GOLEM: Modular Humanoid Autonomy Towards Electric Vehicle Battery Disassembly

Anonymous submission

***Abstract*— Disassembling end-of-life electric vehicle (EV) battery packs is dull and dangerous work, performed almost entirely by humans. We present GOLEM (Generalized Open Library of Embodied Modules), an end-to-end, open-source system architecture for EV battery disassembly with the Unitree H1-2 humanoid robot in which walking, manipulation, dynamic stability, navigation, and spatial memory are independent modules with abstract interfaces, so that methods are easily developed, interchanged, and compared. GOLEM is deployed as a Docker-based ROS 2 abstraction in which MuJoCo and IsaacLab digital twins expose interfaces matching the physical robot. GOLEM's composability and per-module customization enable development and demonstration of humanoid EV battery disassembly, from simulation to reality. GOLEM provides fair comparison between humanoid modules, enabling evaluation as a capability ladder, in which one module is characterized at a time and added as a rung: LiDAR-inertial navigation places the robot within 13.0cm of a 6m goal; a learned standing controller recovers from external disturbances that sampling-based lower-body MPC does not; and grasping loosened fasteners from a real Hyundai Ioniq 5 pack degrades from 97% tethered to 87% free-standing to 37% under navigation-induced pose variance. Source code is available at the project page `https://golem-humanoid.github.io`.**

I. INTRODUCTION

End-of-life electric vehicle lithium-ion batteries (EV-LIBs) must be disassembled so that precious minerals and metals therein can be recovered for reuse [1]. This work is repetitive and hazardous: hundreds of fasteners must be removed from energy-dense packs that arrive in an unknown state of charge, can combust, emit toxic fumes, or electrocute workers [2].

Dedicated automation [2]–[4] is not versatile enough for the large variety of pack designs. Furthermore, recycling facilities are currently designed around humans positioning humanoid robots, which integrate without special accommodations such as ramps or fixed mounts, as appealing candidates. Proving humanoid efficacy in such hazardous industrial tasks represents a step toward more sensitive domains such as the home, healthcare, or disaster response. While recent advances in learned whole-body control [5]–[7] have significantly expanded humanoid capabilities, existing end-to-end paradigms remain ill-suited for industrial EV disassembly. First, these policies primarily target smaller platforms such as the Unitree G1 (1.32 m, 35 kg, 120 Nm knee torque). For disassembly of the 400kg, 1.2m by 2m Hyundai Ioniq 5 battery pack in a human-oriented or shared-autonomy work environment such as in Fig. 1, full-scale humanoids like the Unitree H1-2 with larger workspaces and torque envelopes (1.78 m, 70 kg, 360 Nm) are required.

We present GOLEM (Generalized Open Library of Embodied Modules), an open-source, end-to-end system architecture for the Unitree H1-2 humanoid evaluated on simulated RoboCasa kitchen [8] and sim-to-real EV-LIB disassembly tasks. Within GOLEM, walking, manipulation, dynamic stability, and navigation are independent modules which can connect to arbitrary control policies. GOLEM also provides digital twins in MuJoCo [8], [9] and IsaacLab [10], both isolated and communicating identically to the real robot via a Linux and MacOS-compatible ROS2 Docker container, thereby affording flexible switching and evaluation between various methods to compose and verify task-appropriate humanoid autonomy.

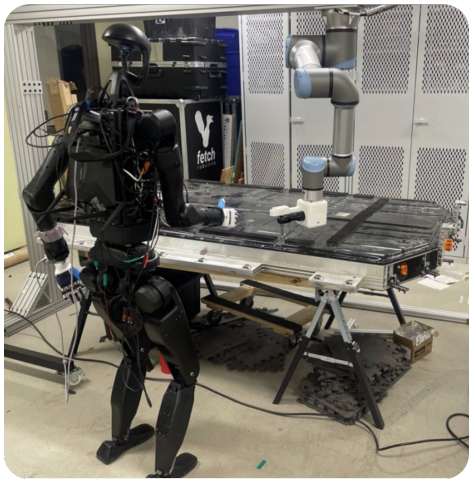


Fig. 1: A Unitree H1-2 humanoid working alongside the gantry-based RAPID disassembly cell [2] on a Hyundai Ioniq 5 battery pack. GOLEM augments automation infrastructure designed around human workers with humanoid autonomy.

The contributions of this paper are twofold:

1) We demonstrate humanoid fastener removal from a Hyundai Ioniq 5 battery pack at three levels of autonomy: tethered, standing, and standing with pose variance post-navigation, characterizing module interplay and where simulation fidelity fails to predict real-world performance.
2) We introduce GOLEM, an open-source, modular humanoid development platform wherein capabilities are made interchangable on matching simulated & real interfaces, enabling fair comparison of arbitrary methods.

Finally, we perform a case study implementing multiple Language-Conditioned Spatial Memory methods and use GOLEM's modular structure and common interfaces as a scaffold to orchestrate our comparison.

