candidates that are feasible for the object's current stable pose.

Once the trial recording ends, the robot places the object back on the table. AutoDex then re-estimates the object's pose and identifies its current stable pose; if unattempted feasible candidates remain for that pose, AutoDex immediately starts the next trial. If no remaining candidate is feasible from the current pose, the reset module in Section 3.4 moves the object to another stable pose where additional candidates can be evaluated. In this way, AutoDex alternates between autonomous grasp

validation and autonomous reset until the candidate set is exhausted or the desired dataset size is reached.

3.4 Reset

The loop can continue only while unattempted candidate grasps remain feasible at the object’s current stable pose. When no such candidate exists, AutoDex selects another stable pose P_j with remaining candidates and reorients the object from the current pose P_i to P_j . To generate a reset grasp, AutoDex combines the pickup constraints at P_i and the placement constraints at P_j into a single collision scene (Fig. 3, left). A grasp that is collision-free in this combined scene is geometrically compatible with both endpoints, allowing the same hand–object grasp to be used throughout the reset. For many pose pairs, the robot can then directly place the object: it grasps the object at P_i , lifts and reorients it, places and releases it at P_j , and retracts, as shown in the top row of Fig. 3.

For flat objects such as plates or trays, however, satisfying the pickup and placement constraints simultaneously can leave too little free space for the hand, even when a valid pickup grasp exists. AutoDex therefore relaxes the placement endpoint by allowing the object to be released from a height h above the target pose, as shown in the bottom row of Fig. 3. This relaxation introduces an additional constraint: after release, the object should descend to the target pose without contacting the fingers, since such contact can perturb the final object pose. During reset-grasp generation, virtual support pillars are added as collision geometry beneath the released object to exclude grasps whose fingers intrude into this descent region. We first try $h = 0$, corresponding to placement at the tabletop, and increase h only when no collision-free reset grasp is found. Reset grasps that successfully complete the physical transition are cached for each stable-pose pair and reused when the same transition is required again.

3.5 AutoDex Database and Retrieval-Based In-the-Wild Execution

We use AutoDex to collect a database of 3,593 physically executed and automatically labeled dexterous-grasp trials across the Allegro and Inspire hands. The object set contains 100 household items, with over 80% sourced from IKEA and the remainder drawn from common household goods; the objects span diverse geometries and materials, including plastic, metal, wood, silicone, paper, tape, and ceramic. Unlike a database of generated grasp candidates, the AutoDex database stores physical executions with success and failure outcomes. This allows real-world validation to be performed offline during data collection, while downstream use reduces to retrieving successful grasps and checking their feasibility in a new scene.

In our deployment setting, a robot observes the workspace with four calibrated RGB cameras and estimates the target object’s 6D pose and nearby obstacle geometry. We use off-the-shelf tools to obtain object masks and an initial pose estimate [33, 34], refine the pose against multi-view silhouettes, and reconstruct surrounding geometry from depth estimates using Depth Anything 3 [35]. Given the object pose and obstacle scene, we retrieve the successful grasps for the target object and transform each stored grasp into the workspace frame. A grasp is accepted only if a collision-free arm trajectory exists that reaches the corresponding wrist pose without intersecting the surrounding obstacles. The first grasp that passes this feasibility check is selected for execution. This retrieval procedure enables in-the-wild execution without policy training or additional real-world trial collection. Fig. 1 shows an example.

4 Experiments

We evaluate AutoDex as an autonomous system for collecting physically labeled real-world dexterous-grasp data. Our evaluation focuses on four aspects: (1) throughput relative to teleoperation, (2) the effect of physical validation on the quality of a grasp database constructed from generator outputs, (3) the role of active reset in enabling efficient unattended collection across object poses, and (4) the effect of camera density on object-pose reliability during collection. For

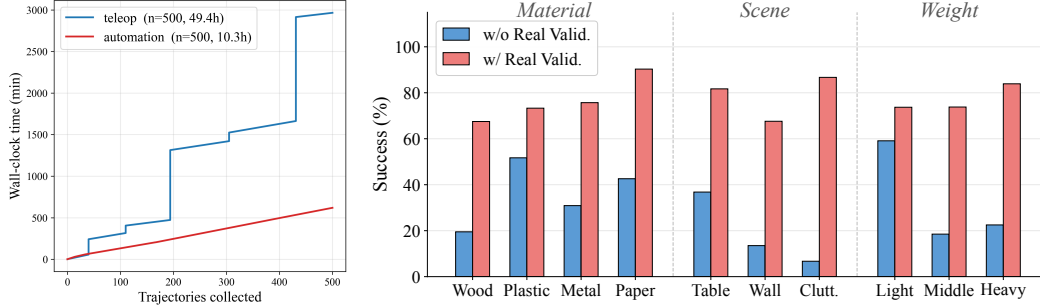


Figure 4: **Left: Throughput comparison.** AutoDex collects 500 trials in 10.3 h, compared with 49.4 h for teleoperation in the same workcell. **Right: Effect of physical validation.** Grasps retrieved from the AutoDex-validated database achieve 76% real-world success, compared with 34% for grasps retrieved from the model-screened database, across 20 objects and 515 trials. The improvement is consistent across material, scene, and weight categories.

evaluation, we use a representative 20-object subset drawn from our 100-object database. A grasp is considered successful if the object is lifted by 5 cm and held for 3 s.

