Belief-Overlap Partial Inversion: Object-Centric Flow Replanning for Humanoid Loco-Manipulation

Anonymous Authors
bopi-loco.github.io

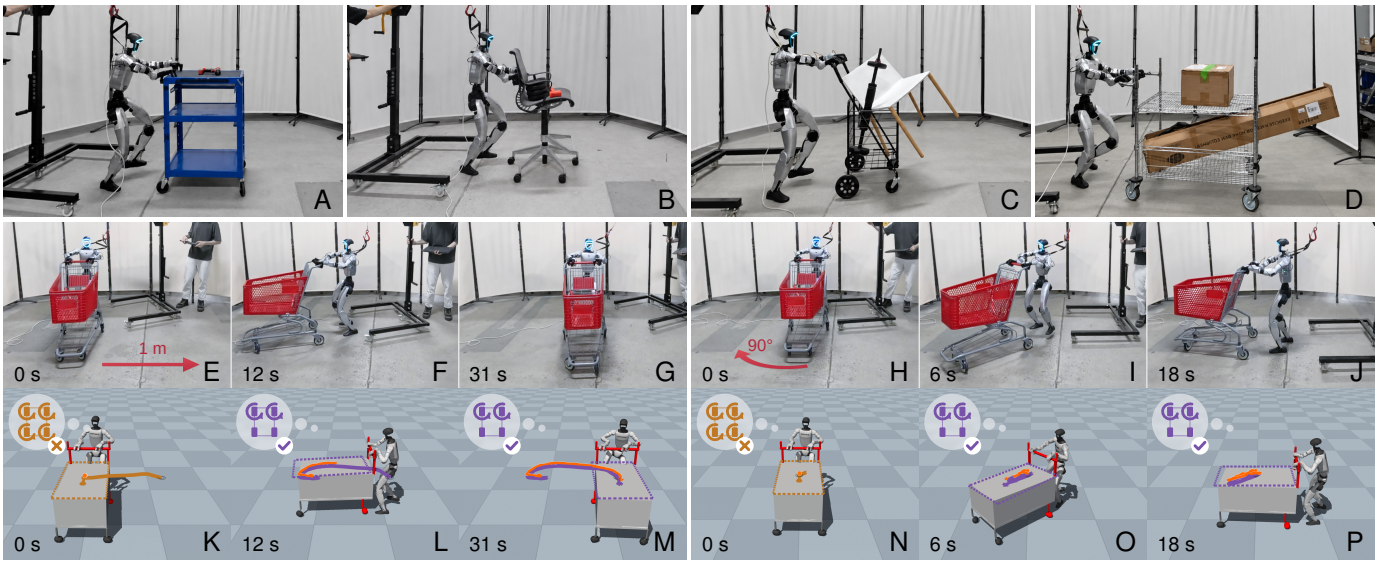

Fig. 1. **From versatile object-centric control to adaptive planning.** Our whole-body controller pushes wheeled objects with different masses, geometries, and holonomic or non-holonomic wheel constraints along their planned paths (A–D). BOPI uses live estimates of the object's physical properties to revise lateral (E–G) and turning (H–J) plans for a non-holonomic shopping cart. Corresponding simulations (K–P) illustrate this adaptation: the planner proposes a wrong initial plan based on an omni-cart assumption. After brief interaction reveals the cart's non-holonomic constraint, BOPI revises it to a correct plan while the cart continues moving. Thought bubbles show each plan's assumed wheel constraints. Orange marks the cart's starting position and executed motion. Videos and code at bopi-loco.github.io.

***Abstract*—People often learn how an unfamiliar object moves through a small exploratory push, using its response to guide their next motion. Inspired by this behavior, we study humanoids pushing carts whose mass and caster stiffness are initially unknown. Our object-centric framework closes online estimation, flow-based planning, and whole-body control into a loop, creating a tension: replacing the plan whenever the belief shifts disrupts the maneuver under way, while keeping it strands the robot on a path its evidence has ruled out. Belief-Overlap Partial Inversion (BOPI) scales revision to agreement, inverting the flow under the old belief only as far as the beliefs disagree before reconstructing under the new one. Across 18 simulated goals it lowers pooled final position and yaw error by roughly 33% and 47% against fixed-depth replanning, and on hardware it translates and turns a non-holonomic cart.**

I. Introduction

People rarely commit to a way of moving an unfamiliar object before testing it; a short push reveals whether it rolls freely, resists, or veers off line, and the plan is adjusted before the motion has finished. We pursue this ability in humanoids pushing carts whose mass and caster stiffness are unknown at the outset [3–5]. What is at stake is the path and not merely the effort: an omnidirectional cart can be driven laterally to its goal, whereas a constrained rear axle admits only a curved approach, so the geometry of the maneuver is settled by a property the robot cannot observe until it is already pushing.

Our framework treats the cart, rather than the robot, as the thing being planned for. A recurrent estimator maintains a belief over the cart's physical properties, a conditional flow planner generates a timed object trajectory conditioned on that belief, and a whole-body controller tracks that trajectory while preserving grasp and balance. Because motion is what produces evidence, the three components close a loop in which every correction arrives mid-maneuver, and that timing is the difficulty: regenerating the plan from each new belief, on top of noisy pose estimates, perturbs the very trajectory the controller is following, while refusing to regenerate leaves the robot committed to a path its own measurements no longer support. Building on plan-reuse methods [1,6,7], we ask: *how much should a new estimate change the robot's plan?*

Belief-Overlap Partial Inversion (BOPI) answers with the beliefs themselves. As the replanning component of our framework, it partially reverses plan generation under the old belief and reconstructs under the updated one, with the depth of reversal set by how far the two beliefs overlap: when successive beliefs agree the plan is barely disturbed, and as they diverge