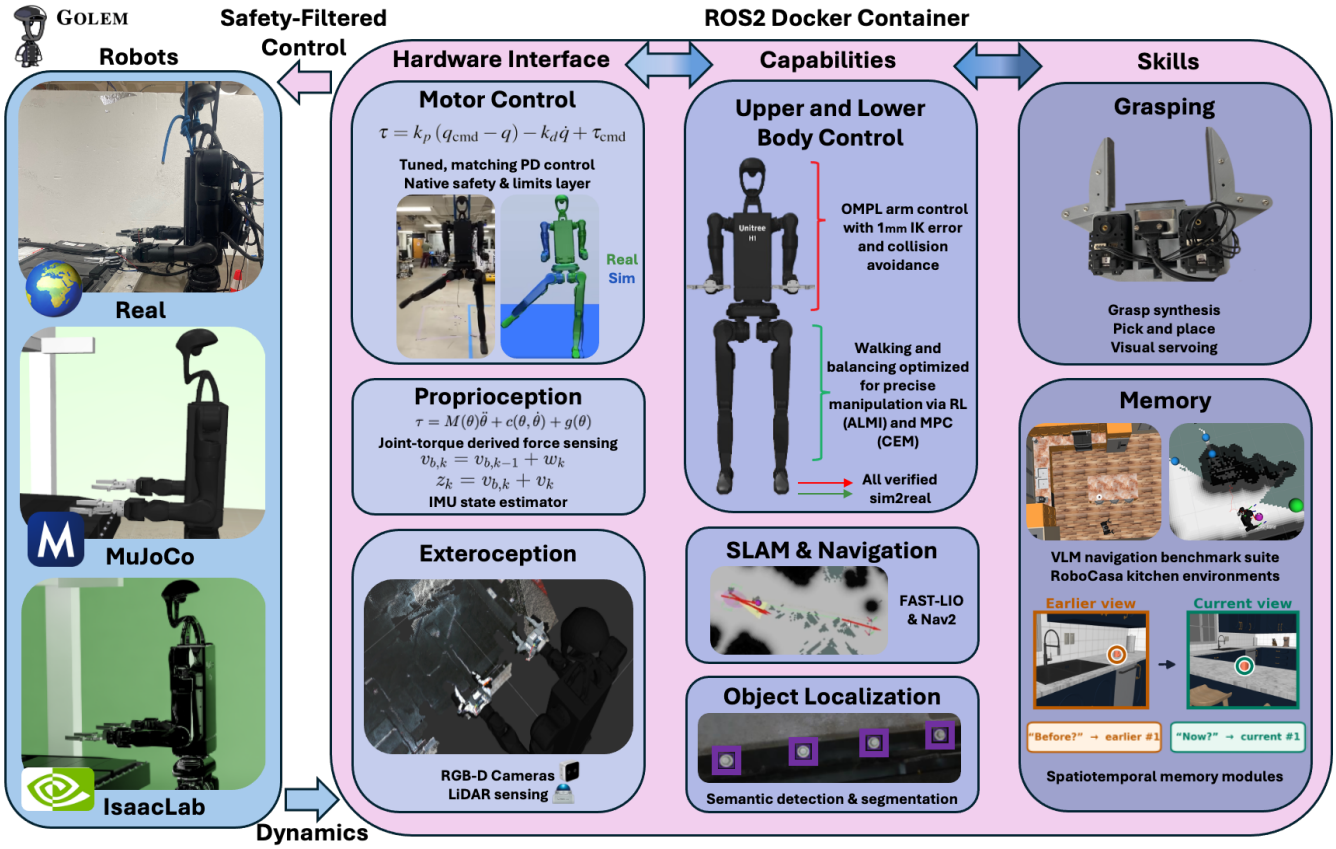


Fig. 2: GOLEM is an open-source development platform for industrial humanoid robots, designed for transfer to real-world electric-vehicle (EV) battery disassembly. The system is deployed as a ROS2 Docker container with **four interoperable, modular layers**, providing a full-stack application suite for the Unitree H1-2 humanoid. GOLEM connects **(1)** three physics engines—the real world, MuJoCo, and PhysX via IsaacLab—with **(2)** hardware interfaces for motor control, proprioception, and exteroception, **(3)** core capabilities spanning learning & planning-based bodily control, navigation, and semantic object localization, and **(4)** skills for contact-rich & precise manipulation and retrieval with spatial memory modules. The platform supports simulation evaluation on RoboCasa kitchen [8] & Hyundai Ioniq 5 EV battery environments and is compatible with Linux and MacOS devices (sans elements that require CUDA), enabling broad accessibility and collaboration.

II. RELATED WORK

A. Robotic Disassembly of EV Batteries

Disassembling EV battery packs enables material recovery [1] but remains largely manual; robotic work targets the smaller class of hybrid packs [3], [11] or task planning [4]. RAPID [2] addresses full-size 800V packs and solves the unscrewing task (fastener operations account for about 75% of manual disassembly labor). Then, fastener removal, also known as peg-hole separation [12] or peg-hole disassembly [13], is a difficult task that requires precise control and benefits from compliance. While RAPID can be extended to also remove screws, such a system requires disassembly factories to be designed around it. A humanoid is complementary to the RAPID system and infrastructure in facilities designed for human workers, and we demonstrate GOLEM removing screws from a commercial EV pack.

B. Learned Whole-Body Control and Loco-Manipulation

Reinforcement learning and large-scale motion tracking have produced increasingly capable whole-body controllers. OmniH2O [14] and HumanPlus [15] learn whole-body policies from human motion, ALMI [7] adversarially couples a lower-body locomotion policy with an upper-body motion-tracking policy on the full-size Unitree H1-2, SONIC [5] scales motion tracking into a generalist controller for the Unitree G1, and LEGS [6] leverages photo-realistic gaussian splatting to transfer loco-manipulation policies trained in simulation to a real Unitree G1. On the H1-2, Humanoid-COA [16] and Calvert [17] demonstrate compositional loco-manipulation behaviors on the Unitree H1-2 with embodied LLM reasoning and behavior trees. Unlike a single method or black-box policy, GOLEM enables learned, model-based, and other various controllers to be wrapped as interchangeable modules and compared against their counterparts. We demonstrate this with MuJoCo-MPC [18] and ALMI [7] in Sec. IV.

C. Middleware, Simulation, and Modular Software

ROS 2 provides communication, hardware abstraction, and reusable packages for modular robot software [19]; MuJoCo enables physics-based evaluation and model-based control [9], [18]; RoboCasa provides large-scale simulated environments for everyday tasks [8]; and IsaacLab facilitates high-fidelity synthetic sensor data generation in GPU-

accelerated environments [10]. Currently individual capabilities are built upon specialized tools: Open Motion Planning Library (OMPL) for upper-body motion planning [20], Nav2 for navigation [21], Pinocchio for rigid-body dynamics [22]. Typically users must hand-craft and assemble these tools and capabilities into a complete control stack; GOLEM reduces engineering labor via its shared abstract interfaces and accelerates system-level comparison with an end-to-end architecture, as well as tool & capability addition. Finally, GOLEM empowers developers and broad collaboration with accessible MuJoCo and IsaacLab digital twins which expose ROS 2/DDS interfaces matching the physical robot.

III. System Architecture

Figure 2 gives an overview of GOLEM. We describe its four layers in turn: the container-based robots (Sec. III-A), the hardware interfaces for motor control and perception (Sec. III-B), the core capabilities of control, navigation, and detection (Sec. III-C), and the grasping and memory skills (Sec. III-D). Throughout, $q \in \mathbb{R}^{d=27}$ denotes the robot's joint configuration, $a^l \in \mathbb{R}^{d_l=12}$ and $a^u \in \mathbb{R}^{d_u=15}$ the commanded lower- and upper-body joint positions.