4.1 Autonomous Data Collection Throughput and Bottlenecks

We compare the throughput of AutoDex with manual teleoperation in the same workcell. Figure 4 (left) shows cumulative trajectory accumulation over active collection time. AutoDex collects 500 trials in 10.3 h, whereas teleoperation requires 49.4 h for the same number of trials, yielding a $4.8\times$ improvement in effective throughput. Individual robot executions have comparable duration in both settings, but teleoperation requires continuous human supervision and accumulates long inactive intervals from operator fatigue and end-of-day stops. In contrast, AutoDex continues collecting autonomously once initialized, enabling sustained unattended accumulation across extended runs.

We further analyze the runtime composition of AutoDex’s autonomous collection loop in Fig. 3 (right). Across 500 fully executed trials, the mean grasp-execution loop duration is 48.2 s. Robot execution accounts for 51.5% of this loop (24.8 s), and retract accounts for 24.6% (11.9 s), while perception and motion planning account for 16.1% (7.8 s) and 7.8% (3.8 s), respectively. Because individual trials are dominated by physical robot motion, AutoDex does not improve throughput by substantially shortening each execution. Instead, its throughput gain comes from eliminating human idle time and enabling sustained unattended collection.

4.2 Impact of Physical Validation on Dataset Quality

We evaluate whether physical validation improves the quality of a grasp database constructed from simulation-generated candidates. Experiments are conducted on the physical robot across three scene types, tabletop, wall, and cluttered, using a 20-object subset with 515 total trials. We compare two databases constructed from the same generator output. The first is the model-screened database, which contains generated candidates screened only by the simulation sanity pre-filter, without real-world validation. The second is the AutoDex-validated database, which retains only candidates that were executed during autonomous collection and passed the lift-and-hold validity test. For each test object pose and scene, we apply the retrieval procedure in Section 3.5: candidates from the corresponding database are transformed into the workspace frame, filtered by collision-free arm planning, and executed on the robot.

Figure 4 (right) shows that the AutoDex-validated database achieves 76% grasp success, compared with 34% when using the model-screened database directly. The improvement holds across the evaluated material categories shown in Fig. 4, as well as across scene types and weight bins, indicating that the gain is not due to a single object category or scene condition. The gap is largest in cluttered scenes, where grasps retrieved from the model-screened database frequently fail under real contact

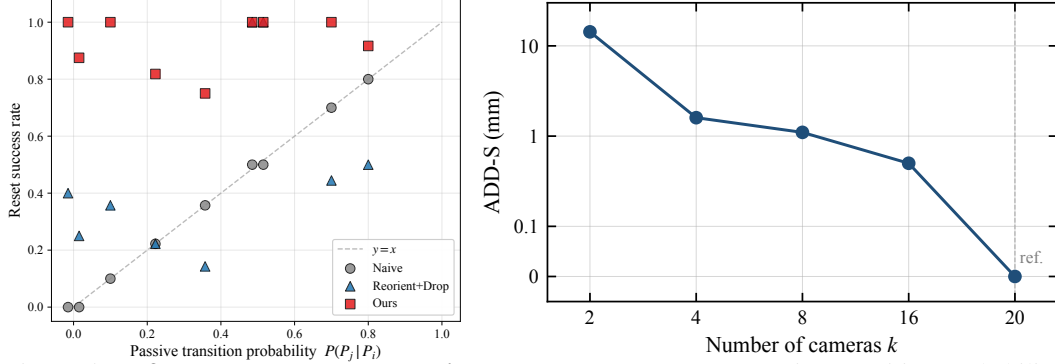


Figure 5: **Left: Reset strategy comparison.** Reset success versus passive transition probability $P(P_j | P_i)$. Naive Drop follows $y = x$ by construction, while Stable Reorient Placement maintains high success even for transitions rarely reached by passive settling. **Right: Pose self-consistency relative to the 20-camera reference as a function of camera count.** Mean ADD-S between the full 20-camera reference pose and k -camera subset poses, over 51 trials and 5 objects.

dynamics, while grasps retrieved from the AutoDex-validated database maintain 80-90% success. These results show that kinematic and collision feasibility alone are insufficient for constructing a reliable dexterous-grasp database. Physical validation substantially improves database quality by filtering out grasps that satisfy geometric constraints but fail under real-world contact effects such as slip, compliance, and multi-finger instability.

