CROMOGEN: CROSS-MORPHOLOGY DEMONSTRATION RETARGETING FOR MOBILE MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Automated demonstration synthesis for imitation learning avoids costly data collection, but transferring demonstrations across morphologies often produces trajectories that the target robot cannot execute or that collide with obstacles. Mobile manipulation amplifies these challenges as the planner must coordinate base placement and arm motion simultaneously. We present CroMoGen, a learning-free, planning-based framework that synthesizes mobile manipulation demonstrations from different robot and human sources. CroMoGen formulates synthesis as a constrained optimization problem that combines kinematic feasibility, whole-body collision avoidance, and a trajectory alignment objective that preserves task-relevant object interactions. The approach adapts demonstrated motions to the target morphology and environment. Further, unlike methods that decouple base placement from arm motion, CroMoGen jointly optimizes both for greater adherence to the source demonstration. We evaluate CroMoGen on four mobile manipulation task families in RoboCasa and ParaHome. In the challenging human-to-robot setting, CroMoGen reaches a 56% demonstration-generation success rate, exceeding the strongest baseline by 37 percentage points. More details are on our project page:https://anonymoussubmissionrl.github.io/cromogen/

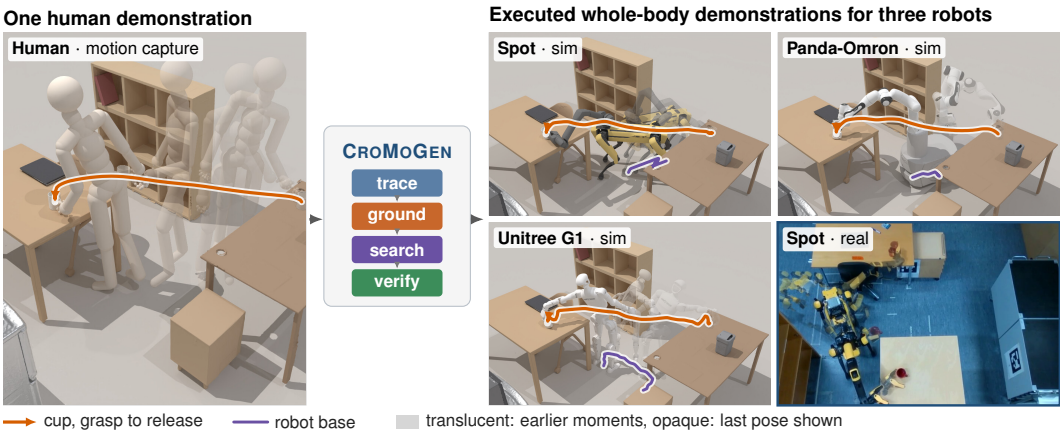


Figure 1: **Cross-morphology demonstration retargeting with CroMoGen**. CroMoGen transfers source mobile manipulation demonstrations to robots with different morphologies. Using the demonstrated object motion as a reference, it adapts each robot's end-effector manipulation motion and jointly plans its base and arm motion.

1 INTRODUCTION

Imitation learning relies on large demonstration datasets, which are expensive and time-consuming to collect with robot teleoperation. Demonstration synthesis aims to reduce this cost by adapting a few source demonstrations to generate numerous synthetic examples in different object configurations. Demonstration-synthesis frameworks such as MimicGen and its successors represent end-effector

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

trajectories relative to the poses of task-relevant objects, and thus can replay the motion adapted to new object poses (Mandlekar et al., 2023; Garrett et al., 2024; Jiang et al., 2024). MimicGen also demonstrates transfer between arms with different parallel-jaw grippers (Mandlekar et al., 2023). It transforms source end-effector targets and reuses source gripper commands, without selecting new contact poses for the target robot.

However, two main bottlenecks still exist in current demonstration-synthesis frameworks. First, when the source and target robots have different kinematics, proportions, or morphologies, the demonstrated motion can become unreachable or collide for the target. Additionally, differences in end-effector design often mean the target robot cannot replicate the demonstrator's manipulation. These challenges motivate solutions that can adapt demonstration motion and contacts to different target morphologies. Prior whole-body retargeting methods such as OmniRetarget and WARP use correspondences between human and robot body landmarks (Yang et al., 2025a; Chen et al., 2026), but they require targets whose bodies resemble the human's. Second, demonstration-synthesis frameworks such as MoMaGen (Li et al., 2025) place the base before each manipulation subtask, whereas humans often move and manipulate at the same time. Retargeting therefore requires adapting contact geometry and coordinating base and arm motion to support the demonstrated interaction.

To address these challenges, we propose CROMOGEN, a planning-based framework for cross-morphology demonstration retargeting in mobile manipulation. CROMOGEN uses an object-interaction-centric representation for transferring motions: the demonstration specifies how the object should move, and the method chooses how the target robot contacts it. For object-guided manipulation, it combines the demonstrated object trajectory with contact poses selected for the target robot to construct the end-effector reference. The reference thus follows the demonstrated object motion and orientation, for example keeping a carried cup upright. CROMOGEN jointly plans base and arm motion across the interaction while avoiding collisions with the scene. Together, contact adaptation and coordinated motion planning support transfer across morphologies without paired source-target demonstrations or a learned retargeting policy.

We evaluate CROMOGEN on human-to-robot transfer from ParaHome (Kim et al., 2024) and on robot-to-robot transfer in ParaHome and from RoboCasa (Nasiriany et al., 2024), across three robots: Spot, Panda-Omron, and Unitree G1. On four ParaHome household tasks, CROMOGEN succeeds in 56.3% of 1,587 retargeting cases. On the 894 cases it solves, MimicGen and WBCMimicGen variants succeed in only 11.0–14.5% when given its end-effector reference. These results show how difficult it is to transfer mobile manipulation demonstrations between morphologies with current demonstration-synthesis methods. On a physical Spot, 12 of 16 selected cup-transport plans from an earlier planner revision succeed. In summary, the main contributions of this work are: 1) an object-interaction-centric framework for transferring mobile manipulation demonstrations across robot morphologies, which adapts contacts to the target robot and jointly optimizes base and arm motion with demonstrated object trajectories as references, and 2) experiments across three mobile embodiments that demonstrate human-to-robot and robot-to-robot transfer and reveal limitations of existing demonstration-synthesis methods.

2 RELATED WORK

Demonstration synthesis across embodiments. Early works demonstration synthesis works mainly focus under the same embodiment setting. MimicGen transforms object-relative motion segments to new object poses (Mandlekar et al., 2023), DexMimicGen extends the approach to bimanual dexterous manipulation (Jiang et al., 2024), SkillMimicGen joins skill segments with motion-planned transitions (Garrett et al., 2024), and DemoGen synthesizes point-cloud observations together with the matching actions (Xue et al., 2025). HumanoidMimicGen separates static manipulation from dynamic locomotion phases (Lin et al., 2026), and MoMaGen selects a base placement before generating the manipulation (Li et al., 2025). Despite these methods can be used for transferring motions between different robotics arms, they cannot guarantee that the target robot can generate feasible motions.CROMOGEN removes this per-embodiment seeding: from a robot or a human demonstration it reconstructs the demonstrated interaction on target embodiments with morphologies.

Mobile manipulation synthesis. Mobile manipulation planning requires reasoning regarding reachability throughout the interaction. DockAnywhere decouples the base from the arm, separating

docking-dependent motion from manipulation skills (Shan et al., 2026). For the coupled solutions such as WBCMimicGen, it tracks end-effector commands with a whole-body velocity quadratic program (Liu et al., 2025). AutoMoMa also jointly optimizes robot-object interaction trajectories from scene models (Niu et al., 2026). Comparing with prior works, CROMOGEN mainly focus on reference-based planning: it plans base and arm motions jointly given a demonstration reference.

Human-to-robot motion retargeting. Human demonstrations are an alternative to robot-specific data collection. DITTO transfers observed manipulation through trajectory transformations (Heppert et al., 2024) and ORION through object-centric interaction graphs (Zhu et al., 2024). PhysicsGen and SPIDER enforce physical consistency with trajectory optimization (Yang et al., 2025b; Pan et al., 2025). At the whole-body level, OmniRetarget preserves human–object–scene relationships (Yang et al., 2025a), and WARP combines an analytical shoulder–elbow–wrist mapping with mobile-base control (Chen et al., 2026). Instead, CROMOGEN follows X-Gen series works, which reconstructs the demonstrated interaction rather than mapping or learning the demonstrated motion. For human sources we assume that metric 3D hand and object trajectories and the scene geometry are available, but not any alignment between the human body and the target robot.