A. Container-based robot architecture

GOLEM is implemented as a set of Docker containers that communicate over a single ROS 2 domain with CycloneDDS. A simulation container optionally runs the digital twin (MuJoCo or IsaacLab), which publishes matching interfaces of the physical robot. Currently for IsaacLab, GOLEM provides verified support only for hardware interfaces (Sec. III-B).

A second container holds the ROS 2 workspace comprising a bringup that launches locomotion, differential inverse kinematics, perception servers, SLAM, navigation, safety monitoring, and the skill servers that expose capabilities such as grasping and exploration as ROS 2 actions.

Since the simulator reproduces the real robot's interfaces exactly, switching between simulation and hardware amounts to selecting the DDS domain. We emphasize that the twin's value lies in this interface equivalence rather than in dynamic fidelity: contact-rich interactions such as screw engagement are not faithfully reproduced in simulation, which we observe in our sim-to-real experiments.

B. Hardware Interfaces

Safety: All motion commands a^l, a^u pass through a low-level safety controller that runs independently of the control modules. The controller continuously monitors joint positions, velocities, and torques against configured limits and immediately stops the robot when any limit is exceeded, for example if a motor experiences an unexpected torque or a state estimate becomes stale. As limits, we are using 90% of the maximum values specified by Unitree.

In addition, the safety controller constantly polls an Arduino Uno microcontroller connected to a physical emergency-stop button, so a human supervisor can halt the robot at any time. Loss of communication with the Arduino is itself treated as a stop condition. Since the safety controller subscribes to the same interfaces in simulation and on hardware, safety behavior can be exercised and tested in the digital twin before deployment.

Motor Control: Each motor controller commands the desired joint position q_{cmd} and the gravity-compensation torque τ_g to the underlying PD servo loops running at 500 Hz in the motor firmware. The desired joint velocity is fixed at $\dot{q}_{\text{cmd}} = 0$, causing the derivative term to act as velocity damping that suppresses oscillation. The firmware therefore computes the actuator torque as

$$\tau = k_p\,(q_{\text{cmd}} - q) - k_d \dot{q} + \tau_{\text{cmd}} \tag{1}$$

with $\tau_{\text{cmd}} = \tau_g$. To reject steady-state position errors caused by model inaccuracies and unmodeled loads, the high-level controller maintains an additional integral bias torque τ_b. At each time step, the bias torque is updated according to

$$\tau_b \leftarrow \tau_b + k_i\,(q_{\text{cmd}} - q)\,\Delta t. \tag{2}$$

The bias torque is added to the gravity-compensation term, yielding the total commanded torque $\tau_{\text{cmd}} = \tau_g + \tau_b$.

MAGPIE Hands: The H1-2 is equipped with parallel jaw MAGPIE [23] hands, a low-cost, force-controlled parallel gripper with built-in 3D perception [23]. Two independently torque-controlled servomotors (Dynamixel AX-12A) drive the fingers through four-bar linkages, providing grasp forces of up to 32N controllable in increments of 0.08N, contact detection, and compliant off-center grasping. This force-controlled architecture also supports integration with semantic physical reasoning frameworks like DeliGrasp [24] to modulate applied grasp forces based on LLM-inferred material properties. A palm-mounted Intel RealSense D405 observes the object, minimizing occlusion compared to head-mounted cameras and improving fine-grained resolution compared to wrist-mounted cameras. The gripper weighs 414 g, is built from 3D-printed and commercial off-the-shelf parts for approximately $460, and its hardware and software are open source. The MAGPIE hands replace the Inspire RH56DFX six-DoF dexterous hands typically paired with the H1-2, since they require significant reorientation and translation to pinch in-place, causing instability [25].

1) Proprioception: The Unitree H1-2's 27 joints are actuated by high-torque, quasi-direct-drive motors, enabling direct external force observation through FOC current sensing. The lower-body knee, hip, torso, and ankle joints have a torque limit of 360, 220, 220, and 75 Nm, respectively, and the upper-body shoulder, elbow, and wrist joints have a limit of 120, 120, and 30 Nm, respectively [26].

Base State Estimation: Without ground truth center-of-mass (CoM) measurements we must infer quantities like CoM velocity from the IMU orientation $\boldsymbol{R}_b$, angular rate $\boldsymbol{\omega}_b$, and joint encoders $\boldsymbol{q}, \dot{\boldsymbol{q}}$; so we track the base velocity $\boldsymbol{v}_b \in \mathbb{R}^3$ with a random-walk Kalman filter,

$$\boldsymbol{v}_{b,k} = \boldsymbol{v}_{b,k-1} + \boldsymbol{w}_k, \qquad \boldsymbol{z}_k = \boldsymbol{v}_{b,k} + \boldsymbol{v}_k, \tag{3}$$

whose measurement is the *planted-foot constraint* [27]: in which each foot contact k gives

$$\boldsymbol{z}_k = -\Big[\boldsymbol{\omega}_b \times \big(\boldsymbol{p}_k(\boldsymbol{q}) - \boldsymbol{p}_b\big) + \boldsymbol{R}_b\,\boldsymbol{J}_k(\boldsymbol{q})\,\dot{\boldsymbol{q}}\Big], \tag{4}$$

with $\boldsymbol{J}_k$ the translational Jacobian of contact k (one forward-kinematics evaluation). Feet are fused with a standard Kalman update using contact-weighted, outlier-gated measurement noise [28]; orientation comes from the IMU, whereas horizontal position, unobservable from proprioception [27], [29], is integrated open-loop.

2) Exteroception: The exteroceptive suite is designed to handle navigation, approach, and precise manipulation at variable distances. For navigation, SLAM and obstacle avoidance are handled by a head-mounted Livox MID-360 LiDAR. On the approach, mid-range object localization and VLM perception rely on the robot's stock forward-facing Intel RealSense D435i camera. Then, for precise manipulation, hand-mounted RealSense D405 cameras, specialized for sensing between 7 cm and 50 cm, facilitate closed-loop visual servoing to reduce accumulated forward-kinematic and head camera calibration errors.