4.3 Reset Strategy Evaluation

We compare our reset method against two passive-drop baselines. *Naive Drop* lifts the object straight up and releases it, allowing it to settle under gravity. *Reorient + Passive Release* first reorients the object in the gripper toward a target pose P_j , then releases it above the table. *Stable Reorient Placement*, our method, places the object onto the table in the target pose before release, removing the dependence on passive settling.

Pose transitions are not equally difficult. Some target poses naturally appear after passive dropping, while others almost never occur without deliberate placement. We therefore evaluate reset success as a function of passive transition probability. We count a reset as successful when the object reaches the target stable pose P_j . For each object, we estimate this probability by repeatedly dropping it from tabletop pose P_i and recording the resulting pose P_j , yielding $P(P_j | P_i)$. Low values of $P(P_j | P_i)$ correspond to hard reset transitions, because passive settling rarely produces the desired target pose.

Figure 5 (left) reports reset success against $P(P_j | P_i)$. By construction, *Naive Drop* tracks $y = x$: its success rate equals the passive transition probability. *Reorient + Passive Release* improves over *Naive Drop* in some cases but remains close to the passive baseline because the final pose is still determined by uncontrolled settling after release. *Ours* maintains consistently high success across the full difficulty range, including transitions with $P(P_j | P_i) \approx 0$ that are essentially unreachable by passive strategies. *Reorient + Passive Release* also exhibits a failure mode incompatible with unattended operation: it ejects the object from the workspace in 5.3% of attempts, requiring human recovery, whereas our method causes no such failures. These results show that explicit reset placement is critical for reliable unattended dexterous-grasp collection.

4.4 Impact of Camera Density on Pose Reliability

We study how perception quality scales with the number of cameras used during silhouette refinement. For each capture, we treat the pose refined from all 20 cameras as the reference and re-refine the object pose using a random subset of k cameras. Figure 5 (right) reports the mean ADD-S [36] between the reference pose and the subset pose for $k \in \{2, 4, 8, 16\}$. Mean ADD-S decreases from

14.3 mm at $k = 2$ to 0.5 mm at $k = 16$, with the largest improvement occurring between $k = 2$ and $k = 4$. The high error at $k = 2$ is driven by occasional catastrophic pose failures on textureless or thin objects; these tail failures are largely eliminated by $k = 8$. Because candidate selection, collision checking, and grasp execution all depend on the estimated object pose, this result indicates that dense multi-view redundancy improves pose reliability for unattended collection.

5 Discussion and Limitations

We presented **AutoDex**, an automated system for collecting physically labeled dexterous-grasp trials through real-world execution, labeling, and reset. Our results show that real-world validation improves collection throughput over teleoperation and produces a grasp database that transfers more reliably than candidates screened without physical execution. AutoDex currently targets stable grasps for a fixed arm-hand setup. Bimanual coordination, mobile manipulation, finger-rolling regrasps, and functional grasps such as tool use and handover remain out of scope. The system also inherits the blind spots of its upstream generator and may miss grasps requiring dynamic finger motion during contact. Finally, while dense multi-view perception improves pose reliability, the highest-throughput setting still relies on a multi-camera workcell.

References

- [1] Y. Liu, Y. Yang, Y. Wang, X. Wu, J. Wang, Y. Yao, S. Schwertfeger, S. Yang, W. Wang, J. Yu, et al. Realdex: Towards human-like grasping for robotic dexterous hand. *arXiv preprint arXiv:2402.13853*, 2024.
- [2] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.
- [3] Z. Chen, K. Van Wyk, Y.-W. Chao, W. Yang, A. Mousavian, A. Gupta, and D. Fox. Dextransfer: Real world multi-fingered dexterous grasping with minimal human demonstrations. *arXiv preprint arXiv:2209.14284*, 2022.
- [4] J. Zhang, H. Liu, D. Li, X. Yu, H. Geng, Y. Ding, J. Chen, and H. Wang. DexGraspNet 2.0: Learning generative dexterous grasping in large-scale synthetic cluttered scenes. In *Conference on Robot Learning (CoRL)*, 2024.
- [5] A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 348–353, 2000.
- [6] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems*, 2023.
- [7] R. Wang, J. Zhang, J. Chen, Y. Xu, P. Li, T. Liu, and H. Wang. Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. *arXiv preprint arXiv:2210.02697*, 2022.
- [8] J. Chen, Y. Ke, and H. Wang. Bodex: Scalable and efficient robotic dexterous grasp synthesis using bilevel optimization. *arXiv preprint arXiv:2412.16490*, 2024.
- [9] D. Turpin, T. Zhong, S. Zhang, G. Zhu, J. Liu, R. Singh, E. Heiden, M. Macklin, S. Tsogkas, S. Dickinson, et al. Fast-grasp’d: Dexterous multi-finger grasp generation through differentiable simulation. *arXiv preprint arXiv:2306.08132*, 2023.
- [10] D. Huang, T. Zhang, Y. Li, L. Zhao, J. Li, Z. Fang, C. Xia, and X. He. Dexterous grasping with real-world robotic reinforcement learning. *arXiv preprint arXiv:2503.04014*, 2025.