3 PROBLEM FORMULATION

Cross-morphology retargeting seeks a target robot trajectory that reproduces a demonstrated object interaction. Inspired by MoMaGen (Li et al., 2025), we formulate it as a constrained optimization problem. We jointly optimize the base and manipulation joint path under hard feasibility constraints, balancing end-effector tracking, travel time, and speed demand.

The inputs are a source demonstration $\mathcal{D}$, a scene model $\mathcal{E}$, and a target robot model $\mathcal{R}$. The demonstration contains timed, metric hand or end-effector poses, gripper state where available, body position and heading, and task-relevant object or articulated-part poses. The scene model contains geometry, articulations, and initial object configurations, and the robot model specifies kinematics, collision geometry, joint limits, and motion limits. Preprocessing crops the demonstration to the task interval and identifies ordered interaction phases, their objects, required contacts, and collision permissions. Let $\tau \in [0, T_{\mathcal{D}}]$ denote reference time on the cropped demonstration and $t \in [0, T]$ execution time on the robot, where T may differ from $T_{\mathcal{D}}$. For a holonomic planar base at fixed height, we seek a continuous trajectory $\xi(t) = (b(t), q(t))$ with base pose $b(t) \in \mathrm{SE}(2)$ and configuration $q(t)$ of the selected manipulation joints, including torso or waist joints where applicable. An execution schedule sets T and a continuous, nondecreasing map $\sigma : [0, T] \to [0, T_{\mathcal{D}}]$ from execution time to reference time, with $\sigma(0) = 0$ and $\sigma(T) = T_{\mathcal{D}}$, so execution preserves the demonstrated interaction order. Constant intervals of σ pause reference progress, for example during gripper actions, which follow the interaction phases. We optimize only the path of ξ: the configurations it passes through and the reference times at which it reaches them. The schedule, which determines T and σ, is not optimized but follows from this path, the reference timing, the robot's motion limits, and the interaction phases.

Let O_j denote the jth task-relevant rigid object or rigid part of an articulated object, and $X_j^{\mathcal{D}}(\tau) \in \mathrm{SE}(3)$ the demonstrated world pose of a frame fixed to O_j at reference time τ. Preprocessing also constructs a target end-effector reference $X_{\mathrm{ee}}^{\mathrm{ref}}(\tau) \in \mathrm{SE}(3)$ in the world frame, which accounts for the target robot's contact geometry (Section 4.1). With $F_{\mathcal{R}}(b, q) \in \mathrm{SE}(3)$ the robot's world-frame end-effector pose, write $(p(t), R(t)) = F_{\mathcal{R}}(\xi(t))$ and $(p^{\mathrm{ref}}(\tau), R^{\mathrm{ref}}(\tau)) = X_{\mathrm{ee}}^{\mathrm{ref}}(\tau)$ for the planned and reference end-effector positions and orientations. Comparing the planned pose at each execution time t with the reference pose at $\sigma(t)$ gives the tracking errors

$$
\begin{aligned}
e_p(t) &= \left\| p(t) - p^{\mathrm{ref}}(\sigma(t)) \right\|_2, \\
e_R(t) &= \arccos\left(\frac{\mathrm{tr}\big(R^{\mathrm{ref}}(\sigma(t))^\top R(t)\big) - 1}{2} \right).
\end{aligned} \tag{1}
$$

We then seek

$$\xi^\star \in \arg\min_\xi \; J_{\text{track}}(\xi) + J_{\text{time}}(\xi) + J_{\text{speed}}(\xi)$$

$$\text{s.t. } e_p(t) \le \epsilon_p^{\max}(\sigma(t)), \quad e_R(t) \le \epsilon_R^{\max}(\sigma(t)) \quad \forall t \in [0, T], \tag{2a}$$

$$\xi \text{ respects joint and motion limits,}$$

$$\xi \text{ avoids disallowed collisions,}$$

$$\xi(0) \text{ satisfies any prescribed start restrictions,}$$

$$\Psi_{\text{task}}(z_\xi; \mathcal{D}) = 1. \tag{2b}$$

Eq. (2a) bounds the tracking errors by maximum tolerances $\epsilon_p^{\max}(\tau)$ and $\epsilon_R^{\max}(\tau)$, which vary along the reference and may differ between tasks and robots (Appendix B.1). The orientation bound applies only where the reference constrains orientation. The collision constraint excludes self-collision and any scene collision that the phase-specific collision permissions do not allow. The initial configuration $\xi(0)$ is part of the solution, subject only to any prescribed start restrictions. In Eq. (2b), z_ξ is the execution trace of ξ under the chosen controller from $\xi(0)$ in the specified initial scene, recording realized robot and object motions and contacts. The task-specific predicate $\Psi_{\text{task}}(z_\xi; \mathcal{D}) \in \{0, 1\}$ evaluates required contacts, object motions, or outcomes, during execution or at its end, using demonstration-derived targets or native benchmark criteria (Section 4.4).

The objective terms are dimensionless and equally weighted. J_{track} averages a quadratic penalty over active pose errors and reference time, with errors normalized by preferred tolerances $\epsilon_p^{\text{pref}}(\tau)$ and $\epsilon_R^{\text{pref}}(\tau)$ that are positive and no greater than the corresponding maxima. It favors smaller errors even below these tolerances and omits the orientation term where orientation is unconstrained. J_{time} is the estimated minimum travel time of the path under the robot's motion limits, divided by $T_{\mathcal{D}}$. J_{speed} is the reference-time-weighted mean of squared speed utilization, the largest ratio of base or joint speed required at reference timing to the corresponding speed limit.

4 CROMOGEN

CROMOGEN computes an approximate solution to Eq. (2), as summarized in Figure 2. Algorithm 1 in Appendix B gives the planning and evaluation loop. Task-specific preprocessing first constructs the end-effector reference, tracking tolerances, and collision permissions. To keep the search tractable, we plan only at planning frames, a sparse subset of reference times, and sample candidate base poses and inverse kinematics (IK) solutions at each frame. Dynamic programming (DP) then selects the minimum-cost route through these candidates. Because checking every candidate against the scene is expensive, collision checks are lazy: only the configurations and transitions on the selected route are checked, colliding ones are repaired or banned, and DP runs again. We then time the route for execution, and physics simulation checks the task-success constraint. Appendix B gives benchmark-specific details.

4.1 REFERENCE CONSTRUCTION

Preprocessing converts the demonstration into a target end-effector reference $X_{\text{ee}}^{\text{ref}}(\tau)$ on the reference clock, together with tolerances and a collision policy for each interaction phase. The key construction for cross-morphology transfer replaces the demonstrator's contact with one suited to the target robot. It keeps a contact pose $X^{\text{c}} \in \text{SE}(3)$, expressed in the frame of the manipulated object or articulated part O_j, fixed while the object follows its demonstrated motion,

$$X_{\text{ee}}^{\text{ref}}(\tau) = X_j^{\mathcal{D}}(\tau)\, X^{\text{c}}, \tag{3}$$

so the end effector reproduces the object's motion rather than the demonstrator's hand motion. For picking, X^{c} is a grasp for the target gripper. We generate grasp proposals with GraspGen (Murali et al., 2025) or GraspGen-X (Han et al., 2026), remove those whose open gripper would collide with the scene at the grasp or at the approach stand-off, and use the highest-ranked remaining grasp. The grasp is fixed before the search, which determines whether it is reachable. For pushing, it is a surface contact placed near the demonstrated hand. Where the demonstrated path is reused, a minimum-jerk endpoint correction bends it onto the chosen contact while preserving its shape, as for ParaHome cup approaches and RoboCasa pick-and-place. RoboCasa fixture manipulation follows