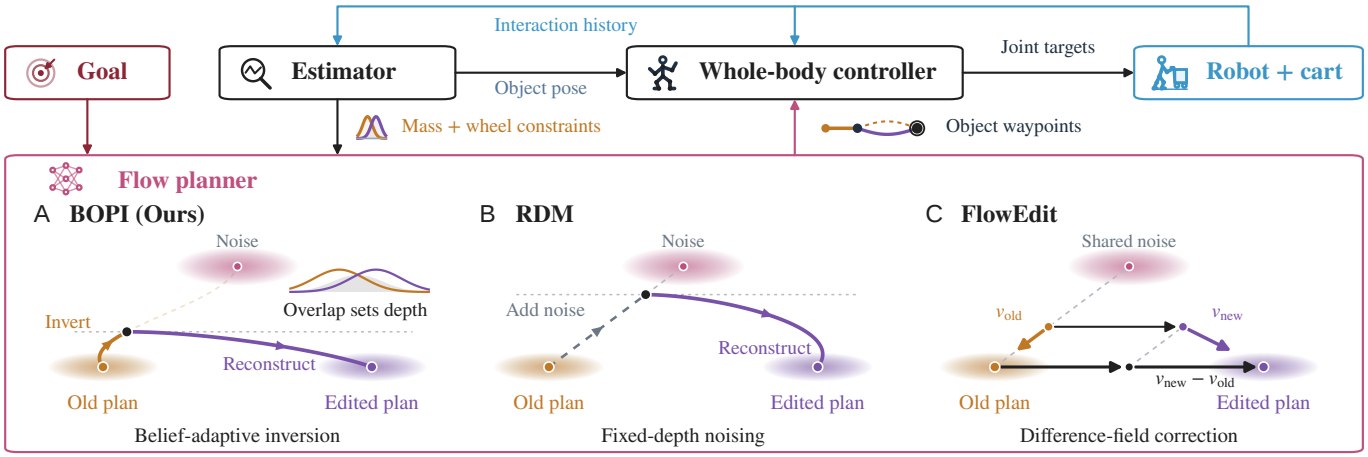

Fig. 2. **An object-centric estimation–planning–control framework.** Online estimation informs object planning and whole-body control (trajectory waypoint tracking). As estimates change, the planner can revise the remaining path using (A) BOPI, which uses belief overlap to decide how far to run the flow backward under the old belief, then runs it forward under the updated belief; (B) RDM [1], which blends the current plan with a fixed proportion of random noise, then runs the flow forward under the updated belief to reconstruct the path; or (C) FlowEdit [2], which uses shared noise to evaluate the flow under both beliefs, then edits the path using the difference between the flow directions under the updated and old beliefs. Orange/purple denote old/new plans and beliefs.

more of it is rewritten. The rule requires no per-situation tuning, and the change it induces in the encoded plan vanishes as the overlap approaches one.

Experiments span a controlled nonlinear benchmark, where BOPI attains higher task success with smaller cumulative plan changes, and 18 goals in G1–Dex3 simulation. Figure 1 shows object-centric control across cart types (A–D), real-robot translation and turning (E–J), and simulation replay of recorded plans and updates (K–P).

Our contributions are: **(i)** an object-centric estimation–planning–control framework that adapts cart trajectories while tracking continues; **(ii)** BOPI, its belief-adaptive replanning method, with an exact-integration bound under which the encoded plan change vanishes as belief overlap approaches one; and **(iii)** simulation and hardware evaluations showing lower goal error in simulation and lower final position and tracking error on hardware.

II. Related Work

A. Object-centric loco-manipulation

Loco-manipulation couples locomotion with manipulation through whole-body control [5,8–12]. A recurring move is to state the task on the object rather than on the robot. Mittal et al. [13] plan interactions with articulated objects in unknown scenes; Omnigrasp [14] and MaskedManipulator [15] drive simulated humanoids from desired object motion; ResMimic [16] adds an object-conditioned residual to a general motion tracker; and OmniContact [17] chains contact-level skills for generalizable loco-manipulation. We share the object-level specification with this line of work and differ in what happens afterwards: the object trajectory is not fixed once issued but revised as the object's properties are inferred.

B. Cart pushing under unknown dynamics

Cart pushing is the classical instance of that coupling. Harada et al. [18] derive walking motion for pushing manipulation, Scholz et al. [19] treat the cart as a movable obstacle during mobile-manipulator navigation, and Hawley and Suleiman [3,20] control a humanoid pushing a loaded cart and later navigate with it under SLAM. Because the cart's response turns on parameters the robot cannot see, several works estimate them while pushing: Sun et al. [21] identify cart parameters online, Foster et al. [4] keep inertial estimates physically consistent for humanoid loco-manipulation, Zhang et al. [22] optimize the push pose and reject disturbances without an accurate dynamic model, and Sriganesh et al. [23] couple estimation with planning for quadrupedal pedipulation. Our question is orthogonal to how the estimate is formed: we ask what the planner should do with it once it moves.

C. Cross-embodiment control

Differences in physical properties or morphology change a system's dynamics and constraints, the concern of cross-embodiment control [24–28]. Rather than adapting control of the robot itself, we study how the object's path within the robot–object system should change as its properties are inferred.

D. Adaptive planning

Once properties are inferred [29–31], the plan has to follow. Diffusion and flow models [32–36] generate trajectories that can be regenerated under a new condition, and several methods reuse the previous solution instead of discarding it. SDEdit [37] noises an input and denoises it under the new condition; Replanning with Diffusion Models (RDM) [1] applies this to replanning by adding a fixed proportion of noise to the current plan before reconstruction. FlowEdit [2] avoids inversion altogether, editing a trajectory by integrating the difference between the flow fields the old and new conditions induce, while DDIM [38] supplies the deterministic inversion such methods rely on. A parallel line decides *when* to re-solve rather

than how much of the old solution to keep, through real-time chunking [6,7,39] and noise-space steering [40]. What none of these supplies is a principled setting for the amount of revision: RDM's noise level is a fixed hyperparameter, and FlowEdit's edit follows from the field difference alone. BOPI ties that amount to the overlap between the beliefs that produced the two plans, so the plan changes as much as the evidence has.

III. Method

Our framework separates object planning from whole-body control (Fig. 2). The planner specifies where the cart should move and when it should reach each pose. The whole-body controller follows this trajectory while maintaining balance and hand contact. As interaction reveals more about the cart, BOPI updates the remaining path to better reach the goal.

A. Object trajectories and the flow planner

We represent a timed object trajectory as $\tau = \{(p_h, t_h)\}_{h=0}^{H-1}$. Each of its H waypoints records planar position and yaw in p_h, together with the elapsed physical time t_h. The user specifies a fixed final position and yaw g^{W} in a world frame maintained by localization. At each replanning step, we express this goal as g: the remaining displacement and heading change needed to reach it from the estimated current object pose. Our experiments use $SE(2)$ goals, but the formulation permits other goal representations with an appropriately conditioned flow.

An estimator uses interaction history to infer a belief $q(\theta)$ over a vector θ of object physical properties, such as mass and caster stiffness. The belief mean $c(q) = \mu_q$, together with the goal g, conditions the flow model.