- [11] Y. Park, J. S. Bhatia, L. Ankile, and P. Agrawal. Dart: Dexterous augmented reality teleoperation platform for large-scale robot data collection in simulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13883–13889. IEEE, 2025.
- [12] T. Liu, Z. Liu, Z. Jiao, Y. Zhu, and S.-C. Zhu. Synthesizing diverse and physically stable grasps with arbitrary hand structures using differentiable force closure estimator. *IEEE Robotics and Automation Letters*, 7(1):470–477, Jan. 2022. ISSN 2377-3774. doi:10.1109/lra.2021.3129138. URL <http://dx.doi.org/10.1109/LRA.2021.3129138>.
- [13] H. Zhang, S. Christen, Z. Fan, O. Hilliges, and J. Song. GraspXL: Generating grasping motions for diverse objects at scale. In *European Conference on Computer Vision (ECCV)*, 2024.
- [14] S. Chen, J. Bohg, and C. K. Liu. Springgrasp: Synthesizing compliant, dexterous grasps under shape uncertainty. *arXiv preprint arXiv:2404.13532*, 2024.
- [15] A. H. Li, P. Culbertson, J. W. Burdick, and A. D. Ames. Frogger: Fast robust grasp generation via the min-weight metric. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6809–6816. IEEE, 2023.
- [16] J. Lundell, F. Verdoja, and V. Kyrki. Ddgc: Generative deep dexterous grasping in clutter. *arXiv preprint arXiv:2103.04783*, 2021.
- [17] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4737–4746, 2023.
- [18] W. Wan, H. Geng, Y. Liu, Z. Shan, Y. Yang, L. Yi, and H. Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3891–3902, 2023.
- [19] J. Ye, K. Wang, C. Yuan, R. Yang, Y. Li, J. Zhu, Y. Qin, X. Zou, and X. Wang. Dex1b: Learning with 1b demonstrations for dexterous manipulation. In *Robotics: Science and Systems (RSS)*, 2025.
- [20] Z. Q. Chen, K. Van Wyk, Y.-W. Chao, W. Yang, A. Mousavian, A. Gupta, and D. Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. *arXiv preprint arXiv:2210.13638*, 2022.
- [21] S. Christen, M. Kocabas, E. Aksan, J. Hwangbo, J. Song, and O. Hilliges. D-grasp: Physically plausible dynamic grasp synthesis for hand-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20577–20586, 2022.
- [22] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.
- [23] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, et al. Solving rubik’s cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.
- [24] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research (IJRR)*, 37(4-5):421–436, 2018.
- [25] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2018.

- [26] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Scaling up multi-task robotic reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2021.
- [27] M. Ahn, D. Dwibedi, C. Finn, M. Arenas, K. Armstrong, V. Baruch, S. Belkhale, A. Brohan, N. Brown, K. Choromanski, et al. AutoRT: Embodied foundation models for large scale orchestration of robotic agents. *arXiv preprint arXiv:2401.12963*, 2024.
- [28] H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine. The ingredients of real-world robotic reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2020.
- [29] A. Sharma, A. M. Ahmed, R. Ahmad, and C. Finn. Self-improving robots: End-to-end autonomous visuomotor reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2023.
- [30] H. Liu, S. Nasiriany, L. Zhang, Z. Bao, and Y. Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. In *Robotics: Science and Systems (RSS)*, 2023.
- [31] S. Mirchandani, S. Belkhale, J. Hejna, E. Choi, M. S. Islam, and D. Sadigh. So you think you can scale up autonomous robot data collection? In *Conference on Robot Learning (CoRL)*, 2024.
- [32] J. Yu, L. Fu, H. Huang, K. El-Refai, R. A. Ambrus, R. Cheng, M. Z. Irshad, and K. Goldberg. Real2Render2Real: Scaling robot data without dynamics simulation or robot hardware. *arXiv preprint arXiv:2505.09601*, 2025.
- [33] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris, C. Ryali, K. V. Alwala, H. Khedr, A. Huang, et al. Sam 3: Segment anything with concepts. *arXiv:2511.16719*, 2025.
- [34] B. Wen, W. Yang, J. Kautz, and S. Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [35] H. Lin, S. Chen, J. Liew, D. Y. Chen, Z. Li, G. Shi, J. Feng, and B. Kang. Depth anything 3: Recovering the visual space from any views. *arXiv:2511.10647*, 2025.
- [36] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bratski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, 2012.
- [37] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi:10.1109/IROS.2012.6386109.
- [38] E. P. Örneke, Y. Labbé, B. Tekin, L. Ma, C. Keskin, C. Forster, and T. Hodaň. Foundpose: Unseen object pose estimation with foundation features. *European Conference on Computer Vision (ECCV)*, 2024.
- [39] V. N. Nguyen, C. Forster, B. Tekin, S. Shkodrani, V. Lepetit, C. Keskin, and T. Hodaň. Gotrack: Generic 6dof object pose refinement and tracking. *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2025.
- [40] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. Van Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. Ramos, et al. Curobo: Parallelized collision-free robot motion generation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8112–8119. IEEE, 2023.