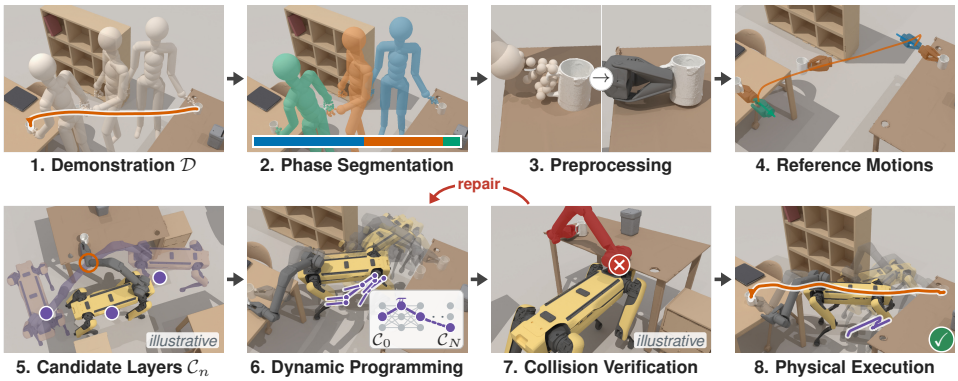


Figure 2: **Overview of CROMOGEN**, retargeting a human ParaHome demonstration to Spot. **(1–4)** The demonstration is segmented into interaction phases, the demonstrator's grasp is replaced by one for Spot's gripper, and the end-effector reference follows the demonstrated cup motion and orientation during the carry. **(5–7)** Dynamic programming selects base poses and arm configurations from candidate layers, with lazy collision checks and repair. **(8)** The trajectory execution.

the recorded end-effector path directly, and the short approach and retreat segments at the contact are straight motions built from the contact geometry. Task and robot settings determine the preferred and maximum tolerances and where orientation is unconstrained. Each phase's collision policy permits specific link–object contacts, such as gripper contact with the target during acquisition, and treats all other scene contact as collision. Appendix B.1 describes the construction for each task family.

4.2 COUPLED BASE–ARM SEARCH

Planning frames. The reference is sampled at the demonstration's frame rate. We select planning times $0 = \tau_0 < \cdots < \tau_N = T_{\mathcal{D}}$ among these dense samples. We keep endpoints, interaction phase boundaries, and changes in tracking requirements or collision policy, and add waypoints where the reference deviates most from interpolation between kept frames. Between planning frames, the route is interpolated and checked during verification (Section 4.3).

Candidate layers. At each τ_n, a layer $\mathcal{C}_n$ holds candidates $x = (b, q)$. We sample base poses on a planar grid covering the reference with a margin for the robot's reach, filter them by body collision and approximate reach, and apply any start restrictions to the first layer. At each remaining base pose, cuRobo (Sundaralingam et al., 2023) solves IK for $X_{\mathrm{ee}}^{\mathrm{ref}}(\tau_n)$ in GPU batches, first under preferred and then, where needed, under maximum tolerances. Screening checks only joint limits and self-collision, and each base pose keeps several solutions chosen for tracking accuracy and joint-space diversity. Arm–scene collisions are checked later, starting from the configurations on routes selected by DP (Section 4.3).

Transitions. A route $\pi = (x_0, \ldots, x_N)$ selects one candidate $x_n \in \mathcal{C}_n$ per layer. Let $\Delta\tau_n = \tau_n - \tau_{n-1}$, and let h_n^b be the base's rest-to-rest motion time from b_{n-1} to b_n under velocity and acceleration limits. We admit a transition if its base motion passes a grid-based collision check and h_n^b is within a budget proportional to the number of dense reference intervals it spans.

Search objective. We minimize a time-discretized version of the objective in Eq. (2), which scores tracking at the planning frames and travel time and speed on the transitions between them,

$$\widehat{J}(\pi) = \frac{1}{T_{\mathcal{D}}} \left[\sum_{n=0}^{N} w_n \bar{\phi}_n + \sum_{n=1}^{N} \left(\max\{h_n^b, h_n^q\} + \Delta\tau_n u_n^2 \right) \right]. \tag{4}$$

Here, $\bar{\phi}_n$ averages the active squared tracking errors of x_n at τ_n, normalized by the preferred tolerances, and the trapezoidal weights w_n sum to $T_{\mathcal{D}}$. The arm travel time h_n^q is the slowest joint's travel time at its speed limit, and u_n is the largest base translation, yaw, or joint speed needed to cross

the interval in $\Delta\tau_n$, relative to its limit. The three terms approximate J_{track}, J_{time}, and J_{speed} with the same normalization and equal weights. Appendix B.2 gives the full definitions.

Dynamic programming. The objective separates into node costs and costs between consecutive layers. A forward DP with backtracking therefore returns a minimum-cost route through the current admitted graph with $O(\sum_{n=1}^{N} |\mathcal{C}_{n-1}||\mathcal{C}_n|)$ candidate-pair evaluations for a fixed number of manipulation joints.

4.3 Collision Verification and Repair

There are mainly three steps in this stage. First, for each DP route, we check the selected arm configurations against the scene under their phase's collision policy. At failing frames, we remove colliding candidates, add scene-aware IK solutions at base poses near the route, and rerun DP. Next, we interpolate the route between planning frames, moving the base through the grid cells used by the search, and sample body and arm motion for collisions. A body collision fails planning, and an arm collision bans that transition before DP reruns. Finally, we recheck the complete route for joint limits, tracking tolerances, and collisions. ParaHome requires the route to pass, whereas RoboCasa executes it regardless. After a localized failure, such as an arm collision between frames or a disconnected candidate graph, we add dense reference samples near the failure as planning frames and search again, within repair and insertion budgets (Appendix B.3).

4.4 Execution Timing and Simulated Evaluation

Timing. We assign execution times to the route, which defines $\xi(t)$, T, and σ. Execution never runs faster than the reference timing or the robot's speed and acceleration limits allow. It slows further around gripper actions and, for pushing and fixture manipulation, throughout contact. In RoboCasa, this slowdown shrinks to fit the task's time limit. The base and arm hold still while the gripper opens or closes, which pauses reference progress.

Simulated evaluation. We place the robot at $\xi(0)$ in the initial scene and track the timed trajectory with the robot's controllers in physics simulation. Retargeting succeeds only if this execution satisfies the task-success constraint (2b), in which case the trajectory and its recorded execution form the generated demonstration. ParaHome checks task-specific contacts and outcomes, and RoboCasa checks the benchmark's own success condition within the time limit. Appendix B.4 gives the timing settings.

5 Experiments

We retarget human motion capture (ParaHome) and robot teleoperation (RoboCasa) to three mobile manipulators and ask three questions. **Q1**: how well does CroMoGen retarget human mobile manipulation across morphologies (Section 5.2)? **Q2**: how well does it retarget between robots, and where does each method fail (Section 5.3)? **Q3**: do its demonstrations execute on a physical robot (Section 5.5)?

5.1 Experimental Setup

Datasets. For **human-to-robot** transfer, we use ParaHome (Kim et al., 2024) as the human motion source, extracting the motion segment of four tasks: Pick_cup (164 motions), Move_cup (164 motions), Push_chair (40 motions), and Close_refrigerator (161 motions), and retarget to Spot, Panda-Omron, and G1 (1,587 transfer cases in total). For **robot-to-robot** transfer, we consider two demonstration sources. First, we use RoboCasa teleoperation demonstrations collected on Panda-Omron and retarget them to Spot and G1 across 10 atomic tasks, four pick-and-place and six open/close, taking the first 100 demonstrations per task. Second, we use the 894 ParaHome demonstrations successfully generated by CroMoGen (315 on Panda-Omron, 281 on Spot, and 298 on G1) to serve as robot-sourced demonstrations. We built a cross-robot-to-robot demonstration-retargeting matrix to evaluate and compare different methods' demonstration-retargeting performance (1,788 cases). Unlike prior datasets in which manipulation and base motion occur sequentially, our tasks require the base to move during manipulation.