A fixed encoder A converts the trajectory into normalized coordinates $x = A(\tau)$. A conditional rectified flow [34,35] uses flow time $\lambda = 0$ for noise and $\lambda = 1$ for a plan. The intermediate encoding x_λ evolves according to the flow field v_φ, with network parameters φ, as $dx_\lambda/d\lambda = v_\varphi(x_\lambda, \lambda \mid g, c(q))$. We denote integration from flow time a to b, starting at x with g and $c(q)$ fixed, by $\Phi_q^{a\to b}(x; g)$. A fresh plan's encoding is $\Phi_q^{0\to 1}(\epsilon; g)$, where $\epsilon \sim \mathcal{N}(0, I)$ and I is the identity matrix.

B. Belief-Overlap Partial Inversion

BOPI repairs the remaining plan using the same trained flow. Let x_o encode this plan, q_o denote its stored belief, and q_n the updated belief.

Overlap determines depth: We measure agreement between the beliefs with their Bhattacharyya coefficient [41,42]. For continuous densities,

$$\begin{aligned} \rho &= \mathrm{BC}(q_o, q_n) = \int \sqrt{q_o(\theta) q_n(\theta)}\, d\theta, \\ d &= 1 - \rho, \qquad \lambda_\star = \rho. \end{aligned} \tag{1}$$

The overlap ρ lies between zero and one. We stop the backward integration at $\lambda_\star = \rho$, so it covers a distance $d = 1 - \rho$ in flow time. Similar beliefs give a shallow reversal; less overlap allows a deeper one. This depth is the squared Hellinger distance and requires no extra threshold or repair-specific training.

Reverse the old plan, then reconstruct: We reverse x_o from flow time 1 to ρ under q_o, obtaining z_ρ. We then reconstruct from z_ρ under q_n:

$$\boxed{z_\rho = \Phi_{q_o}^{1\to\rho}(x_o; g), \qquad x_n = \Phi_{q_n}^{\rho\to 1}(z_\rho; g).} \tag{2}$$

Both steps use the same g. The intermediate state is a trajectory encoding, not a physical pose for the robot to visit. No fresh noise is added. At $\rho = 1$, both integrations have zero length and the plan stays unchanged. At $\rho = 0$, they perform full inversion and reconstruction; values between these extremes give partial repair.

Size of a repair: The following bound describes how much the encoded plan can change under exact integration. Write $v_i(\cdot, \lambda) = v_\varphi(\cdot, \lambda \mid g, c(q_i))$ for $i \in \{o, n\}$.

Proposition 1 (Exact-flow repair): Suppose the solutions exist on $[\rho, 1]$. On a common region containing their paths, assume both fields are L-Lipschitz in encoded state, bounded in norm by V, and satisfy $\|v_n - v_o\|_2 \le \Delta_v$, uniformly on $[\rho, 1]$, with $L, V, \Delta_v \ge 0$. Then

$$\|x_n - x_o\|_2 \le \min\{2Vd,\ d\Delta_v \exp(Ld)\}\,. \tag{3}$$

Here L bounds sensitivity to the input encoding, V bounds the field magnitude, and Δ_v bounds the difference caused by changing the belief.

Proof: Each integration travels at most Vd in encoded space, giving the first bound, $\|x_n - x_o\|_2 \le 2Vd$. For the second bound, run the old flow forward from z_ρ. By uniqueness, this returns x_o. The new flow starts from the same z_ρ, so the distance $e(\lambda)$ between the two forward solutions satisfies $e(\rho) = 0$ and $e' \le Le + \Delta_v$ almost everywhere. Grönwall's inequality gives

$$\|x_n - x_o\|_2 = e(1) \le \Delta_v \int_\rho^1 e^{L(1-s)}\, ds \le d\Delta_v e^{Ld}.$$

Taking the smaller of the two bounds proves Eq. (3). ■

Under these uniform bounds, the change in the encoded plan approaches zero as belief overlap approaches one.

C. Applying updates while tracking continues

While tracking continues, the estimator updates the belief q_n. Before repair, we resample the remaining path into H waypoints p_h in the current estimated cart frame, matching the flow model's input representation. BOPI revises the remaining timed path using the stored belief q_o and updated belief q_n, following Eq. (2), while keeping the final goal g^{W} unchanged. We preserve the first four resampled waypoints to reduce abrupt changes when updating path.

D. Cart-pushing implementation

Belief representation: Gated recurrent unit (GRU) ensembles predict the mean and uncertainty of rear-caster stiffness k_{r} and cart mass m_{cart} in standardized log coordinates: $\log k_{\mathrm{r}}$ and $\log(1 + m_{\mathrm{cart}})$. Here k_{r} denotes rear-caster swivel stiffness ($\mathrm{N\,m/rad}$): small values allow the rear wheels to steer freely, while large values approximate a locked rear axle. We average

the property estimates across models. Let $q_{i,j} = \mathcal{N}(\mu_{i,j}, \sigma_{i,j}^2)$ denote the old ($i = o$) or updated ($i = n$) belief for property j. Its mean is $\mu_{i,j}$ and its standard deviation is $\sigma_{i,j} > 0$. We compute $\rho = \prod_j \mathrm{BC}(q_{o,j}, q_{n,j})$ using the analytic Gaussian overlap [42].

Generating training trajectories: For each physical-property setting θ, an analytic teacher optimizes a sequence of cart-frame controls $\mathbf{u}$ (forward/lateral velocity and yaw rate) and duration T:

$$\min_{\mathbf{u},T} \quad T + \sum_r w_r C_r(\mathbf{u}, T; \theta), \tag{4}$$

subject to the goal pose, specified boundary velocities, and limits on motion and force/torque. Each cost term C_r, indexed by r, sums a normalized squared quantity over time: predicted hand force, force rate, acceleration, lateral velocity, yaw rate, or sideways velocity at an axle. The weights satisfy $w_r \geq 0$. Mass changes the force and torque needed to accelerate the cart. Larger caster stiffness increases the penalty on sideways axle motion and reduces its allowed magnitude.

Training the flow: We convert each teacher trajectory into $H = 32$ timed waypoints (p_h, t_h) for flow training. A temporal U-Net [32,33], using GroupNorm [43] and FiLM conditioning [44], predicts the flow field.