Supplementary Material

A Candidate Generation and Execution Details

Stable tabletop poses. For each object, we precompute the set $\mathcal{P}$ of stable resting poses. For objects with registered axial symmetry, multiple simulated resting poses can be physically equivalent up to an in-plane rotation, so we canonicalize these duplicates. We transform the object’s registered symmetry axis by each candidate tabletop pose and compare it with the gravity direction. If the transformed axis is near vertical, i.e., its world z -component exceeds 0.95, the pose is rotationally symmetric about gravity and is kept as a single canonical pose. Otherwise, we instantiate five evenly spaced in-plane yaw variants, $\{0, 72, 144, 216, 288\}^\circ$. These yaw variants are an efficiency measure for collection: physically placing the object at several yaw angles increases scene coverage for each tabletop pose without requiring a separate correctness assumption.

Candidate generation per scene. For each stable pose and scene primitive (wall, shelf, box), we generate candidates with BODex [8]. Each primitive’s tightness is defined relative to the object rather than as an absolute scene size, and is swept to span a range of difficulty. The *wall* is a vertical plane placed at a gap of $\{2, 4, 6, 8\}$ cm from the object; the *shelf* encloses the object with the same gap to the side walls, back wall, and ceiling; and the *box* leaves only the top $\{5, 8\}$ cm of the object exposed above its rim as the graspable region, enclosing everything below. For each configuration, BODex is run with an escalating number of optimization seeds, $N \in \{200, 1000\}$, until at least five generator-valid candidates are found, so that easy scenes terminate cheaply while constrained scenes receive more optimization compute. Inter-trial reset grasps are generated through the same BODex interface, but in the combined pickup–placement collision scene that the reset module uses to carry the object between stable poses; there, the allowed release height above the target pose is increased from zero until a collision-free reset grasp exists.

Simulation sanity pre-filter. BODex enforces geometric feasibility but can still produce candidates that are obviously unstable even under a weak perturbation. Before committing real-robot time, we screen each candidate in MuJoCo [37]: with the object in contact, the hand is driven through pre-grasp $\rightarrow$ grasp $\rightarrow$ squeeze, and a deliberately weak external force of 0.1 N is applied along the gravity direction for 50 simulation steps. A candidate is rejected if the object translates by more than 5 cm under this perturbation. This filter is not intended to validate physical grasp success; it only removes candidates that fail a minimal plausibility check, leaving marginal cases to be evaluated by real-world execution.

Open pose, pre-grasp, grasp, and squeeze. The *pre-grasp* q_{pre} and *grasp* q_{grasp} hand configurations are produced directly by BODex and used as-is. When planning the approach, however, we command a separately computed *open pose* rather than q_{pre} , since a more-open hand provides greater obstacle clearance and more often admits a collision-free arm trajectory. Holding the wrist at the candidate grasp pose T_w and initializing from q_{pre} , we open the fingers by projected gradient descent to maximize clearance from the object and table,

$$q_{\text{open}} = \arg \max_{q \in [q_{\text{min}}, q_{\text{max}}]} \sum_f \min_{s \in \mathcal{S}_f} d_s(q; T_w),$$

where $\mathcal{S}_f$ is the set of collision spheres on finger f and $d_s(q; T_w)$ is the signed clearance from sphere s to the nearest obstacle. Optimizing the worst-clearance sphere for each finger prevents a fingertip from being sacrificed for a proximal-link gain. The hand approaches at q_{open} , closes to q_{pre} and then q_{grasp} , and finally over-closes to a *squeeze* that drives the fingers slightly past contact to resist slip, by linear extrapolation away from the pre-grasp,

$$q_{\text{sqz}} = c q_{\text{grasp}} - (c - 1) q_{\text{pre}},$$

with a per-hand gain $c = 6$ for the 16-DoF Allegro hand and $c = 2$ for the 6-DoF Inspire hand, clipped to the joint limits. The robot then lifts the object and holds for the fixed validation duration.