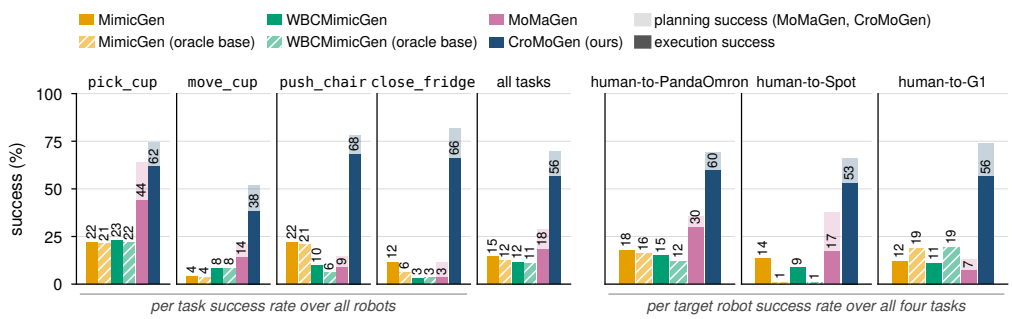


Figure 3: **Human-to-robot demonstration retargeting evaluation.** Human-to-robot demonstration retargeting synthesis success rate of each method per task over all robots (left), and per target robot over all four tasks (right). CROMOGEN leads on every task and robot and succeeds in 56.3% of all cases, while the baselines reach only 11.0–18.3% even on the cases CROMOGEN solved.

Baselines. We compare against five baselines. **MimicGen** (Mandlekar et al., 2023) adapts each source end-effector segment to the new object pose, with the base held fixed. **MimicGen (oracle base)** instead replays the source's base path. **WBCMimicGen** (Liu et al., 2025) tracks the transformed path with a whole-body velocity controller that moves the base and arm together. **WBCMimicGen (oracle base)** replays the source's base path and solves only for the arm. **MoMaGen** (Li et al., 2025) samples a base placement per subtask and then plans base and arm motion with cuRobo (Sundaralingam et al., 2023), so base motion and manipulation are separate phases. From human motion, the baselines use CROMOGEN's preprocessing, which produces the end-effector reference and contact poses, and replace its retargeting with their own, on the cases CROMOGEN solves. Between robots in ParaHome, they start from CROMOGEN's demonstrations for the source robot.

5.2 HUMAN-TO-ROBOT RETARGETING

We first evaluate the success rate of human-to-robot demonstration retargeting, the most difficult setting because of morphology mismatch and inaccurate, noisy data. Givem in total 529 ParaHome human motions, CROMOGEN produces 894 executable demonstrations (56.3% of the 1 587 human-to-robot cases), where human-to-PandaOmron achieves a 59.5% success rate, human-to-Spot 53.1% and human-to-G1 56.3%. The details and per-task breakdown are also presented in Figure 3.

As comparisons, all five baselines perform poorly on all human-to-robot demonstration retargeting tasks, where they only succeed in only 11.0–18.3% (Figure 3). MimicGen methods fail because they perform only object-centric end-effector pose transformations and ignore workspace constraints in mobile manipulation tasks. WBCMimicGen suffers from collision-unaware planning. MoMaGen fails because it often cannot find a feasible base location (50.0% of its cases) and fails to generate physically valid arm motions due to inverse-kinematics infeasibility in cluttered environments (18.7%). The performance gap reflects a fundamental limitation of baselines: they decouple base locomotion from arm manipulation, while human demonstrations tightly couple whole-body behaviors. Without joint optimization under workspace and collision constraints, existing methods tend to generate unreachable or physically infeasible motions.

5.3 ROBOT-TO-ROBOT RETARGETING

Beyond human-to-robot retargeting, we also evaluate how well demonstrations can be transferred across different mobile robot morphologies. The experiments are conducted in two simulations: ParaHome and RoboCasa. In ParaHome, we use CROMOGEN's generated successful mobile manipulation trajectories as source demonstrations. Then we conduct exhaustive pairwise retargeting and report the results as a cross-robot transfer matrix, where each row indicates the source robot and each column indicates the target robot. In RoboCasa, where only official Panda-Omron teleoperation demonstrations are available, we transfer Panda-Omron demonstrations to Spot , G1, and Panda-Omron itself and report the corresponding success rates.

As shown in Fig. 4, our method CROMOGEN achieves the highest success rate for all ParaHome cross-robot pairs, reaching 66.7%. On RoboCasa, it respectively reaches 86.2%, 48.6%, and 23.2%

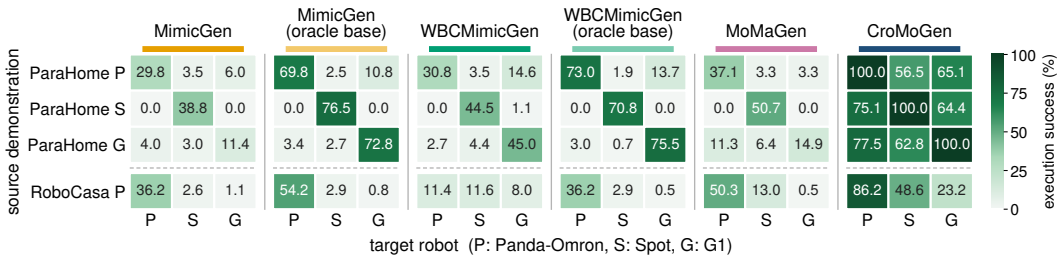


Figure 4: **Robot-to-robot demonstration retargeting evaluation.** Robot-to-robot demonstration retargeting synthesis success rate of each method from each source demonstration (rows) to each target robot (columns). Here P is for Panda-Omron, S for Spot and G is for Unitree G1 robot. CROMOGEN leads on every cross-robot transfer and succeeds in 66.7% of all ParaHome cross-robot cases, while the baselines reach only 2.9–4.5% on the same cases.

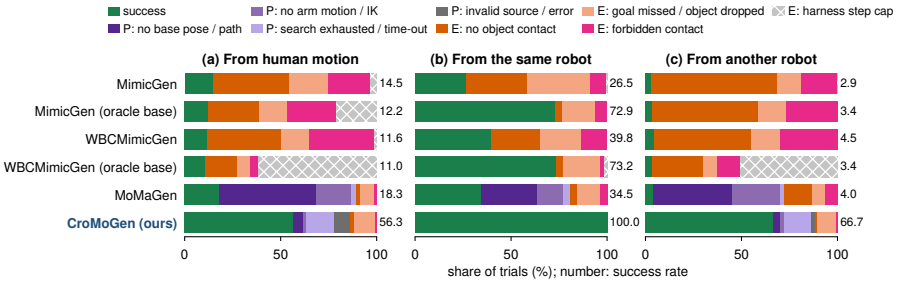


Figure 5: **Where each method fails**: the failure cases analysis. Where P indicates planning, E indicates execution and the number is for success rate, %).

on Panda-Omron, Spot, and G1. Almost all baselines methods are only capable of doing same morphology mobile manipulation demonstration retargeting. In addition, performance varies substantially across different robots. Panda-Omron is the most successful target robot in ParaHome tasks. Its 7-DoF arm and torso lift potentially help accommodate different source motions by providing flexibility in reaching the required contact poses (Table 3). RoboCasa-to-G1 transfer remains difficult, all failures are mostly due to frequent search exhaustion and timeouts. The baseline failures show why geometric trajectory transfer alone is insufficient. A transformed end-effector path and an inherited base trajectory can be incompatible with the target robot's workspace. In our experiments, inverse-kinematics-based or whole-body tracking often fails to establish contact or avoid collisions (Sec 5.4). The MoMaGen method checks collisions but frequently fails to find a feasible base placement or arm trajectory. These results highlight the importance of jointly optimizing manipulation and mobile-base motion for cross-morphology demonstration retargeting.

5.4 WHERE AND WHY FAILURES HAPPEN

Baseline failures. MimicGen and WBCMimicGen family fail mainly during execution (Figure 5). In ParaHome cross-robot transfer, 65.8% and 50.4% of their trials failed because they cannot establish object contact. MimicGen and WBCMimicGen transformed end-effector path can be unreachable from the target robot's chosen or replayed base poses, preventing the gripper from reaching the object. More specifically, WBCMimicGen also makes unintended contact with the refrigerator door through non-gripper links. Our MoMaGen baseline implementation fails mainly during planning, when it cannot find a feasible base placement or arm trajectory, and its base-sampling configuration may contribute to these failures (Appendix D).