We use the standard flow-matching loss [34,35], with training interpolation $x_\lambda = (1-\lambda)\epsilon + \lambda x$. The expectation is over training examples $(x, g, c(q))$ and independent $\epsilon \sim \mathcal{N}(0, I)$, $\lambda \sim \mathcal{U}[0,1]$:

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}\left[\|v_\varphi(x_\lambda, \lambda \mid g, c(q)) - (x - \epsilon)\|_2^2\right]. \tag{5}$$

We also add a condition-smoothness loss to encourage similar flow predictions for nearby physical-property inputs. With $c = c(q)$, we perturb only c, keeping x_λ, λ, and g fixed:

$$\begin{aligned}\mathcal{L}_{\mathrm{smooth}} = \mathbb{E}\Big[&\big\|v_\varphi(x_\lambda, \lambda \mid g, c+\delta) \\ &- v_\varphi(x_\lambda, \lambda \mid g, c)\big\|_2^2\Big].\end{aligned} \tag{6}$$

Here $\delta \sim \mathcal{N}(0, \sigma_c^2 I)$, with $\sigma_c \simeq 0.0041$ in standardized physical-property coordinates. The total loss is $\mathcal{L} = \mathcal{L}_{\mathrm{FM}} + 0.1\mathcal{L}_{\mathrm{smooth}}$. All compared methods use this same trained flow.

Object-centric whole-body control: The controller maps incoming timed waypoints (p_h, t_h) and robot observations to targets for all joints except the fingers. Inputs include joint positions and velocities, base angular velocity, gravity direction, previous actions, gait phase, hand targets, hand and handle geometry, and an eight-step observation history.

We train the controller with PPO in mjlab [45,46]. Its main task rewards encourage accurate cart position and heading:

$$r_{\mathrm{track}} = 15 \exp\left(-\frac{\|e_p\|_2^2}{0.15^2}\right) + 20 \exp\left(-\frac{e_\psi^2}{0.30^2}\right). \tag{7}$$

Here e_p and e_ψ measure planar position (m) and wrapped heading (rad) differences from the current target interpolated from the timed waypoints (p_h, t_h). Supporting terms encourage stable grasping, balance, wheel contact, and terminal settling, while penalizing excessive speed and sideways wheel slip. We combine these with FALCON's locomotion rewards [9] and will release the complete reward specification and training configuration with our code.

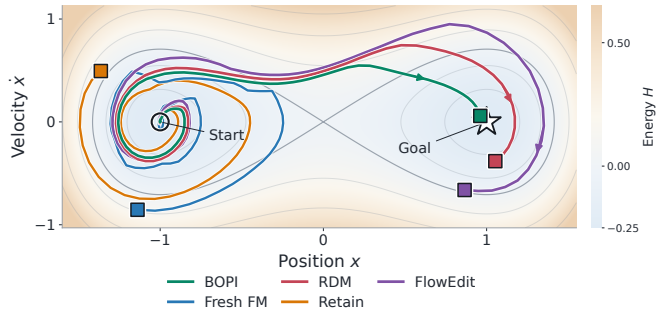

Fig. 3. **Planning in a nonlinear toy problem.** At 10 s (squares), BOPI is near the goal at low speed; RDM and FlowEdit overshoot with higher residual speeds. Background color shows mechanical energy H.

TABLE I
TOY PROBLEM RESULTS.

Method	Success (%)	Position error	Final speed error	Cumulative mutation	Strategy switches
BOPI (Ours)	$\mathbf{54.9 \pm 14.6}$	**0.189**	**0.116**	**0.704**	**2.144**
RDM	40.0 ± 6.1	0.283	0.211	2.212	8.617
FlowEdit	12.6 ± 3.7	1.043	0.367	4.026	9.260
Fresh FM	4.0 ± 0.9	1.567	0.422	23.631	21.045
Retain	2.4 ± 0.3	2.090	0.491	N/A	N/A

IV. EXPERIMENTS

A. Evaluation protocol and baselines

We compare BOPI with four baselines:

- **RDM** adds noise to the old plan at a fixed depth, then reconstructs it under the updated belief [1] (Fig. 2B).
- **FlowEdit** performs inversion-free repair using the difference between new- and old-conditioned flow fields evaluated at paired noisy states [2] (Fig. 2C).
- **Fresh FM** generates a complete plan from fresh Gaussian noise under the updated belief.
- **Retain** keeps the initial plan unchanged.

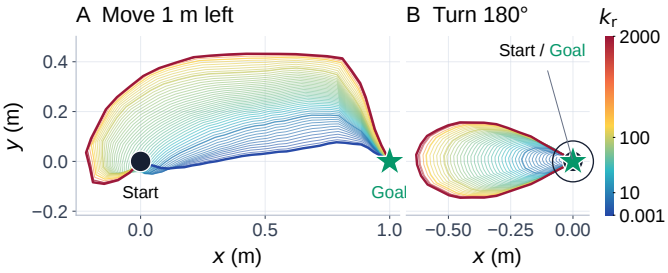

Fig. 4. **Flow-generated plans adapt to wheel constraints.** With the same initial flow noise, the flow generates different paths as rear-caster stiffness changes: move one meter left (A), or turn 180° and return to the starting position (B). Color indicates stiffness; bold curves mark its extremes.

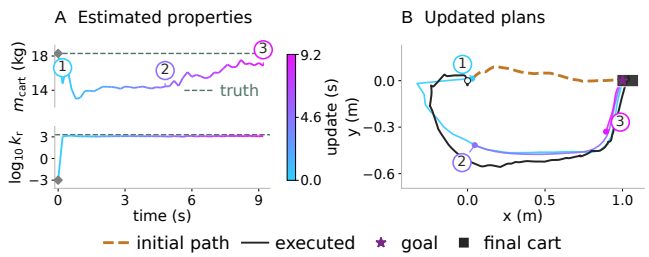

Fig. 5. **From property estimates to updated plans.** As estimates of cart dynamic properties evolve (A), BOPI revises the initial near-straight path into a curved maneuver (B). Matching colors and numbers link estimates to plan updates; black shows the executed motion.

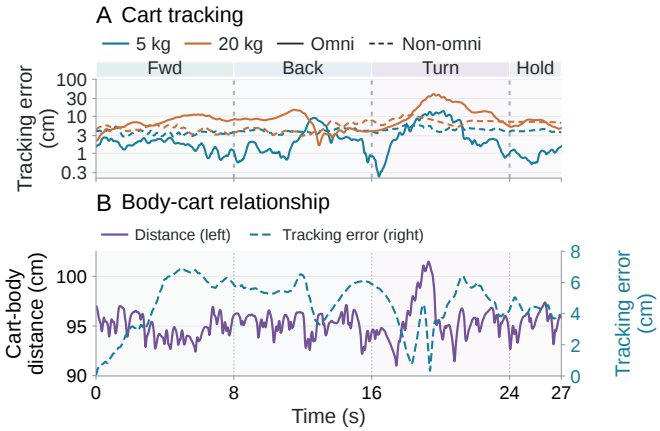

Fig. 6. **Accurate cart tracking with a changing body–cart distance.** We design a path covering common maneuvers: 0.8 m forward travel, 0.6 m reverse travel without changing heading, and a 45° rolling turn. Panel A compares tracking across masses and wheel constraints. In an example execution with a 20 kg cart (B), body–cart distance changes during turning while tracking error remains low, illustrating that the robot can follow the object's path without maintaining a fixed body–cart offset.