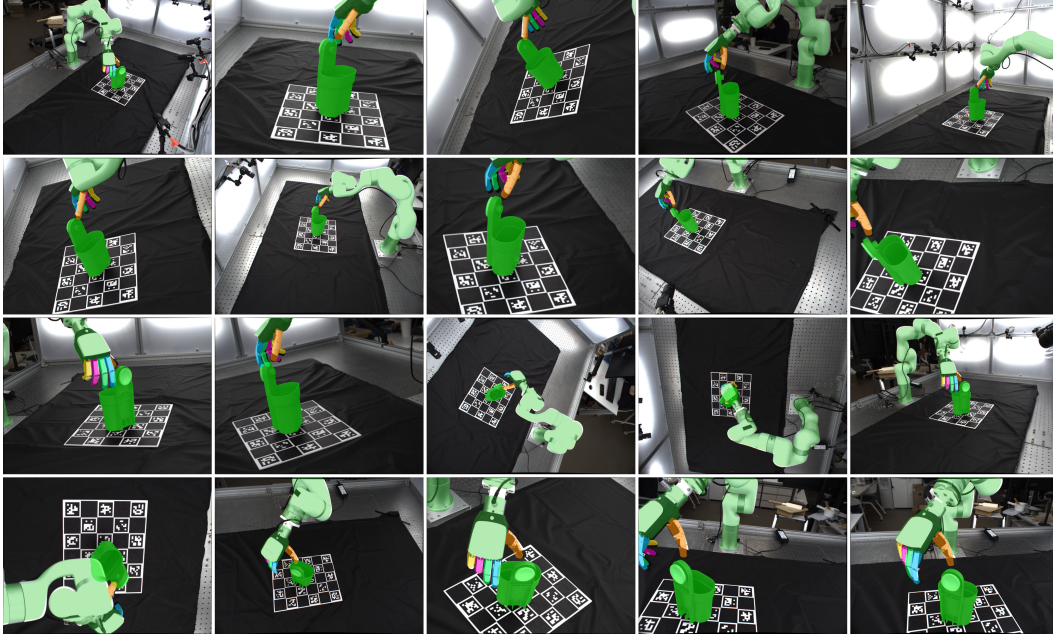
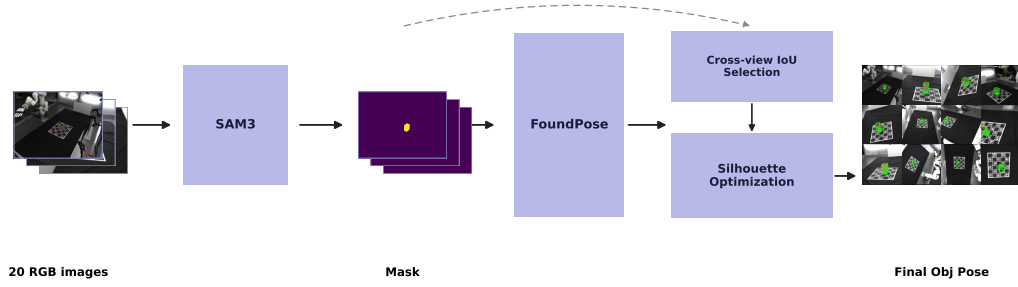


Figure 6: **End-to-end dataset alignment.** We reproject the calibrated robot mesh and the object mesh rendered at the estimated 6D pose into all 20 synchronized camera views. The visual overlap with the RGB observations provides an end-to-end check that camera extrinsics, hand-eye calibration, robot-state timing, and object-pose estimates are consistently aligned.

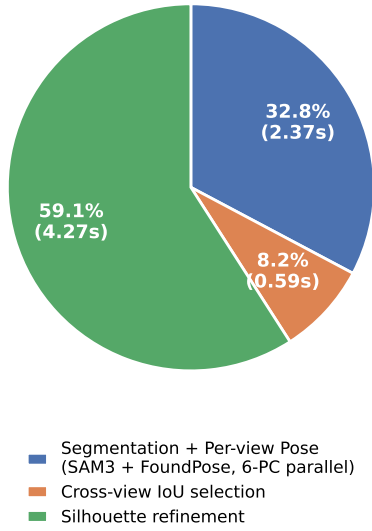
B Workcell, Calibration, and Object Perception

Calibration and synchronization. Per-camera intrinsics are calibrated before mounting by sweeping a ChArUco board across the full field of view, including the peripheral regions that become unreachable once the camera is fixed in the rig. Per-session extrinsics are recovered by global bundle adjustment (COLMAP) to a mean reprojection error of 0.2–0.5 px. A ChArUco target mounted on the end-effector is observed from multiple robot poses to solve the hand-eye calibration. Together, these calibrations provide the multi-view alignment used for object-pose estimation, which empirically yields sub-millimeter pose self-consistency in the 20-camera setting. For temporal alignment we measure the stable constant offset between camera timestamps and robot-state arrival times, subtract it, and linearly interpolate robot states to each camera exposure. To verify the resulting dataset alignment, we reproject both the calibrated robot mesh and the estimated object mesh into all synchronized camera views (Fig. 6); the overlap provides an end-to-end check of camera extrinsics, hand-eye calibration, robot-state timing, and object-pose estimation.