Failures of CROMOGEN. In ParaHome human-to-robot transfer, planning-stage failures account for 30.1% of all trials, while execution failures account for 13.5%. Planning failures include missing approach segments, unsuccessful trajectory searches, and exhausted search or repair budgets. Execution failures mainly involve incomplete object interactions, such as refrigerator doors remaining open

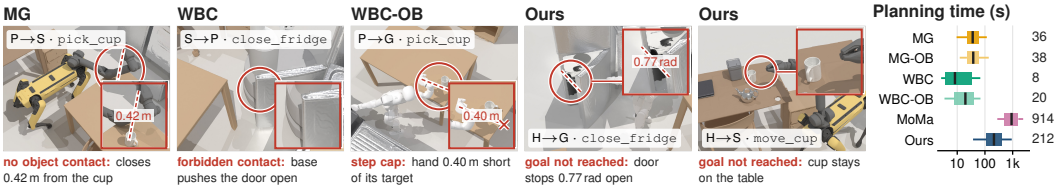


Figure 6: **Representative execution failures.** The red circle marks the failure and the inset magnifies it (P: Panda-Omron, S: Spot, G: G1). The baselines fail away from the task, closing on air, pushing the door with the base, or stalling short of an unreachable target point, while CROMOGEN's execution failures happen at the contact itself.

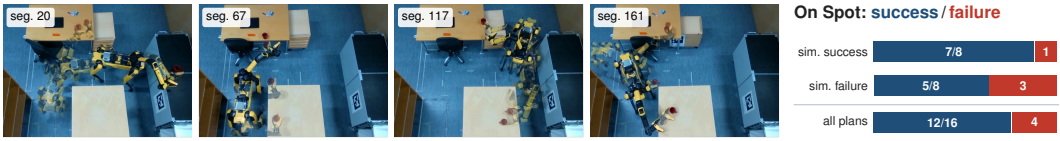


Figure 7: **Physical execution on Spot**: `move_cup` plans examples (left) and the outcome of success rate on the robot (right).

or cups failing to reach their destination. These cases suggest that reliable contact execution remains a challenge even after geometric planning succeeds.

5.5 REAL WORLD EVALUATION

This experiment is designed to answer this question: can CROMOGEN generated plans execute on a real physical-world robot? To answer this, we reproduced the ParaHome setting in our robot lab. In this setting, we aligned the furniture, obstacles, robot positions, and object locations with the ParaHome simulation. During testing, we use CROMOGEN to generate 16 `move_cup` plans in simulation, and execute the plan rollouts in similar real-world physical settings. In simulation, we have eight trajectories successfully planned and executed and eight trajectories successfully planned but failed. In the real world, we have twelve trajectories succeed. Seven of the eight simulated successes transfer (Fig. 7). Five of the eight simulated failures succeed. We found that the reason is that, due to the incorrect and unstable simulation collision-checking mechanism, the robot encounters problems grasping or moving the objects, resulting in objects flying or dropping motions, which is impossible in the real world. This explains why the plans failed in simulation but succeeded in the real world. The failure case in the real world is mainly due to misalignment between real-world and simulation settings (e.g., table size differences). In real experiments, Spot's collision-avoidance, safety-constrained behavior can also cause the plan to drift from the planned trajectory, leading to failure. In summary, CROMOGEN can generate real-world feasible mobile manipulation plans.

6 CONCLUSION AND LIMITATIONS

We presented CROMOGEN, a framework for cross-morphology demonstration retargeting in mobile manipulation. It combines object-interaction-centric motion references with joint optimization of manipulation and mobile-base motion, effectively converting mobile manipulation sdemonstrations to robots with different morphologies. Experiments provide evidence: on ParaHome, CROMOGEN achieves 56.3% success in human-to-robot transfer in a difficult, cluttered environment and 66.7% in all cross-robot demonstration transfer. Our real-world case study provides initial evidence that plans from an earlier planner revision are executable in practice.

For **limitations**, we emphasize that our work focuses on *demonstration synthesis and retargeting*, rather than proposing a new imitation learning solution, so it requires source demonstrations, known scene geometry, and task-specific contact rules. It assumes planar holonomic base motion and fixed contact poses, limiting its ability to handle locomotion constraints and contact adjustments during execution. Whole-body planning remains relatively computationally demanding compared with methods that do not plan. Future work will improve planning efficiency, automate contact specification, and extend this method to a real-time planning policy.

AI USE STATEMENT

In this work, we used generative AI tools for limited research assistance. AI coding tools were used to assist with the implementation of baseline methods, evaluation scripts, and to review the correctness and consistency of these implementations. AI tools were also used to assist with paper editing by refining sentence clarity and readability; they were not used to generate the full manuscript. Additionally, AI-assisted tools were used to render simulation results and generate figures and plots. All AI-assisted outputs were reviewed and verified by the authors, who take full responsibility for the final content of this work.

ETHICS STATEMENT

The human demonstrations come from the publicly released ParaHome dataset (Kim et al., 2024), used under its license. This work collects no new human motion data. The generated data are robot state trajectories in simulation. The physical Spot runs took place in a laboratory and involved no human subjects.

REPRODUCIBILITY STATEMENT

Section 4 and Appendix B specify every stage of CROMOGEN, including the objective in closed form, the tracking tolerances and the grid resolution, and Table 3 lists the per-robot parameters. Section 5.1 defines the cases and sources, Section 4.4 the success criteria, and Appendix D records every declared deviation of the baseline implementations. Every number in Section 5 except the hardware runs (Section 5.5) is regenerated by the table and figure scripts from a table with one row per trial and a per-case cup-orientation table derived from the rollouts. The anonymized supplementary archive contains these tables, the per-trial metrics and the scripts. The code of CROMOGEN is available at https://anonymous.4open.science/r/cromogen/. The baseline implementations and the preprocessed demonstrations will be released.

REFERENCES

Sai Haneesh Allu, Jishnu Jaykumar P, Ninad Khargonkar, Tyler Summers, Jian Yao, and Yu Xiang. HRT1: One-Shot Human-to-Robot Trajectory Transfer for Mobile Manipulation, 2025. URL https://arxiv.org/abs/2510.21026.

Zhenyang Chen, Chuizheng Kong, Chuye Zhang, Yuanshao Yang, Lawrence Y. Zhu, Shreyas Kousik, and Danfei Xu. WARP: Whole-Body Retargeting for Learning from Offline Human Demonstrations, 2026. URL https://arxiv.org/abs/2606.29940v2.

Caelan Garrett, Ajay Mandlekar, Bowen Wen, and Dieter Fox. SkillMimicGen: Automated Demonstration Generation for Efficient Skill Learning and Deployment, 2024. URL https://arxiv.org/abs/2410.18907.

Beining Han, Yu-Wei Chao, Erwin Coumans, Clemens Eppner, Jia Deng, Stan Birchfield, and Adithyavairavan Murali. GraspGen-X: Cross-Embodiment 6-DOF Diffusion-based Grasping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20878–20889, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Han_GraspGen-X_Cross-Embodiment_6-DOF_Diffusion-based_Grasping_CVPR_2026_paper.html.

Nick Heppert, Max Argus, Tim Welschehold, Thomas Brox, and Abhinav Valada. DITTO: Demonstration Imitation by Trajectory Transformation, 2024. URL https://arxiv.org/abs/2403.15203.

Zhenyu Jiang, Yuqi Xie, Kevin Lin, Zhenjia Xu, Weikang Wan, Ajay Mandlekar, Linxi Fan, and Yuke Zhu. DexMimicGen: Automated Data Generation for Bimanual Dexterous Manipulation via Imitation Learning, 2024. URL https://arxiv.org/abs/2410.24185.

Justin Kerr, Chung Min Kim, Mingxuan Wu, Brent Yi, Qianqian Wang, Ken Goldberg, and Angjoo Kanazawa. Robot See Robot Do: Imitating Articulated Object Manipulation with Monocular 4D Reconstruction, 2024. URL https://arxiv.org/abs/2409.18121.

Jeonghwan Kim, Jisoo Kim, Jeonghyeon Na, and Hanbyul Joo. ParaHome: Parameterizing Everyday Home Activities Towards 3D Generative Modeling of Human-Object Interactions, 2024. URL https://arxiv.org/abs/2401.10232.