B. Controlled toy problem

Setup: We study plan repair on a controlled Duffing oscillator [47], $\ddot{x} = x - x^3 - \beta\dot{x} + 0.43u$, with position x, velocity $\dot{x}$, control $u \in [-1, 1]$, and damping $\beta \sim \mathcal{U}[0.04, 0.42]$. Flows learn optimized controls for moving from $(x, \dot{x}) = (-1, 0)$ to $(1, 0)$ over 10 seconds. Each trial starts with a teacher plan for $\beta = 0.04$. Scripted equal-variance Gaussian beliefs first correct the damping, then fluctuate with correlated noise (belief/noise standard deviation 0.03, correlation 0.95). At 4 Hz, methods revise the full plan but execute only future controls. RDM uses the fixed midpoint flow restart time $\lambda = 0.50$. Terminal success requires $|x - 1| < 0.12$ and $|\dot{x}| < 0.15$. We evaluate each method on 256 trials per flow across four independently trained flows.

Results: An incorrect damping estimate makes the initial plan unreliable: retaining it succeeds in only 2.4% of trials, compared with 97.2% when planning with true damping. BOPI improves success over fixed-depth RDM by 14.8 ± 10.4 percentage points, while reducing cumulative plan changes by 68.2% and final speed error by 45.1% (Table I). Figure 3 shows a representative successful BOPI rollout.

C. Belief-conditioned flow generation

In the toy problem, beliefs about damping guide control generation; for cart pushing, beliefs about mass and wheel constraints guide the generation of timed cart trajectories. Figure 4 shows how wheel constraints change the paths: nearly free rear casters allow direct lateral motion and near-in-place rotation, whereas a constrained rear axle produces curved lateral paths and wider turning maneuvers. Beyond path shape, the plans also adapt their timing: for example, at $m_{\text{cart}} = 18.3\,\text{kg}$, increasing rear-caster stiffness extends the lateral plan's duration from 5.11 to 9.42 s. Separately, the increase in mass belief from 3.3 to 18.3 kg at $k_{\text{r}} = 2000$ increases in the lateral plan length by 8.5 cm.

D. Humanoid cart-pushing simulation

1) Simulation setup: We evaluate all methods in the mjlab [46] G1–Dex3 environment, using a whole-body controller while the robot grasps the cart with both hands. At each reset, we randomize the robot posture and handle height over a range of 0.78–1.00 m, then use inverse kinematics to place both hands on the handle. During controller training, we also sample episodes without a cart so that the deployed policy remains stable when contact is not established.

2) Belief estimator: The belief estimator predicts cart mass and rear-wheel constraint from interaction history. After four seconds of interaction, the estimator distinguishes free from constrained rear wheels with 91.5%–98.4% accuracy and estimates cart mass with approximately 1 kg mean absolute error on held-out trajectories.

Figure 5 illustrates how updated estimates guide BOPI's path repairs. In the illustrated rollout, the cart reaches the goal with mean position and yaw errors of 0.036 m and 0.007 rad.

3) Object-centric whole-body control: Across masses and wheel constraints, our controller follows cart waypoints with a mean RMS position error of 6.53 cm over 8 trials (Fig. 6A). Object-centric tracking allows the robot to adjust its position relative to the cart during turning while continuing to track the cart's waypoints (Fig. 6B).

4) End-to-end planning results: We evaluate 18 goals (32 seeds each): one-meter left/right translations; ±60° or ±90° rotations ending at the starting position; one-meter left/right/forward/backward translations with ±90° turns; and one-meter left/right translations with ±45° turns under an incorrect mass belief. Directions are relative to the initial pushing direction. The first three categories incorrectly assume freely steering rear wheels; wrong-mass tasks use an omnidirectional cart with an assumed mass of 3.3 kg instead of 18.3 kg. We compute cart position and yaw errors from ground-truth poses against the fixed goal g^{W}, averaging over each rollout's final two seconds and then across all rollouts. Replanning is scheduled at 5 Hz; RDM's restart time $\lambda = 0.25$ was selected by a development sweep.

Table II shows that retaining the incorrect initial plan leaves substantial final position and yaw errors (0.465 m /

TABLE II

SIMULATION RESULTS: END-TO-END PLANNING. ENTRIES ARE MEAN FINAL POSITION ERROR (M) / YAW ERROR (RAD).

Task category	**BOPI (Ours)**	RDM	FlowEdit	Fresh FM	Retain
Translation	**0.349 / 0.163**	0.625 / 0.561	0.637 / 0.590	0.870 / 1.039	0.927 / 0.330
Rotation	**0.109 / 0.150**	0.134 / 0.195	0.251 / 0.693	0.754 / 1.209	0.286 / 0.306
Translation + rotation	**0.209 / 0.205**	0.296 / 0.336	0.240 / 0.228	0.835 / 1.165	0.495 / 0.329
Translation + rotation (wrong mass)	**0.105 / 0.039**	0.163 / 0.139	0.277 / 0.129	1.353 / 1.114	0.352 / 0.115
Pooled	**0.179 / 0.152**	0.267 / 0.286	0.295 / 0.349	0.936 / 1.149	0.465 / 0.277