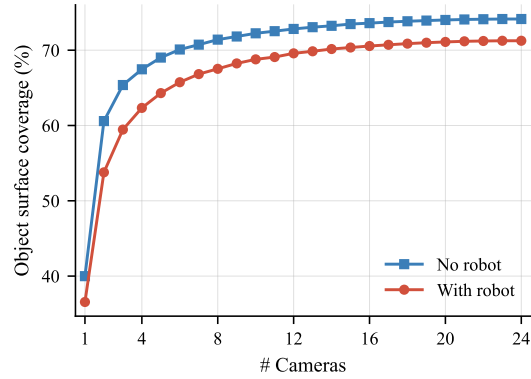
Object pose estimation and tracking. At the start of each trial, AutoDex estimates the object’s initial 6D pose from 20 RGB views using a multi-view pose pipeline (Fig. 7a). The dense camera array provides redundant observations for robust initial pose estimation and, during execution, helps recover object motion when the hand occludes much of the object. From the 20 RGB images, we predict object masks with SAM3 [33] and run FoundPose [38], an RGB-only foundation-feature pose estimator, on each view whose object mask exceeds an area threshold. Each pose hypothesis is rendered into all camera views and scored by the mean IoU between its rendered silhouette and the observed masks; the hypothesis with the best cross-view silhouette–mask agreement is selected. Views whose hypotheses disagree with this cross-view consensus are discarded as outliers and the pose is re-estimated from the remaining views, then refined by silhouette optimization against the same masks and used for grasp planning and execution. Success/failure labeling is automatic but performed post hoc from the recorded stream rather than in the real-time control loop. After each trial, we run multi-view GoTrack [39] on the recorded stream to estimate the object’s pose trajectory



(a) Multi-view object pose estimation pipeline.



(b) Perception time distribution.



(c) Object surface coverage vs. camera count.

Figure 7: **Multi-view object perception.** (a) Multi-view object pose estimation pipeline. (b) Run-time distribution across perception stages. (c) Visible object-surface coverage from the best k -camera subset of a larger 24-camera candidate rig, with and without robot occlusion. The final data-collection setup uses 20 cameras. Coverage saturates at around 8 cameras, while robot occlusion consistently reduces the visible surface fraction.

through the lift-and-hold phase, and apply the 5 cm-lift and 3 s-hold criterion to assign the success/failure label. This design avoids adding online tracking latency to robot execution while still producing automatic physical outcome labels. At deployment, the same localization pipeline runs on the sparser four-camera rig, trading multi-view redundancy for a lighter sensing footprint.

C Learned Residual-Torque Collision Detection

Overview. The main text summarizes the residual-torque monitor used for unattended execution; here we provide the implementation details and collection-time statistics. Because the xArm6 factory collision detector produced false aborts under grasped-object loads and with the mounted multi-finger hand, AutoDex learns a nominal free-space torque model for the deployed arm-hand assembly and detects contacts from the residual torque.

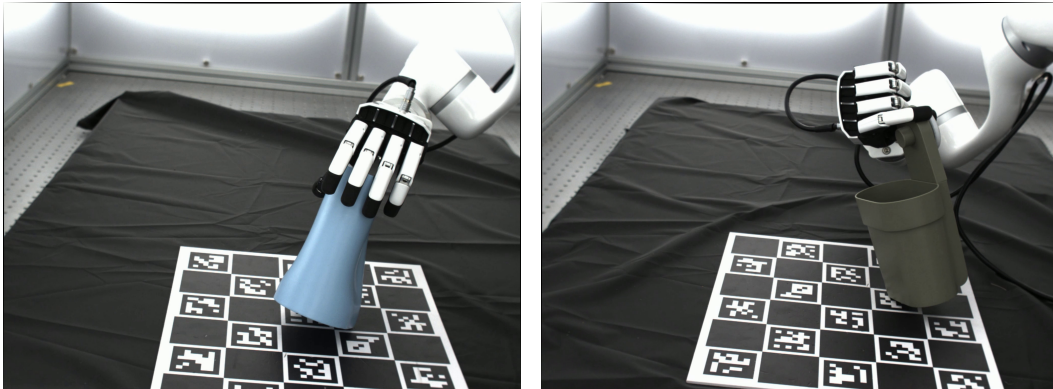


Figure 8: **Residual-torque contact detection examples.** Two placement trials in which the grasped object contacts the tabletop during descent. The residual-torque monitor detects the unexpected contact and halts the motion before continued descent can load the arm–hand assembly.