Chengshu Li, Mengdi Xu, Arpit Bahety, Hang Yin, Yunfan Jiang, Huang Huang, Josiah Wong, Sujay Garlanka, Cem Gokmen, Ruohan Zhang, Weiyu Liu, Jiajun Wu, Roberto Martín-Martín, and Li Fei-Fei. MoMaGen: Generating Demonstrations under Soft and Hard Constraints for Multi-Step Bimanual Mobile Manipulation, 2025. URL https://arxiv.org/abs/2510.18316v4.

Kevin Lin, Ajay Mandlekar, Caelan Reed Garrett, Nikita Chernyadev, Yu Fang, Runyu Ding, Yuqi Xie, Justin Tran, Linxi Fan, and Yuke Zhu. HumanoidMimicGen: Data Generation for Loco-Manipulation via Whole-Body Planning, 2026. URL https://arxiv.org/abs/2605.27724.

Minghuan Liu, Zhengbang Zhu, Xiaoshen Han, Peng Hu, Haotong Lin, Xinyao Li, Jingxiao Chen, Jiafeng Xu, Yichu Yang, Yunfeng Lin, Xinghang Li, Yong Yu, Weinan Zhang, Tao Kong, and Bingyi Kang. Manipulation as in Simulation: Enabling Accurate Geometry Perception in Robots, 2025. URL https://arxiv.org/abs/2509.02530.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A Data Generation System for Scalable Robot Learning using Human Demonstrations, 2023. URL https://arxiv.org/abs/2310.17596.

Adithyavairavan Murali, Balakumar Sundaralingam, Yu-Wei Chao, Wentao Yuan, Jun Yamada, Mark Carlson, Fabio Ramos, Stan Birchfield, Dieter Fox, and Clemens Eppner. GraspGen: A Diffusion-based Framework for 6-DOF Grasping with On-Generator Training, 2025. URL https://arxiv.org/abs/2507.13097.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-Scale Simulation of Everyday Tasks for Generalist Robots, 2024. URL https://arxiv.org/abs/2406.02523.

Quang-Nam Nguyen and Quang-Cuong Pham. Planning Optimal Trajectories for Mobile Manipulators under End-effector Trajectory Continuity Constraint. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 14356–14362, 2024. doi: 10.1109/ICRA57147.2024.10611630.

Yida Niu, Xinhai Chang, Xin Liu, Ziyuan Jiao, and Yixin Zhu. Scalable Trajectory Generation for Whole-Body Mobile Manipulation, 2026. URL https://arxiv.org/abs/2604.12565.

Chaoyi Pan, Changhao Wang, Haozhi Qi, Zixi Liu, Homanga Bharadhwaj, Akash Sharma, Tingfan Wu, Guanya Shi, Jitendra Malik, and Francois Hogan. SPIDER: Scalable Physics-Informed Dexterous Retargeting, 2025. URL https://arxiv.org/abs/2511.09484.

Ziyu Shan, Yuheng Zhou, Gaoyuan Wu, Ziheng Ji, Zhenyu Wu, and Ziwei Wang. DockAnywhere: Data-Efficient Visuomotor Policy Learning for Mobile Manipulation via Novel Demonstration Generation, 2026. URL https://arxiv.org/abs/2604.15023.

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, Nathan Ratliff, and Dieter Fox. cuRobo: Parallelized Collision-Free Minimum-Jerk Robot Motion Generation, 2023. URL https://arxiv.org/abs/2310.17274.

Zhengrong Xue, Shuying Deng, Zhenyang Chen, Yixuan Wang, Zhecheng Yuan, and Huazhe Xu. DemoGen: Synthetic Demonstration Generation for Data-Efficient Visuomotor Policy Learning, 2025. URL https://arxiv.org/abs/2502.16932.

Lujie Yang, Xiaoyu Huang, Zhen Wu, Angjoo Kanazawa, Pieter Abbeel, Carmelo Sferrazza, C. Karen Liu, Rocky Duan, and Guanya Shi. OmniRetarget: Interaction-Preserving Data Generation for Humanoid Whole-Body Loco-Manipulation and Scene Interaction, 2025a. URL https://arxiv.org/abs/2509.26633.

Lujie Yang, H. J. Terry Suh, Tong Zhao, Bernhard Paus Graesdal, Tarik Kelestemur, Jiuguang Wang, Tao Pang, and Russ Tedrake. Physics-Driven Data Generation for Contact-Rich Manipulation via Trajectory Optimization, 2025b. URL https://arxiv.org/abs/2502.20382.

Justin Yu, Letian Fu, Huang Huang, Karim El-Refai, Rares Andrei Ambrus, Richard Cheng, Muhammad Zubair Irshad, and Ken Goldberg. Real2Render2Real: Scaling Robot Data Without Dynamics Simulation or Robot Hardware, 2025. URL https://arxiv.org/abs/2505.09601.

Yifeng Zhu, Arisrei Lim, Peter Stone, and Yuke Zhu. Vision-based Manipulation from Single Human Video with Open-World Object Graphs, 2024. URL https://arxiv.org/abs/2405.20321.

APPENDICES

A Positioning against prior systems **14**

B Method details **14**

B.1 Reference construction 14

B.2 Search 16

B.3 Collision verification and repair 16

B.4 Execution timing and evaluation 16

C Target embodiments **16**

D Baseline implementations **17**

D.1 Why we re-implement MoMaGen rather than cite its numbers 18

A Positioning against prior systems

Table 1 compares CroMoGen with demonstration generators and offline retargeting systems by source, target robots, coupling of base and arm, base motion during contact, and enforced constraints.

Table 1: **Demonstration-generation and retargeting systems.** *Targets*: robots that one source demonstration has been shown to produce data for. *Coupling*: span over which base and arm are optimized jointly. *Base in contact*: whether the base may move during manipulation. Constraints: S success, K kinematics, C collision, V visibility, D dynamics, T temporal.

System	Source	Targets	Coupling	Base in contact	Constr.
MimicGen (Mandlekar et al., 2023)	TeleOp	arm models[a]	per-subtask	✗[b]	S
SkillMimicGen (Garrett et al., 2024)	TeleOp	arm models[a]	per-subtask	✗	S, K, C
DexMimicGen (Jiang et al., 2024)	TeleOp	re-instantiated[c]	per-subtask	✗	S, T
DemoGen (Xue et al., 2025)	TeleOp	single	per-subtask	✗	K, C
PhysicsGen (Yang et al., 2025b)	Human (VR)	arm models[d]	per-subtask	✗	K, C, D
MoMaGen (Li et al., 2025)	TeleOp	mobile dual-arm[e]	per-subtask	carry[e]	S, K, C, V
DockAnywhere (Shan et al., 2026)	TeleOp	single	per-subtask	✗	K, C[f]
WARP (Chen et al., 2026)	Human	human-like arm[g]	whole-body	lazy[g]	K
WBCMimicGen (Liu et al., 2025)	TeleOp	single	per-subtask	✓[h]	S, K
HumanoidMimicGen (Lin et al., 2026)	TeleOp	single[i]	per-subtask	carry[i]	S, K, C, T
DITTO, RSRD, R2R2R[j]	Human (video)	1–2 arms[j]	per-subtask	✗	K, C[j]
OmniRetarget (Yang et al., 2025a)	Human (mocap)	humanoids[k]	per-frame[k]	✓	K, C
HRT1 (Allu et al., 2025)	Human (video)	single (Fetch)	base-first[p]	✗	K
Planners without a source demonstration					
AutoMoMa (Niu et al., 2026)	none	—[l]	whole segment[l]	✓	K, C
Nguyen & Pham (2024)	EE path	single	base-first[m]	✓[m]	K, C[m], T
CroMoGen (ours)	Human (mocap)	Spot, Panda-Omron, G1 (planar bases)[n]	contact segment[o]	✓	S, K, C, T

[a] MimicGen transfers Panda demonstrations to Sawyer, IIWA and UR5e, and SkillMimicGen to Sawyer. All are fixed-base arms that share an end-effector convention (Mandlekar et al., 2023, Sec. 4.2). [b] Base actions are copied from the source, assuming base and arm never move together. [c] Each embodiment has its own source demonstrations, so source and target hand must match. [d] A VR-captured human hand demonstration, retargeted to bimanual Kuka iiwa and Franka Panda. [e] One proof-of-concept transfer (Galaxea R1 to TIAGo), without reported rates. The base moves between subtasks and is fixed within each. [f] Stated as "feasibility constraints" without further detail. [g] The solver requires a human-like arm, and the base moves only for large corrections. [h] The formulation allows base motion, but all reported experiments use a fixed-base UR5. [i] Walking and standing manipulation are planned separately. [j] DITTO (Heppert et al., 2024), RSRD (Kerr et al., 2024) and R2R2R (Yu et al., 2025), each on one or two arms. C holds for RSRD only. [k] G1, H1 and Booster T1 qualitatively, with quantitative results for G1. Solved frame by frame. [l] Each robot is planned from a task specification, with one optimization per object-attached segment. [m] Mobile 3D printing without a grasped object. Only the base is collision-checked. [n] All three bases are planned as planar holonomic bases (Table 3). [o] All planning frames of a manipulation segment are coupled in one search. [p] One base pose is optimized first, then the arm trajectory. Systems first posted in 2026 (DockAnywhere, AutoMoMa, HumanoidMimicGen, WARP) are contemporaneous work.