C. Capabilities

1) Lower-Body Control: This module exposes a common interface for the H1-2's 12 leg joints, mapping commanded base twists $u_{\text{base}} = (v_x, v_y, \omega_z)$ to lower body commands a^l for the servo layer. The waist and arms remain controlled by the IK stack (Sec. III-C.2). Input twists are clamped to the active controller's valid envelope, and standing is handled implicitly as a zero-twist command. We integrate three controllers, which are wrapped in the GOLEM interface for fair comparison:

1) **MJPC [18]** A sampling-based model predictive controller planning at 67 Hz with a 1s planning horizon.
2) **ALMI [7]:** A learned policy running at 50 Hz. Stance behavior emerges implicitly from reward terms gated on $\|u_{base}\| < 0.1$.
3) **Vendor Policy:** Unitree's default black-box controller.

Model-Based Lower-Body Control: We formulate lower-body locomotion and dynamic stabilization as a finite-horizon optimal control problem solved in receding-horizon fashion using MuJoCo-MPC (MJPC) [18]. Let $x_t = (q_t, \dot{q}_t, b_v)$ denote the whole-body state and base velocity; at each control time step, the planner solves

$$\min_{a^l_{0:T}} \sum_{t=0}^{T} \ell(x_t, a^l_t) \quad \text{s.t.} \quad x_{t+1} = f(x_t, a^l_t, \Delta t), \tag{5}$$

where f is MuJoCo's contact dynamics [9] and ℓ is a weighted sum of task-residual norms over objectives such as center-of-mass height and velocity, torso orientation, foot placement, and control effort. In MJPC, we employ predictive sampling (CEM) [30] to optimize Eq. (5) in real time, warm-starting each step by the time-shifted plan.

Learning-Based Lower-Body Control (ALMI): ALMI [7] formulates locomotion training as a two-player zero-sum Markov game between the lower-body policy π^l and an adversarial upper-body policy π^u sharing state s_t comprised of proprioception, commanded velocity, and phase parameter. To ensure robust velocity tracking under dynamic upper-body disturbances, π^l maximizes the command-following return r^l while π^u acts as an adversary seeking to minimize it:

$$\max_{\pi^l} \min_{\pi^u} \mathbb{E}\left[\sum_{t=0}^{T} r^l(s_t, a^l_t, a^u_t)\right] \tag{6}$$

During locomotion training, π^u generates adversarial actions $a^u_{\text{adv}} \sim \pi^u(\cdot \mid s^{\text{prop}}_t, g^u)$ driven by reference posture commands g^u sampled via a dual-difficulty curriculum. This min-max formulation hardens π^l against dynamic recoil, payload swings, and momentum shifts. In GOLEM, we instantiate only the 12-DoF leg policy π^l through the lower-body control interface, and replace π^u with our upper-body controller (Sec. III-C.2).

2) Upper-Body Control: Upper-body motion is driven by desired end-effector poses. The upper-body controller interface expects two end-effector poses in $SE(3)$ and outputs upper-body motor commands a^u to achieve these poses. Upper-body motion generation combines differential inverse kinematics with sampling-based motion planning. GOLEM integrates direct differential inverse kinematics for short, unobstructed motions and sampling-based planning for larger motions that require collision avoidance. For efficient collision checking, GOLEM uses MorphIt [31] to construct a packed-sphere approximation of the robot's upper body, reducing the median collision-checking time from 38.8 μs with the mesh representation to 5.8 μs (6.7$\times$ speedup).

Differential Inverse Kinematics: End-effector goals are tracked by a task-space controller implemented with Pink [32]. Each task i defines a residual $r_i(q)$, e.g., the error between the current and target end-effector pose, whose Jacobian $J_i(q) = \partial r_i / \partial q$ is computed with Pinocchio [22]. At each control step, the controller solves the quadratic program

$$\min_{\dot{q}} \sum_i w_i \left\| J_i(q)\,\dot{q} - \alpha_i r_i(q) \right\|^2 + \lambda \|\dot{q}\|^2 \text{ s.t. } \dot{q}^- \le \dot{q} \le \dot{q}^+, \tag{7}$$

where the w_i weight competing tasks, α_i are task gains, λ regularizes the solution, and the box constraints enforce joint position and velocity limits. The optimizer $\dot{q}^\star$ is integrated as $q \leftarrow q + \dot{q}^\star \Delta t$ to produce a^u.

Sampling-Based Planning: For longer-range motions in cluttered environments, we use OMPL's implementation of RRT-Connect [33] to generate collision-free paths in the 15-DOF upper-body joint space. The sampled configurations are validated against joint limits, self-collision, and configurable workspace constraints. For battery disassembly, the position of each grasp frame is constrained to a fixed Cartesian region in front of the robot to limit disturbances to the balance:

$$x \le 0.60\,\text{m}, \quad y \in [-0.45, 0.45]\,\text{m}, \quad z \in [-0.05, 0.60]\,\text{m}. \tag{8}$$

3) Object Localization: GOLEM's modular architecture easily incorporates any bounding box or segmentation mask producing model. The GOLEM interface expects object detection models to output point clouds of requested objects from RGB-D cameras. In generic scenes like RoboCasa we leverage internet-scale VLMs like Gemini Robotics [34], and for niche battery components, we fine-tune YOLO-World [35] on battery components such as screws, nuts, and bus

bars. RGB detections are applied to the RGB-D point cloud, where they can be consumed by higher-level skills.

4) Navigation: Navigation builds on a ROS 2 & Nav 2 stack [21]. FAST-LIO [36] provides LiDAR-inertial odometry and mapping from the robot's Livox MID-360; the resulting point cloud is projected into a 2D occupancy grid, while the full 3D cloud is incorporated into the costmap for collision checking. Nav2's global path planner performs a graph search over this costmap: writing $C : \mathcal{G} \to [0, c_{\max}]$ for the cost of traversing grid cell $g \in \mathcal{G}$, it finds a cell path $g_{0:N}$ from the robot's cell to the goal cell minimizing

$$\sum_{k=0}^{N-1} \Big(\|g_{k+1} - g_k\| + \beta\, C(g_{k+1}) \Big), \tag{9}$$

using A* search with an admissible Euclidean heuristic, where β trades path length against clearance from obstacles. A local controller tracks the resulting path and emits velocity commands to the general lower-body controller interface.

D. Skills

Skills are the highest level of abstraction in GOLEM; skills orchestrate capabilities through their well-defined interfaces and enable completion of high level objectives.