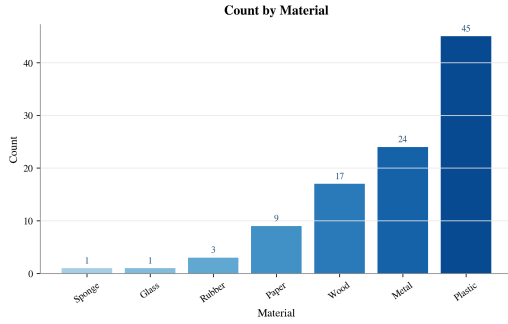
Training. We collect free-space $(q, \dot{q}, \tau_{\text{motor}})$ samples on the same arm–hand assembly used at deployment. All static and dynamic training trajectories are pre-screened with cuRobo [40] to avoid workspace, floor, and self-collisions. The static pass samples settled configurations to capture gravity loads, while the dynamic pass executes the screened trajectories in servo mode to capture velocity-dependent friction and other repeatable dynamic torque biases. The nominal model is a small MLP (under 100 KB) trained with a robust loss on smoothed joint velocities and current-derived motor torques.

Runtime detection and recovery. The robot controller streams state at 120 Hz, and the residual monitor runs in the servo loop at 100 Hz, evaluating the nominal-torque model in $\sim 30 \mu\text{s}$ on CPU and low-pass filtering the residual. Before each monitored motion segment, AutoDex records a short static baseline to absorb slow torque-calibration drift. A contact is declared when the baseline-subtracted residual on the watched shoulder joints, J_1 and J_2 , exceeds $30 \text{ N}\cdot\text{m}$ for a sustained window of 50 samples ($\sim 0.5 \text{ s}$). We watch these proximal joints because their torque is the most sensitive to vertical-axis loading and the least confounded by hand-orientation changes. Monitoring is enabled only during motions with a downward Cartesian component, where unexpected contact with the object or tabletop is most damaging. When the monitor fires, the active primitive aborts and the planner generates a recovery trajectory back to the home configuration with monotonically non-decreasing end-effector height, so the arm disengages upward before any lateral motion. We do not enable the monitor during lift motions, because the grasped object introduces a persistent external load that would be indistinguishable from contact under the pre-lift baseline.

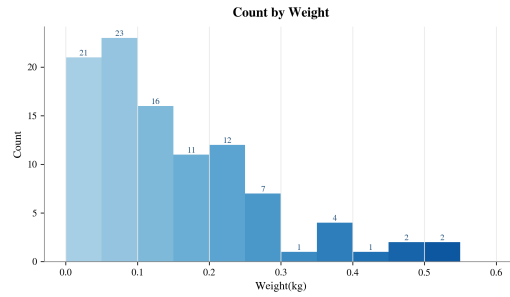
Collisions caught during collection. This safeguard was necessary in practice. In an earlier 1,870-trial run without the monitor, before the placement phase was introduced, two collisions damaged the arm–hand mounting adapter. After enabling the monitor, no further adapter damage occurred in our collection runs, even though the placement phase added further contact-prone motions. The monitor halted the arm during downward approach in 2.7% of all trials and during placement in 8.9% of all trials (Fig. 8). Visible contacts accounted for 0.3% and 3.1% of all trials, respectively; the remaining monitor triggers were conservative aborts without visible contact. Approach contacts typically occurred when the object shifted between perception and open-loop execution, for example due to a cable tugging it out of place or imperfect tracking of the planned hand trajectory, while placement contacts occurred when the grasped object touched the tabletop earlier than expected during descent. This conservative bias is desirable for unattended operation: an abort only skips or retries a trial, whereas a missed collision can damage the robot or hand.



(a) Object library (100 objects).



(b) Material distribution.



(c) Weight distribution.

Figure 9: **AutoDex object library and diversity.** (a) The 100-object library spans diverse geometries, materials, and functional categories from everyday household items. (b, c) The objects cover seven dominant material categories and a wide weight range.

D Object Library

The dataset spans 100 diverse everyday objects (Fig. 9a), with more than 80% sourced from IKEA household products for commercial availability and reproducibility. The objects cover a broad range of geometries and seven dominant material categories—plastic, metal, wood, silicone, paper, tape, and ceramic (Fig. 9b)—and a wide weight range from lightweight containers (< 50 g) to heavy kitchen tools (> 500 g) (Fig. 9c). Quantitative evaluation in the main text uses a representative 20-object subset of this library.