B Method details

This appendix gives the benchmark-specific details of Section 4. Algorithm 1 summarizes the planning and evaluation loop.

B.1 Reference construction

Contact poses. Table 3 lists the grasp generator for each robot. GraspGen grasps are generated in the Franka-hand convention and converted to the robot's end-effector frame. For the G1, this conversion used an identity calibration that placed the grasp's closing axis a quarter turn off, which likely lowered its results on the two cup tasks. Chair pushing and refrigerator closing use a surface contact built from the object and gripper geometry. The refrigerator contact is placed near the demonstrated hand at contact onset. The chair contact lies on the chair's centre line at the demonstrated hand's height, and the chair is pushed along its demonstrated displacement.

ParaHome references. Demonstrated wrist poses are used as the target's tool center point and converted to its end-effector frame. For cup tasks, a minimum-jerk endpoint correction bends the demonstrated hand path onto a pre-grasp pose 0.20 m from the grasp along the approach axis, and a straight approach with fixed orientation then reaches the grasp. Push tasks approach in a straight line with fixed orientation from a 0.30 m stand-off and do not replay the demonstrated hand motion.

Algorithm 1: CROMOGEN cross-morphology retargeting and evaluation

Input : Demonstration $\mathcal{D}$, scene $\mathcal{E}$, robot $\mathcal{R}$
Output : Trajectory ξ and trace z_ξ, or failure

Construct the reference, tolerances, collision permissions, and interaction phases
Select planning times $\tau_0 < \cdots < \tau_N$
repeat
  foreach τ_n **do**
    Sample collision-free base poses on a grid
    Solve IK at each pose, checking joint limits and self-collision, to form $\mathcal{C}_n$
  Admit collision-free base transitions within the time budget
  repeat
    $\pi \leftarrow$ DP route minimizing $\widehat{J}$
    Check the configurations in π against the scene, and replace colliding ones with scene-aware IK solutions
    Check the transitions in π, fail on a body collision, and ban arm-colliding transitions
  until π *passes the scene checks, or repair stops*
  Validate π. After a localized failure of repair or validation, insert planning frames
until π *passes validation, or insertion stops*
if *no route satisfies the benchmark's acceptance policy* **then return** failure
Retime π and add gripper actions to obtain ξ
Simulate execution to obtain z_ξ
if $\Psi_{\text{task}}(z_\xi; \mathcal{D}) = 1$ **then return** (ξ, z_ξ)
return failure

Every task ends with a straight retreat to a stand-off pose when the demonstration releases within the segment.

RoboCasa references. Fixture manipulation follows the recorded end-effector path, converted to the target robot's end-effector frame and shifted by task-specific contact offsets. For Spot and G1 pick-and-place, a straight approach along the grasp's approach axis from a 0.15 m stand-off precedes the carry, a straight 0.15 m retreat follows release, and minimum-jerk connectors join both to the recorded path. When a Panda-Omron recording is retargeted to Panda-Omron, pick-and-place keeps the demonstrated grasp and the recorded end-effector path unchanged, including the carry, and adds the recorded pre-grasp and post-release frames as planning frames.

Tolerances. Table 2 lists the tolerance profiles, which are the same for Spot, Panda-Omron, and G1. In RoboCasa opening tasks, the tolerance from contact onward is 0.01 m and 0.05 rad for a Panda-Omron target retargeting its own recording, and at most 0.05 m and 0.5 rad for Spot and G1. For Spot and G1 in opening and pick-and-place tasks, the position tolerance before contact, and after the retreat in pick-and-place, is up to 0.5 m while the demonstrated hand is near its rest pose and narrows as the hand moves away. Orientation is unconstrained while this tolerance exceeds 0.10 m.

Table 2: Tracking tolerance profiles, as position (m) and orientation (rad).

Profile	Used for	Preferred	Maximum
Free space	ParaHome, RoboCasa closing and pick-and-place, except the grasp frame	0.05 / 0.5	0.10 / $\pi/2$
Grasp	The frame where the gripper closes, Panda-Omron opening tasks before contact	0.02 / 0.15	0.05 / 0.5

Collision policies. In RoboCasa, contacts recorded in the demonstration also enter the collision policies. A Panda-Omron target may repeat its recording's arm and base contacts, whereas other targets inherit only the recorded finger and palm contacts, on their own finger and palm links.

B.2 SEARCH

Grid. Base poses are sampled on a 0.05 m grid with 32 yaw bins for Spot and 16 for Panda-Omron and G1.

Search objective. The base motion time h_n^b is the larger of the rest-to-rest translation and yaw times. Let $e_{p,n}$ and $e_{R,n}$ be the errors between $F_{\mathcal{R}}(b_n, q_n)$ and $X_{\text{ee}}^{\text{ref}}(\tau_n)$, computed as in Eq. (1). With $\mathcal{A}_n = \{p, R\}$ where orientation is constrained and $\mathcal{A}_n = \{p\}$ otherwise, the tracking term in Eq. (4) is

$$\bar{\phi}_n = \frac{1}{|\mathcal{A}_n|} \sum_{k \in \mathcal{A}_n} \left(\frac{e_{k,n}}{\epsilon_k^{\text{pref}}(\tau_n)} \right)^2 . \tag{5}$$

The trapezoidal weights are $w_0 = \Delta\tau_1/2$, $w_N = \Delta\tau_N/2$, and $w_n = (\Delta\tau_n + \Delta\tau_{n+1})/2$ for $0 < n < N$. For each transition, let d_n be the planar distance between b_{n-1} and b_n, and $\Delta\theta_n$ the absolute value of their shortest yaw difference. Let $v_{\max}$, $\omega_{\max}$, and $\dot{q}_i^{\max}$ denote the base translation, base yaw, and joint speed limits. The arm travel time h_n^q and speed utilization u_n are

$$\begin{aligned} h_n^q &= \max_i \frac{|q_{n,i} - q_{n-1,i}|}{\dot{q}_i^{\max}}, \\ u_n &= \max\left\{ \frac{d_n}{v_{\max}\Delta\tau_n}, \frac{\Delta\theta_n}{\omega_{\max}\Delta\tau_n}, \frac{h_n^q}{\Delta\tau_n} \right\}. \end{aligned} \tag{6}$$

Panda-Omron corridor variant. For Panda-Omron recordings retargeted to Panda-Omron, RoboCasa first searches within a corridor around the recorded base path, which keeps the base close to the demonstrator's while allowing a better pose nearby. The corridor limits position and yaw deviations to 0.5 m and $45°$, and each squared deviation, divided by the square of its limit, is added to the candidate's cost. If this search finds no complete route, or its route fails any check, we search again without the corridor.

B.3 COLLISION VERIFICATION AND REPAIR

Budgets. Repair runs at most three rounds per search, and frame insertion at most five rounds.

Carried objects. In ParaHome cup tasks, the cup's geometry is attached to the gripper during collision checks. RoboCasa leaves carried objects out of the search's collision model. It checks the carried object and the finger contacts separately along the timed route, and these checks affect planning only by triggering the second search of the corridor variant.