Grasping: To evaluate grasping within the GOLEM architecture, we implement manipulation modules with a standardized skill interface. We compare multiple grasp synthesis strategies within the same architecture: (i) an open-world pipeline in which an object is detected via the object localization capability III-C.3, from which the segmented object's point-cloud can be passed as input to neural six-DoF grasp pose synthesis methods such as GraspGenX (GGX) [37] or classical methods such as antipodal grasp sampling; and (ii) a top-down visual-servoing approach that aligns the palm-mounted or otherwise wrist-mounted camera with the target before closure.

Spatiotemporal Memory: Spatiotemporal memory expands GOLEM's spatial understanding beyond the robot's current field of view [38]. An agent can query an object or region that is not presently visible, navigate to the returned location in memory, and then use local perception and manipulation skills. GOLEM uses VLMaps [39] to back-project dense visual-language features from posed RGB-D observations and fuse them into a persistent top-down semantic map. Following the retrieval-augmented memory architecture, ReMEmbR [40], we also include a backend adapted to store SigLIP image embeddings together with their poses and timestamps. An optional VLM re-ranks the retrieved candidates based on the query and their temporal metadata.

During memory retrieval, GOLEM accepts a free-text query referring to an object and returns candidate locations, relevance scores, and backend metadata. Exploration, navigation, and agentic programs can therefore access spatial and temporal information through a common interface, independent of the underlying memory implementation.

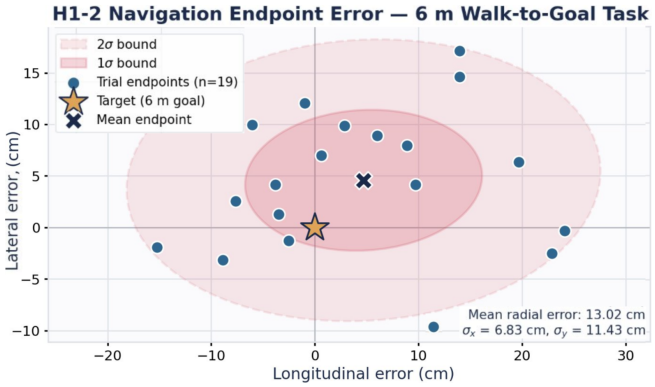


Fig. 3: Six meter start-to-goal walking accuracy using the FAST-LIO and Nav2 navigation stack to localize from noisy initialization with an off-the-shelf Unitree walking policy.

IV. EXPERIMENTS

We structure the evaluation as a capability ladder, enabled by GOLEM: each experiment introduces one additional module, quantifies the error it contributes, and passes that error on to the next. We first establish navigation accuracy under interchangeable walking policies (Sec. IV-A), then the robot's ability to hold position against unmodeled forces (Sec. IV-B), and finally we evaluate grasping of Hyundai Ioniq 5 battery pack screws at three levels of autonomy that successively integrate these modules (Sec. IV-C).

A. Walking Policy and Navigation Accuracy

The Unitree-provided H1-2 walking policy wrapped in GOLEM's velocity interface subscribes to GOLEM's FAST-LIO and Nav2 stack and is evaluated on a 6 m straight-line point-to-point navigation task over 20 trials, reaching the goal within a mean radial error of 13.0 cm (σ_x=6.83 cm, σ_y=11.43 cm) on 19 of 20 runs (95%), shown in Fig. 3, with the failed trial veering several meters off due to policy instability. This endpoint error defines the base-placement variance that downstream manipulation must tolerate when the robot walks to its work station (Sec. IV-C). Preliminary testing of ALMI for walking demonstrated significant inconsistency in stopping and stabilizing to a standing stance, precluding it from further evaluation.

B. Force-Adaptivity

We compare the force-adaptivity of two standing policies, adversarial RL-based ALMI and the lower-body MPC. Preliminary testing of the vendor-provided policy showed an inability to stand still, precluding it from standing evaluation and motivating policy switch-over to a distinct standing policy. Pushes are applied manually at the pelvis with a load-cell mounted instrumented rod (Fig. 4). Mean peak per-trial force is similar between MJPC (20.4 ± 7.0 N) and ALMI (22.1 ± 6.9 N), and in both controllers the CoM is driven outside the support polygon by the push. However, MJPC decreases monotonically to a -0.166 m margin with $16.3 \pm 6.3^\circ$ peak pitch-axis lean, requiring an operator e-stop at 4.68 ± 2.86 s. In contrast, ALMI reaches a minimum margin of -0.044 m with $6.8 \pm 1.4^\circ$ peak pitch-axis lean and notably

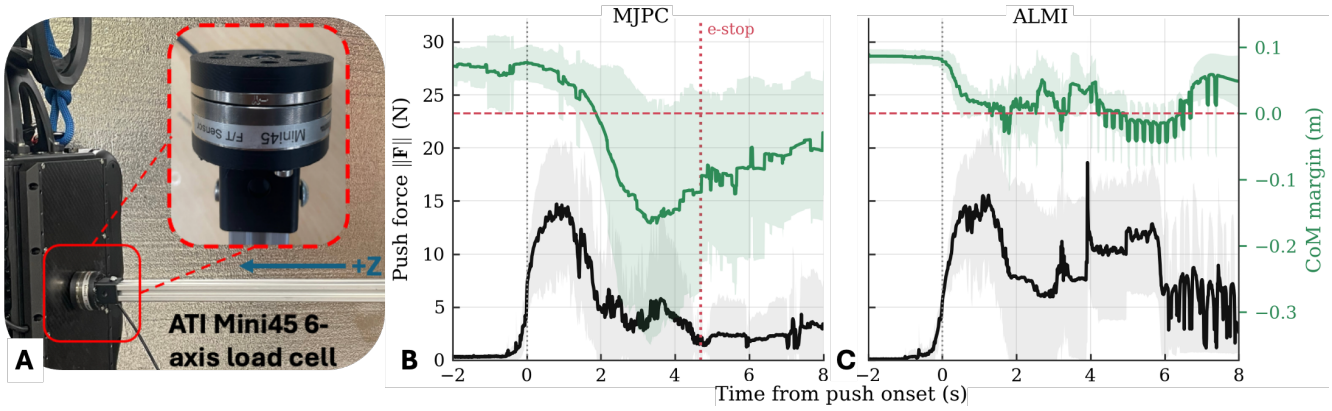


Fig. 4: **Standing with force adaptivity, MPC vs. RL** We evaluate standing robustness with **(a)** a pushing rod mounted with a ATI Mini45 six-axis load cell applying an oppositional force to the robot torso, **(b-c)** indicated in **black** until the robot loses balance, measured by the center-of-mass (CoM) margin from the support polygon (green) becoming negative (horizontal red line). Both standing methods exhibit similar initial force adaptivity, but **(b)** sampling-based model predictive control (MJPC) is not able to recover from the initial loss in balance and must be emergency stopped (vertical red line). In comparison, **(c)** adversarial reinforcement learning (ALMI) exhibits a recovery behavior to restore a positive CoM margin.

recovers a positive margin in 93% of trials (14/15). Due to this desirable recovery behavior and robustness, subsequent experiments deploy ALMI for lower-body standing.