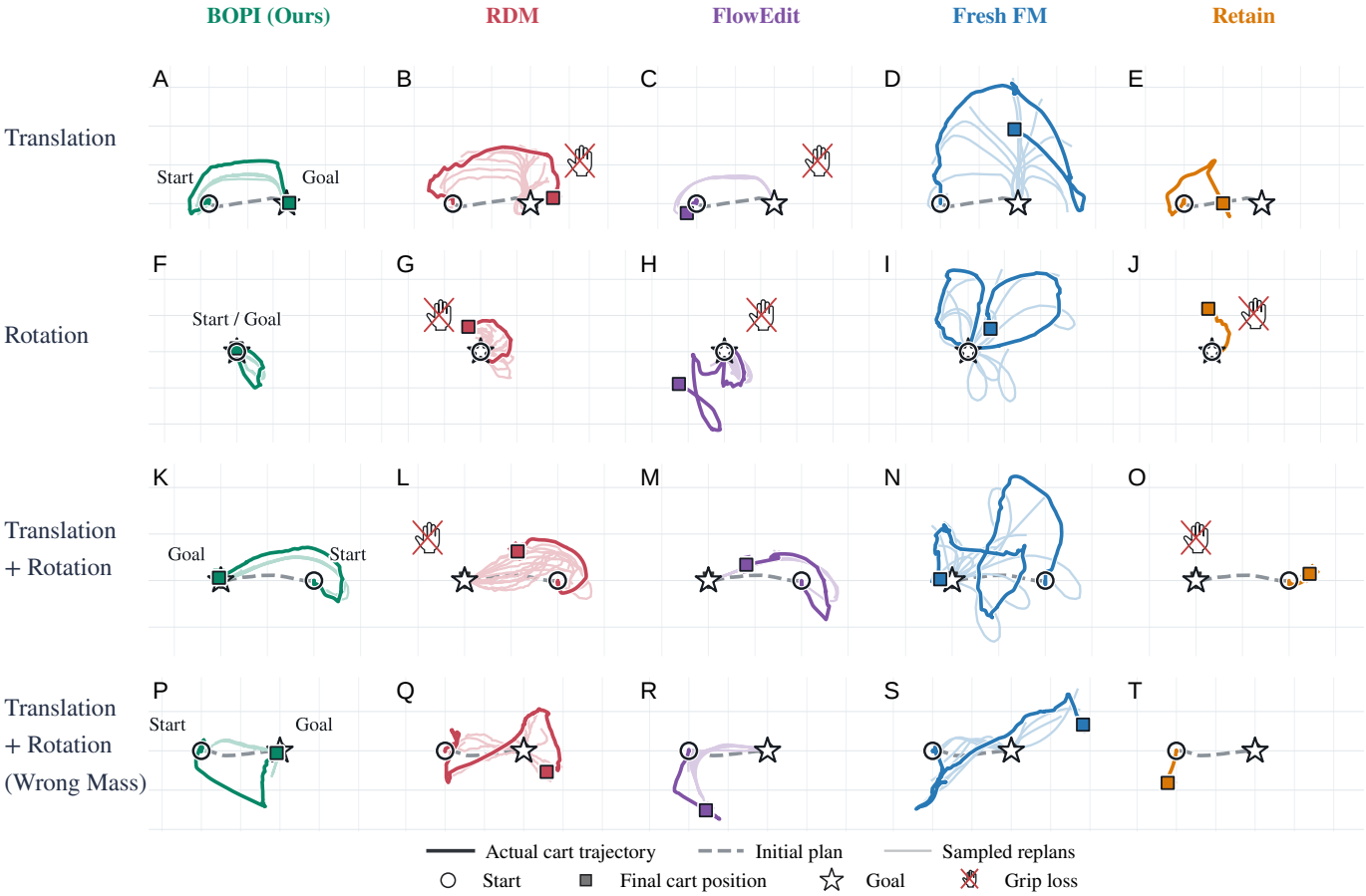

Fig. 7. **BOPI maintains goal-directed motion during replanning in simulation.** Planning examples span translation (A–E), rotation (F–J), combined translation–rotation (K–O), and combined motion with an incorrect initial mass estimate (P–T). BOPI makes smaller changes between successive plans, producing consistent replanned trajectories as the robot progresses toward the goal. This consistency helps explain the lower final position and yaw errors reported in Table II. Grid spacing is $0.5\,\mathrm{m}$ in both directions.

$0.277\,\mathrm{rad}$), while repeatedly replacing it with Fresh FM yields even larger errors ($0.936\,\mathrm{m}$ / $1.149\,\mathrm{rad}$). BOPI instead scales revision to belief agreement, achieving the lowest errors in every category ($0.179\,\mathrm{m}$ / $0.152\,\mathrm{rad}$ overall, approximately 33% / 47% below RDM and 39% / 56% below FlowEdit). Fig. 7 provides further evidence: BOPI makes smaller changes between successive plans, producing consistent replanned trajectories as the robot progresses toward the goal. Together, these results favor adapting the plan to updated estimates while limiting unnecessary changes to the path the robot is following.

5) Adaptive-repair ablation: We further test a key design choice: whether to first invert the current plan (*flow inversion*) or directly add noise, as in RDM (*forward noising*). Both use belief overlap to set the depth and reconstruct under the updated belief. Across the same 18 goals with 64 seeds, forward noising increases mean final position and yaw errors by 18.2% and 20.9%, respectively, compared with flow inversion. This suggests that BOPI benefits not only from adapting the revision depth, but also from reversing the old plan through the learned flow rather than perturbing it with fresh noise.

E. Real-robot experiments

Figure 1 (A–D) shows adaptable object-centric control across wheeled objects. We test two goals on a non-holonomic shopping cart: move one meter left (Fig. 1E–G), or turn 90°

TABLE III
REAL-ROBOT RESULTS: TRANSLATION.

Method	Position error (cm)	Yaw error (°)	Position tracking RMS (cm)	Yaw tracking RMS (°)
BOPI (Ours)	**6.48 ±3.48**	1.64 ±1.52	**7.20 ±1.41**	**3.89 ±0.59**
RDM	7.35 ±2.98	**1.17 ±1.23**	8.98 ±1.93	3.98 ±0.60
FlowEdit	30.64 ±40.41	17.33 ±31.83	16.44 ±13.47	6.73 ±5.06
Fresh FM	43.32 ±29.40	21.08 ±16.04	13.01 ±3.70	6.60 ±1.36
Retain	85.48 ±12.11	5.37 ±4.63	40.58 ±5.25	7.22 ±0.85

TABLE IV
REAL-ROBOT RESULTS: TURNING.

Method	Position error (cm)	Yaw error (°)	Position tracking RMS (cm)	Yaw tracking RMS (°)
BOPI (Ours)	**5.07 ±2.47**	4.51 ±1.63	**10.20 ±2.24**	**5.05 ±0.79**
RDM	9.83 ±3.89	**3.11 ±1.68**	16.05 ±4.51	6.43 ±0.79
FlowEdit	8.29 ±1.42	3.89 ±1.32	13.80 ±1.43	5.85 ±0.56
Fresh FM	8.80 ±2.81	4.01 ±1.50	11.71 ±1.17	5.70 ±0.49
Retain	22.98 ±16.24	7.96 ±1.90	25.15 ±6.63	7.86 ±0.52