B.4 EXECUTION TIMING AND EVALUATION

Motion limits. Execution limits every robot to 0.9 rad/s per joint, 0.2 m/s for the Panda-Omron torso lift, and 0.45 m/s (0.54 m/s for G1) and 0.8 rad/s for the base, below the speed limits used in the search. Panda-Omron uses its own acceleration limits. Spot and G1 specify none, so we use provisional limits under which each arm joint reaches its velocity limit in 0.5 s and the base uses the same acceleration limits as in the search. The acceleration limits hold within straight stretches of the route, and at a bend the velocity may change by up to twice the limit in one control step.

Slowdown and gripper actions. ParaHome slows execution threefold from 0.5 s before to 1.0 s after each grasp and release, measured in reference time, and push tasks slow the whole interval from contact to release. RoboCasa applies the same windows around gripper actions in pick-and-place and around each whole contact phase in fixture manipulation. It reduces the slowdown uniformly when the route would exceed the task's time limit minus the 7.5 s reserved at the end, during which the robot holds still. An inserted gripper action takes 1.2 s to close, 0.4 s to hold, and 1.2 s to open.

C TARGET EMBODIMENTS

Table 3: **Target embodiments.** All bases are planned as planar holonomic bases.

Robot	Base	Planned DoF	Gripper	Approach axis	Grasps
Spot	legged	6 arm	single moving jaw	wrist $+x$	GraspGen-X
Panda + Omron	wheeled	7 arm + torso lift	Franka hand	tool $+z$	GraspGen
Unitree G1 (29-DoF)	virtual planar	7 arm + 3 waist	Dex3-1 hand	tool $+x$	GraspGen

D BASELINE IMPLEMENTATIONS

All three baselines were re-implemented for the two simulators used in this paper, and every departure from the official code is recorded as a declared deviation in the implementation. We summarize the ones that bear on the reported numbers. All baselines hold objects by friction, as CROMOGEN does.

Source demonstrations. In RoboCasa every baseline starts from the first 40 human-teleoperated Panda-Omron episodes of each task and makes one generation attempt per episode, for Panda-Omron, Spot and G1. ParaHome provides only human motion capture, which none of the baselines can consume: each transforms an existing robot trajectory. We therefore seed them with the 894 successful CROMOGEN executions, either for the target robot itself (*same robot*) or for another robot (*other robot*, all six ordered pairs), and with the human motion as CROMOGEN preprocesses it (*human*: the hand path mapped to end-effector targets through CROMOGEN's grasp for the target gripper, and the human pelvis as the reference base path). Replayed on their own robot in the baseline harness, 828 of the 894 seeds succeed.

Oracle base. MimicGen and WBCMimicGen also run with the source's base path replayed instead of their own base behaviour (held fixed for MimicGen, solved online for WBCMimicGen): the seed's base path for a robot seed, and the human pelvis path for the human seed. MimicGen with the oracle base succeeds about as often as WBCMimicGen with it (12.2, 3.4 and 72.9% from human motion, another robot and the same robot, against 11.0, 3.4 and 73.2%). With the oracle base, WBCMimicGen's program solves only for the arm along the replayed base path, offline: it iterates toward each end-effector waypoint up to a budget (the official 500 iterations for the start and grasp waypoints, 200 for the release and last, 3 for the others, each multiplied by 12 in our port, a declared deviation), and every iteration becomes a row of the reference. On ParaHome, the harness executes at most 10^4 control steps per trial, and a trial whose retimed reference is longer fails.

MimicGen (Mandlekar et al., 2023). In RoboCasa we run the data generator of the RoboCasa fork of MimicGen without modification. As in MimicGen, the base is held fixed during generation. Declared deviations: each generation pool contains a single source demonstration, which disables MimicGen's nearest-neighbour segment selection. The subtask decomposition of the composite tasks is ours, since the fork provides none. An issue inherited from the fork, in which the conversion from actions to target poses is not the exact inverse of the conversion from target poses to actions, introduces errors at non-zero base yaw. In ParaHome, segmentation and interpolation follow the official code, and the transformed end-effector trajectory is tracked by offline inverse kinematics rather than closed-loop control.

WBCMimicGen (Liu et al., 2025). The official implementation targets Isaac Lab and cannot run in RoboCasa, so we re-implement it and keep the official code as a reference only. The whole-body velocity quadratic program solves for the base. Without the oracle base, the source base motion is never replayed. Declared deviations: the planar free-base branch of the quadratic program is used although the paper reports only fixed-base experiments, and default rather than per-configuration gains are used. RoboCasa's base joints carry no velocity limit, so the program may command base velocities the platform does not realize. The figures and tables report the closed-loop formulation. An open-loop variant, run as a control on ParaHome, executes 35.9% of the same-robot and 2.7% of the other-robot seeds. In RoboCasa, as in the official code, an attempt ends when a non-final subtask ends without its termination signal, read once after its last waypoint (a subtask gate). The official code then resamples the attempt. With one attempt per episode, we count it as a failure, a test MimicGen's generator does not apply.

MoMaGen (Li et al., 2025). Inverse kinematics uses cuRobo's batched solver with the official seed count. The official implementation grasps with OmniGibson's assisted mode, which attaches the object kinematically (Appendix D.1). We use friction grasps and keep a weld constraint only as a control. Declared deviations: the start pose is searched per target robot, the subtask specifications of the composite tasks are ours, and the base-sampling region is a fixed box. On ParaHome, MoMaGen runs on the first 40 seeds per task and source robot with a 600 s planning cap, and in RoboCasa each attempt has an 1800 s horizon. We report the ParaHome MoMaGen rows as *provisional*: they were produced with the configuration above (a 600 s planning cap, a fixed-box base sampler, our subtask specifications, the first 40 seeds), and a later fidelity audit changed several defaults to the official ones. A validation run with those defaults had not finished at submission.

D.1 Why we re-implement MoMaGen rather than cite its numbers

We port MoMaGen into the simulators used in this paper rather than quoting its reported generation success rates. The reason is not caution about reproduction: the two quantities count different physical events, and a head-to-head table would favour one method or the other depending only on the simulator. We record the reason here so that the choice can be audited.

MoMaGen is defined only for OmniGibson. Its release provides a single environment interface, `env_interfaces/omnigibson.py`, and no robosuite, RoboCasa or MuJoCo backend exists in the code. Every published number is an OmniGibson/PhysX measurement on a Galaxea R1.

Grasps are kinematic attachments, not friction. All twelve source and processed demonstrations shipped with the release set `grasping_mode: "assisted"` in the `env_args` from which generation reconstructs the environment, so the mode is active during generation and not only during teleoperated collection. In OmniGibson this mode "will magnetize any object touching and within the gripper's fingers": `_establish_grasp` creates a `FixedJoint` between the end-effector link and the object link (a `SphericalJoint` when the target link has a non-fixed parent joint). Once that joint exists, friction, finger force and wrench control no longer bear on whether the object is retained, so it cannot slip. Consistently, the grasp termination signal reads the attachment bookkeeping rather than contact state, returning `_ag_obj_in_hand[arm] == candidate_obj`, and the released fork sets `GRASP_WINDOW = 0`, removing the upstream 0.3 s of sustained contact otherwise required before the joint is formed.

This is a reasonable engineering choice in its own setting, and it is the stricter of OmniGibson's two magnetic modes: the object must contact at least two finger links *and* intersect a gripper raycast, so the hand does close around it. Assisted grasping is standard practice in BEHAVIOR-1K because PhysX articulation friction is unreliable at these scales. We note it not as a defect but because it fixes what a generation success rate measures.

Consequence. A MoMaGen success counts a base and camera placement from which an attachment formed and, by construction, held. A success in a friction-grasping simulator counts a placement from which a grasp survived transport under the realized controller. The second is a strictly stronger event, so the difference between the two numbers is not attributable to the sampling strategy, which is the quantity we wish to isolate. Running MoMaGen in our simulators with friction grasps, as above, holds the simulator, contact model, source demonstrations and success predicate fixed, so the remaining difference is the generation strategy.

The same reasoning constrains us. We do not claim our absolute yields are comparable to MoMaGen's, in either direction, and we report no such comparison.