C. Manipulation

We evaluate the five following grasp synthesis baselines (Fig. 5a): Centroid grasps the segmented object centroid with a fixed top-down or forward approach; Antipodal samples antipodal pairs from the point cloud; GGX samples grasps from GraspGenX [37]; GGX-F reranks GGX candidates by wrist reachability; C-VS closes the loop on the centroid with visual servoing. We select two task domains: grasping and opening of a fridge handle in RoboCasa kitchens and grasping of screws loosened by the RAPID gantry [2] from the Hyundai Ioniq 5 pack. Finally, we evaluate our methods on three autonomy levels that successively integrate the modules validated above: (a) *tethered*, with the robot supported by a hoist so that its legs are unloaded, isolating the perception and manipulation modules from balance; (b) *standing*, with the robot balancing at a precomputed screw-reachable base pose, adding the standing controller of Sec. IV-B; and (c) *standing with pose variance*, with the robot first navigating to the pack using the stack of Sec. IV-A, adding its base-placement error to the grasping problem. The difference in success rate between consecutive conditions isolates the contribution of each module to overall task failure. In Fig. 5b, stability sensitivity to grasp methods is measured by center-of-mass XY-projected signed distance to the H1-2 support polygon boundary (CoM margin).

Fridge opening (sim). Under tether, centroid (100%) and GGX-F (96.7%) methods are effective, while antipodal (63.3 %) and GGX (53.3 %) are less reliable. Standing incurs a further 20.0, 26.7, 0.0, and 36.6% success reduction, respectively (Fig. 5a). Due to centroid grasping generating a single, stereotyped grasp in each trial, it is the most stable method with a CoM margin distribution (median 115.8 mm, IQR 95.0–119.3 mm), 2.3–9.0× narrower than any pose-sampling method, and only 13.3 % of its episodes push the projected CoM outside the support polygon, against 26.7 % (GGX-F) and 30.0 % (antipodal).

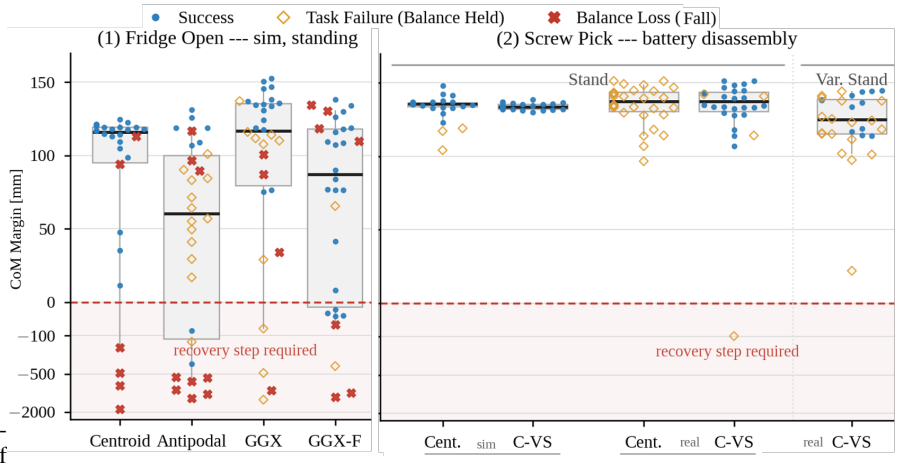
Screw picking (sim). On the screw, both GGX and GGX-F fail completely at both tiers due to imprecision in grasp generation, while centroid (90.0 %) and C-VS (100 %) are effective and do not leave the support polygon.

Screw picking (real). Reflecting a large sim-to-real gap in visual uncertainty, open-loop centroid fails completely in real trials despite CoM margins matching simulation (median 137.2 vs. 133.5 mm). Imperfect depth-segmentation and head camera calibration produce significant positional error (≥5cm) that control from single-shot detection is not robust to. Visual servoing recovers near-perfect performance (96.7% task success) while tethered, and 86.7 % while standing. Base motion sways significantly more in the real-world deployment (radial sway RMS 4.6mm) with a median peak base excursion of 18.0 mm per reach, compared to sway in simulation (0.10 mm), which effectively is static.

Despite this increased sway, in all C-VS trials the initial grasp closes on the screw; the five failures across real-world tether and stand removal are caused by off-axis extraction. Deviation from a vertical extraction due to non-linear OMPL motion plans at robot joint limits cause the screw to jam against the bore wall. Then, the gripper's grasp force is insufficient to overcome the resulting friction, leading to slip. Across all C-VS trials, the average wallclock time from image detection trigger to grasp completion is 75±16.1s. *Placement variance (Var. Stand).* Sampling five grasps with visual servoing at each of six stand poses drawn from the navigation placement distribution, three near poses achieve 11/15 successful grasps and three (one very near, two far) fail completely since the pose is out of reach. Stability at the near poses (median 137.2mm) decreases to 124.8 mm ($p = 0.022$) at the off-nominal poses. The GOLEM architecture enables characterization and pinpointing of failure directly to standing poses, rather than to grasp or removal imprecision.