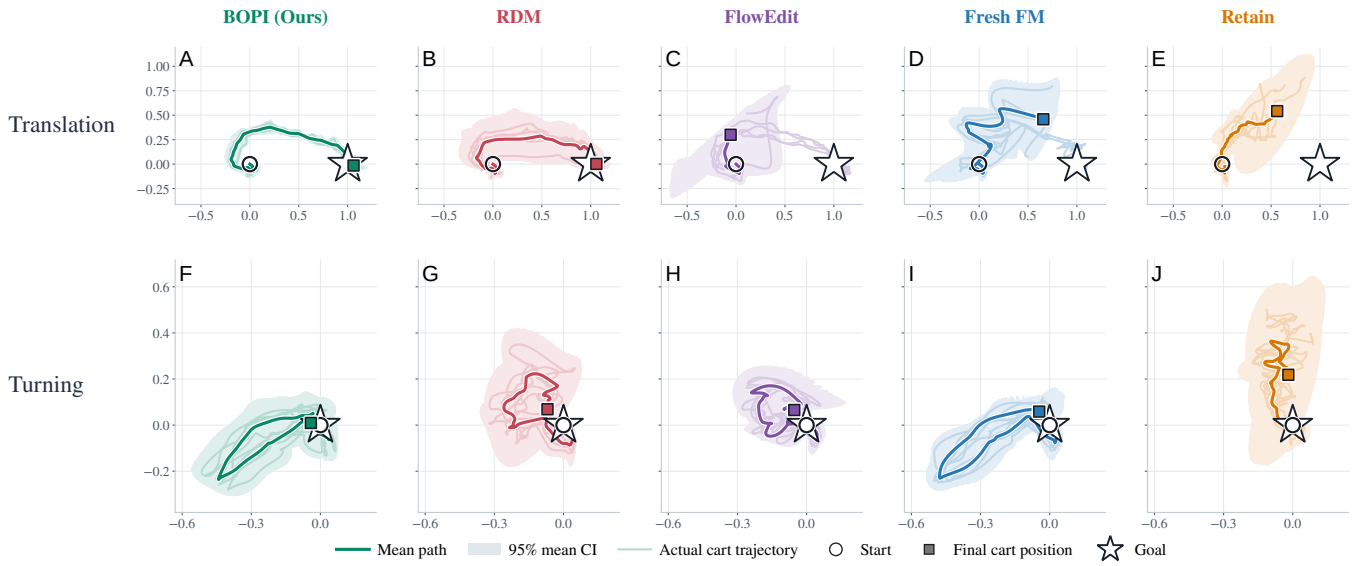

Fig. 8. **BOPI produces consistent trajectories for a non-holonomic cart in real-robot experiments.** The goals are one-meter lateral motion (A–E) and a 90° clockwise turn returning to the starting position (F–J). Corresponding BOPI photos appear in Fig. 1 (E–J). Light curves show five estimated cart trajectories; solid curves and shading show their time-aligned mean and pointwise 95% confidence regions. Squares mark the mean at each trajectory's end. All paths share display-only smoothing ($\sigma = 0.20\,\mathrm{s}$). Tables III and IV compare complete trials.

clockwise and return to the starting position (Fig. 1H–J), with five trials per method and goal. All methods start from the same plan, generated under the incorrect assumption that the rear casters swivel freely. For hardware safety, we reduce the replanning rate to 1 Hz.

Tables III and IV report mean ± sample standard deviation. Final errors measure distance and heading difference to the goal; tracking RMS measures deviation from the commanded cart trajectory. Metrics use odometry estimates.

Retaining the incorrect initial plan leads to large final position errors ($54.23\,\mathrm{cm}$ on average). Fresh FM uses updated estimates but repeatedly replaces the plan, exhibiting more variable paths (Fig. 8D,I) and excessive leaning and sideways leg extension (Fig. 9). BOPI instead scales revision to belief agreement, achieving lower mean final position error and both tracking RMS errors than RDM and FlowEdit on both tasks (mean final position error: $5.78\,\mathrm{cm}$ versus $8.59\,\mathrm{cm}$ for RDM and $19.47\,\mathrm{cm}$ for FlowEdit). Together, these results highlight the value of incorporating new evidence while limiting disruption to ongoing motion when manipulating objects with initially unknown physical properties.

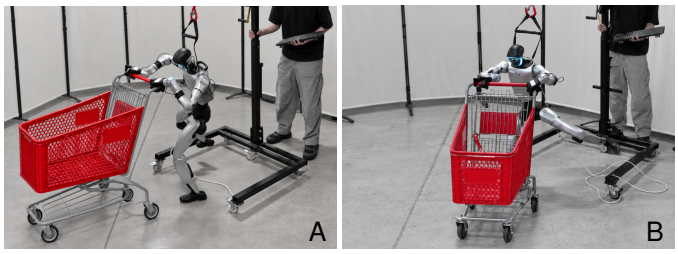

Fig. 9. **Unstable executions of the Fresh FM baseline.** Excessive leaning (A) and sideways leg extension (B) during lateral pushing illustrate how replanning can disrupt whole-body motion.

V. Conclusion

We presented an object-centric framework for humanoid cart pushing that combines online property estimation, flow-based planning, and whole-body control. BOPI, its replanning component, uses belief overlap to determine how far to reverse plan generation before reconstructing the timed waypoints under updated estimates. This lets the planner incorporate new evidence while the controller follows the object's path and adjusts the robot's body configuration. Experiments support both the repair mechanism and its use in closed-loop loco-manipulation. BOPI achieves higher success with smaller plan changes in the controlled benchmark and reduces mean final position and yaw errors by approximately 33% and 47% relative to fixed-depth RDM across 18 simulated goals. On a real non-holonomic cart, BOPI performs translation and turning with lower mean final position and tracking errors than the compared baselines. Together, these results support our central insight: effective replanning should incorporate new evidence while preserving useful parts of the current plan. Extending the framework beyond planar cart pushing to three-dimensional goals and non-wheeled objects is a next step toward humanoids that learn to manipulate any unfamiliar objects through interaction.

References

[1] S. Zhou *et al.*, "Adaptive online replanning with diffusion models," in *NeurIPS*, vol. 36, 2023.

[2] V. Kulikov *et al.*, "FlowEdit: Inversion-free text-based editing using pre-trained flow models," in *ICCV*, 2025.

[3] A. Rioux *et al.*, "Autonomous SLAM based humanoid navigation in a cluttered environment while transporting a heavy load," *Robot. Auton. Syst.*, vol. 99, pp. 50–62, 2018.

[4] J. Foster *et al.*, "Physically consistent online inertial adaptation for humanoid loco-manipulation," in *IROS*, 2024.

[5] D. Li *et al.*, "HAIC: Humanoid agile object interaction control via dynamics-aware world model," in *RSS*, 2026.