Task (Env)	Cent.	Antip.	GGX	GGX-F	C-VS
Fridge (sim)					
Tether	**100** [89, 100]	63.3 [46, 78]	53.3 [36, 70]	96.7 [83, 99]	—
Stand	**80.0** [63, 90]	26.7 [14, 44]	53.3 [36, 70]	70.0 [52, 83]	—
Screw (sim)					
Tether	**100** [89, 100]	—	0 [0, 11]	0 [0, 11]	**100** [89, 100]
Stand	90.0 [74, 97]	—	0 [0, 11]	0 [0, 11]	**100** [89, 100]
Screw (real)					
Tether	0 [0, 11]	—	—	—	**96.7** [83, 99]
Stand	0 [0, 11]	—	—	—	**86.7** [70, 95]
Var. Stand	0 [0, 11]	—	—	—	**36.7** [22, 55]

(a) Grasp success (%, 95% confidence intervals below) across two tasks, three levels of autonomy, and five methods ($n = 30$ trials).



(b) center-of-mass (CoM) margin evaluation across simulated and real-world trials.

Fig. 5: **(a) Manipulation experiments on fridge handle opening and battery screw picking**, across autonomy levels of *Tether* (weight assisted by rope tether); *Stand* (standing with ALMI); and *Var. Stand* (stand pose perturbed from locomotion placement variance). Five grasp synthesis methods are evaluated on vision-processed segmented point clouds: centroid (Cent.), antipodal (Antip.), GraspGenX [37] (GGX), wrist-reachability re-ranked GraspGenX (GGX-F), and centroid visual servoing (C-VS). **(b) Stability sensitivity to grasp synthesis methods.** Across **(1)** fridge opening tasks in RoboCasa and **(2)** battery screw picking (sim and real), we measure episode-level average CoM margin: the signed distance from the floor-projected center-of-mass to the boundary of the robot support polygon (ground truth in simulation, IMU-state-estimated in reality). Negative values indicate a CoM exiting the support polygon, leading to a recovery step or failure (shaded band in red, boxes are median and IQR with 10–90 % whiskers).

D. Case Study: Language-Conditioned Spatial Memory

GOLEM provides a platform for fair comparison of humanoid modules. We demonstrate how heterogeneous memory backends can leverage GOLEM's common RGB-D, pose, and navigation interfaces for language-conditioned spatial and temporal retrieval benchmarking in Tab. I. The benchmark contains 20 controlled RoboCasa object-relocation episodes: five each for a mug, bowl, can, and bottled water. In each episode, the camera follows the same eight-view route before and after the target moves to a different counter, producing 320 RGB-D observations. The evaluator issues 153 static, current-location, historical-location, and absent-object queries.

Static Mean Reciprocal Rank (MRR) measures basic retrieval quality. Current-location Top-1 measures whether the highest-ranked result reflects the object's present location. Outdated-location Top-1 measures how often stale evidence is ranked first. History coverage@3 measures retrieval of the object's previously observed locations among the top three results. Median query time measures retrieval efficiency.

SigLIP+FAISS adapted from [40] provides the lowest retrieval latency but has limited temporal awareness. VLMaps provides map-based spatial aggregation at moderate latency but frequently prioritizes outdated observations after objects move. VLM re-ranking improves retrieval and temporal disambiguation, at the cost of increased query latency on the order of $10^3\times$. GOLEMexposes these alternatives through the same interface, allowing selection of a memory backend suited for a task's spatiotemporal, and runtime requirements.

V. CONCLUSION

We present GOLEM, an end-to-end open source system architecture for humanoid autonomy motivated by the industrial task of EV battery disassembly. By exposing simulated and physical resources through identical ROS 2/DDS interfaces and capabilities as reusable modules, the architecture supports development by seamless mixing of the digital twin with real hardware, culminating in a Unitree H1-2 removing screws from a Hyundai Ioniq 5 battery pack.

We have compared grasping and balancing methods for screw removal as part of a Hyundai Ioniq 5 battery pack disassembly process and used GOLEM as a scaffold to perform fair comparisons between functionally equivalent modules. GOLEM enables the finding that screw removal errors reside primarily in locomotion error, motivating future work on precise relocation and repositioning. The digital twins provided by GOLEM reveal that simulation underpredicts perception error and base-motion coupling, so sim success rates on contact-rich, precise tasks are not predictive.

In future work, we will expand GOLEM by forceful manipulation routines, which will complement visual servoing

TABLE I: Comparison of memory modules through GOLEM's common interface. Average success over 20 RoboCasa episodes; bounds show 95% confidence intervals.

Method	Basic retrieval: Static MRR ↑	Temporal awareness: Current-location Top-1 ↑	Temporal awareness: Outdated-location Top-1 ↓	Temporal memory: History coverage@3 ↑	Efficiency: Median query time ↓
VLMaps	$77.5^{94.1}_{60.9}$ %	$20.0^{38.0}_{2.0}$ %	$53.5^{75.3}_{31.6}$ %	$45.0^{54.8}_{35.2}$ %	$20.4^{21.3}_{19.5}$ ms
SigLIP+FAISS	$67.5^{84.8}_{50.2}$ %	$35.7^{55.8}_{15.7}$ %	$41.4^{63.1}_{19.7}$ %	$42.5^{58.8}_{26.2}$ %	$3.6^{3.6}_{3.5}$ ms
SigLIP+FAISS+rerank	100.0%	$95.4^{98.6}_{92.2}$ %	0.0%	$97.5^{100}_{92.6}$ %	$\approx 19.7^{21.0}_{18.4}$ s

to address peg extraction and peg-in-hole assembly that occur during EV-LIB disassembly and reassembly.

We also note that as humanoid systems move toward deployment alongside people, the ability to attribute failures to specific components becomes a safety concern rather than only a scientific one; we intend GOLEM's modular evaluation to support this accountability, while noting that benchmark performance in simulation or controlled settings does not establish readiness for unsupervised operation around humans.

REFERENCES

[1] G. Harper, R. Sommerville, E. Kendrick, L. Driscoll, P. Slater, R. Stolkin, A. Walton, P. Christensen, O. Heidrich, S. Lambert, *et al.*, "Recycling lithium-ion batteries from electric vehicles," *Nature*, vol. 575, no. 7781, pp. 75–86, 2019.

[2] Z. Allen, M. Conway, L. Antieau, A. Devaraj, and N. Correll, "RAPID: Robotic agentic platform for intelligent disassembly — platform design and unscrewing evaluation," in *Proceedings of the 2026 IEEE International Conference on Automation Science and Engineering (CASE)*, 2026.

[3] K. Wegener, W. H. Chen, F. Dietrich, K. Dröder, and S. Kara, "Robot assisted disassembly for the recycling of electric vehicle batteries," *Procedia CIRP*, vol. 29, pp. 716–721, 2015.

[4] M. Choux, E. Marti Bigorra, and I. Tyapin, "Task planner for robotic disassembly of electric vehicle battery pack," *Metals*, vol. 11, no. 3, p. 387, 2021.

[5] Z. Luo, Y. Yuan, T. Wang, C. Li, F. Castañeda, S. Chen, Z.-A. Cao, J. Li, *et al.*, "Sonic: Supersizing motion tracking for natural humanoid whole-body control," *arXiv preprint arXiv:2511.07820*, 2025.

[6] H. Kim, T. Chen, J. Sun, L. W. Osterberg, Q. Chen, K. Wang, and M. Schwager, "Legs: Fine-tuning teleop-free vlas for humanoid loco-manipulation in an embodied gaussian splatting world," *arXiv preprint arXiv:2606.01458*, 2026.

[7] J. Shi, X. Liu, D. Wang, O. Lu, S. Schwertfeger, F. Sun, C. Bai, and X. Li, "Adversarial locomotion and motion imitation for humanoid policy learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[8] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu, "Robocasa: Large-scale simulation of everyday tasks for generalist robots," in *Robotics: Science and Systems (RSS)*, 2024.

[9] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

[10] M. Mittal, P. Roth, *et al.*, "Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning," *arXiv preprint arXiv:2511.04831*, 2025.

[11] M. Qu, D. T. Pham, F. Altumi, A. Gbadebo, N. Hartono, K. Jiang, M. Kerin, F. Lan, M. Micheli, S. Xu, and Y. Wang, "Robotic disassembly platform for disassembly of a plug-in hybrid electric vehicle battery: A case study," *Automation*, vol. 5, no. 2, pp. 50–67, 2024.

[12] S. Su, D. T. Pham, C. Ji, Y. Wang, J. Huang, W. Zhou, and H. Wang, "Design of a compliant device for peg-hole separation in robotic disassembly," *The International Journal of Advanced Manufacturing Technology*, vol. 124, no. 9, pp. 3011–3019, 2023.

[13] J. Liu, X. Zhang, X. Tu, W. Xu, and Z. Zhou, "Integration of compliant disassembly strategy and state recognition for robotic peg-hole disassembly," *Computers & Industrial Engineering*, p. 111772, 2025.

[14] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi, "Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning," in *Conference on Robot Learning (CoRL)*, 2024.

[15] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn, "Humanplus: Humanoid shadowing and imitation from humans," in *Conference on Robot Learning (CoRL)*, 2024.

[16] C. Wen, G. C. R. Bethala, Y. Hao, N. Pudasaini, H. Huang, S. Yuan, B. Huang, A. Nguyen, M. Wang, A. Tzes, and Y. Fang, "Humanoid agent via embodied chain-of-action reasoning with multimodal foundation models for zero-shot loco-manipulation," 2025.

[17] D. W. Calvert, "A system for fast, resilient, and adaptable loco-manipulation behaviors on humanoid robots," 2026.

[18] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive sampling: Real-time behaviour synthesis with MuJoCo," *arXiv preprint arXiv:2212.00541*, 2022.

[19] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022.

[20] W. Guo, T. Tyrovouzis, E. Flores, C. W. Ramsey, Z. K. Kingston, I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library 2.0," 2026.

[21] S. Macenski, F. Martín, R. White, and J. G. Clavero, "The marathon 2: A navigation system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2718–2725, 2020.

[22] J. Carpentier, G. Saurel, *et al.*, "The pinocchio c++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE/SICE International Symposium on System Integration*, pp. 614–619, 2019.

[23] N. Correll, D. Kriegman, S. Otto, and J. Watson, "A versatile robotic hand with 3d perception, force sensing for autonomous manipulation," *arXiv preprint arXiv:2402.06018*, 2024.

[24] W. Xie, M. Valentini, J. Lavering, and N. Correll, "Deligrasp: Inferring object properties with llms for adaptive grasp policies," 2024.

[25] X. Tan, W. Xie, and N. Correll, "Characterization, analytical planning, and hybrid force control for the Inspire RH56DFX hand," *arXiv preprint arXiv:2603.08988*, 2026.

[26] support.unitree.com, "About unitree h1-2."

[27] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart, "State estimation for legged robots—consistent fusion of leg kinematics and IMU," in *Proceedings of Robotics: Science and Systems (RSS)*, (Sydney, Australia), 2012.

[28] M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, and C. Semini, "Probabilistic contact estimation and impact detection for state estimation of quadruped robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1023–1030, 2017.

[29] N. Rotella, M. Bloesch, L. Righetti, and S. Schaal, "State estimation for a humanoid robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Chicago, IL, USA), pp. 952–958, 2014.

[30] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, pp. 19–67, 2005.

[31] N. Nechyporenko, Y. Zhang, S. Campbell, and A. Roncone, "Morphit: Flexible spherical approximation of robot morphology for representation-driven adaptation," 2026.

[32] S. Caron, Y. De Mont-Marin, R. Budhiraja, S. H. Bang, *et al.*, "Pink: Python inverse kinematics based on Pinocchio." https://github.com/stephane-caron/pink, 2025.

[33] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, pp. 72–82, December 2012. `https://ompl.kavrakilab.org`.

[34] G. R. Team *et al.*, "Gemini robotics: Bringing ai into the physical world," 2025.

[35] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "YOLO-World: Real-time open-vocabulary object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16901–16911, 2024.

[36] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[37] B. Han, Y.-W. Chao, E. Coumans, C. Eppner, B. Sundaralingam, J. Deng, S. Birchfield, and A. Murali, "Graspgen-x: Cross-embodiment 6-dof diffusion-based grasping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[38] P. Li, W. Guo, H. Zhang, T. Cai, X. He, Y. Guo, and H. Xiong, "Spatial memory for out-of-vision manipulation in vision-language-action," *arXiv preprint arXiv:2605.22283*, 2026. Accepted at ICML 2026.

[39] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10608–10615, 2023.

[40] A. Anwar, J. Welsh, J. Biswas, S. Pouya, and Y. Chang, "ReMEmbR: Building and reasoning over long-horizon spatio-temporal memory for robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2838–2845, 2025.