[6] K. Black *et al.*, "Real-time execution of action chunking flow policies," in *NeurIPS*, vol. 38, 2025.

[7] Y. Duan *et al.*, "Real-time iteration scheme for diffusion policy," in *IROS*, 2025.

[8] J.-P. Sleiman *et al.*, "A unified MPC framework for whole-body dynamic locomotion and manipulation," *IEEE RA-L*, vol. 6, no. 3, 2021.

[9] Y. Zhang *et al.*, "FALCON: Learning force-adaptive humanoid loco-manipulation," in *L4DC*, ser. Proceedings of Machine Learning Research, vol. 331, 2026, pp. 265–281.

[10] D. Li *et al.*, "VAIC: Vision-guided humanoid agile object interaction control via decoupled commands," arXiv:2606.09286, 2026.

[11] M. Byrd *et al.*, "AdaptManip: Learning adaptive whole-body object lifting and delivery with online recurrent state estimation," arXiv:2602.14363, 2026.

[12] Z. Gu *et al.*, "REFINE-DP: Diffusion policy fine-tuning for humanoid loco-manipulation via reinforcement learning," arXiv:2603.13707, 2026.

[13] M. Mittal *et al.*, "Articulated object interaction in unknown scenes with whole-body mobile manipulation," in *IROS*, 2022.

[14] Z. Luo *et al.*, "Omnigrasp: Grasping diverse objects with simulated humanoids," in *NeurIPS*, vol. 37, 2024.

[15] C. Tessler *et al.*, "MaskedManipulator: Versatile whole-body control for loco-manipulation," in *SIGGRAPH Asia*, 2025, pp. 1–11.

[16] S. Zhao *et al.*, "ResMimic: From general motion tracking to humanoid whole-body loco-manipulation via residual learning," arXiv:2510.05070, 2025.

[17] R. Yu *et al.*, "OmniContact: Chaining meta-skills via contact flow for generalizable humanoid loco-manipulation," in *CoRL*, 2026.

[18] K. Harada *et al.*, "Walking motion for pushing manipulation by a humanoid robot," *J. Robot. Soc. Japan*, vol. 22, no. 3, pp. 392–399, 2004.

[19] J. Scholz *et al.*, "Cart pushing with a mobile manipulation system: Towards navigation with moveable objects," in *ICRA*, 2011, pp. 6115–6120.

[20] L. Hawley *et al.*, "Control strategy and implementation for a humanoid robot pushing a heavy load on a rolling cart," in *IROS*, 2017, pp. 4997–5002.

[21] Y. Sun *et al.*, "On-line parameter identification of a cart by mobile manipulation pushing," *Robot. Auton. Syst.*, vol. 46, no. 1, pp. 29–46, 2004.

[22] Z. Zhang *et al.*, "Integrating maneuverable planning and adaptive control for robot cart-pushing under disturbances," arXiv:2506.18410, 2025, version 2, revised March 30, 2026.

[23] P. Sriganesh *et al.*, "Action-informed estimation and planning: Clearing clutter on staircases via quadrupedal pedipulation," in *ICRA*, 2026.

[24] Open X-Embodiment Collaboration, "Open X-Embodiment: Robotic learning datasets and RT-X models," in *ICRA*, 2024, pp. 6892–6903.

[25] C. Yu *et al.*, "Multi-embodiment legged robot control as a sequence modeling problem," in *ICRA*, 2023, pp. 7250–7257.

[26] T. He *et al.*, "OmniH2O: Universal and dexterous human-to-humanoid whole-body teleoperation and learning," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 270, 2025, pp. 1516–1540.

[27] Q. Ben *et al.*, "HOMIE: Humanoid loco-manipulation with isomorphic exoskeleton cockpit," in *RSS*, 2025.

[28] L. Yang *et al.*, "OmniRetarget: Interaction-preserving data generation for humanoid whole-body loco-manipulation and scene interaction," in *ICRA*, 2026.

[29] W. Yu *et al.*, "Preparing for the unknown: Learning a universal policy with online system identification," in *RSS*, 2017.

[30] A. Kumar *et al.*, "RMA: Rapid motor adaptation for legged robots," in *RSS*, 2021.

[31] G. Puthumanaillam *et al.*, "Belief-conditioned one-step diffusion: Real-time trajectory planning with just-enough sensing," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 305. PMLR, 2025, pp. 68–92.

[32] M. Janner *et al.*, "Planning with diffusion for flexible behavior synthesis," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 162, 2022, pp. 9902–9915.

[33] C. Chi *et al.*, "Diffusion Policy: Visuomotor policy learning via action diffusion," in *RSS*, 2023.

[34] Y. Lipman *et al.*, "Flow matching for generative modeling," in *ICLR*, 2023.

[35] X. Liu *et al.*, "Flow straight and fast: Learning to generate and transfer data with rectified flow," in *ICLR*, 2023.

[36] K. Nguyen *et al.*, "FlowMP: Learning motion fields for robot planning with conditional flow matching," in *IROS*, 2025, pp. 11 291–11 297.

[37] C. Meng *et al.*, "SDEdit: Guided image synthesis and editing with stochastic differential equations," in *ICLR*, 2022.

[38] J. Song *et al.*, "Denoising diffusion implicit models," in *ICLR*, 2021.

[39] X. Feng *et al.*, "Denoising tells when to replan: Denoising-variance adaptive chunking for flow-based robot policies," arXiv:2606.03847, 2026.

[40] J. Lu *et al.*, "UniSteer: Unified noise steering for efficient human-guided VLA adaptation," arXiv:2605.10821, 2026.

[41] A. Bhattacharyya, "On a measure of divergence between two multinomial populations," *Sankhyā: The Indian Journal of Statistics*, vol. 7, no. 4, pp. 401–406, 1946.

[42] F. Nielsen, "Revisiting Chernoff information with likelihood ratio exponential families," *Entropy*, vol. 24, no. 10, p. 1400, 2022.

[43] Y. Wu *et al.*, "Group normalization," in *ECCV*, 2018, pp. 3–19.

[44] E. Perez *et al.*, "FiLM: Visual reasoning with a general conditioning layer," in *AAAI*, 2018.

[45] J. Schulman *et al.*, "Proximal policy optimization algorithms," arXiv:1707.06347, 2017.

[46] K. Zakka *et al.*, "mjlab: A lightweight framework for GPU-accelerated robot learning," arXiv:2601.22074, 2026.

[47] J. Suhardjo *et al.*, "Non-linear optimal control of a Duffing system," *Int. J. Non-Linear Mech.*, vol. 27, no. 2, pp. 157–172, 1992.