GROUNDED WORLD MODEL: LATENT PLANNING WITH LANGUAGE GOALS

Anonymous authors
Paper under double-blind review

ABSTRACT

World models such as DINO-WM and LeWM specify the goal with an image, which is difficult to obtain in advance for novel tasks. We present the Grounded World Model (GWM), a latent world model that enables zero-shot planning in the real world from language goals alone. Given a candidate action sequence and the current observation, GWM predicts the future in the visual space of a pretrained video-language embedding model. The frozen readout of this embedding model maps this imagined future and the task description into the same embedding space, where their negative cosine similarity serves as the planning cost. Training GWM requires only offline and task-agnostic video-action pairs and no language labels. In simulated experiments on WISER, planning with GWM, which executes the candidate action of lowest cost, solves 87% of 288 tasks with unseen instructions and visual signals, while ten fine-tuned VLAs average 22%. We then scale GWM up with real robot data, and use it for zero-shot planning in realistic simulation and real scenes. In the IsaacSim evaluation, planning with GWM completes all 70 trials across 14 tasks that require reasoning over referring expressions, matching a modular planner grounded by a frontier VLM, while $\pi_{0.5}$ reaches 37/70. Deployed on a real Franka, the same stack completes 55/60 separately evaluated pick-and-place sub-tasks, comparable to the modular planner's 52/60, with the full system running locally on a single consumer GPU.

1 INTRODUCTION



Figure 1: World Model planning with image goals and with language goals. *Left*: the image interface, where candidate futures are imagined and compared to the goal in a pure visual latent space. *Right*: planning with GWM, where the same imagination step runs inside the visual latent space of a video-language embedding model and a fixed readout compares the imagined future to the task description.

A world model predicts how a scene evolves under an agent's actions, enabling it to optimize actions toward a goal by minimizing the cost function (Ha & Schmidhuber, 2018). A recent line of work has made this approach practical for visuomotor control by predicting in the representation space of a pretrained vision encoder. DINO-WM plans over frozen DINOv2 patch features (Zhou et al., 2025a; Oquab et al., 2024), V-JEPA 2-AC post-trains an action-conditioned predictor on a self-supervised video encoder (Assran et al., 2025), and subsequent methods refine the encoder, predictor, and training recipe (Terver et al., 2026; Destrade et al., 2025; Zhang et al., 2026b; Maes et al., 2026; Sobal et al., 2025; Zhang et al., 2026a). These systems share a common planning interface, in which candidate action sequences are imagined forward in latent space and the candidate whose predicted future is closest to the representation of a *goal image* is executed (Fig. 1, left). However, specifying the goal

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.

with images is restrictive. A photograph of the desired end state in the target scene must be obtained before the robot can act, and a multi-stage task requires a separate photograph for each sub-goal. For example, to perform a single pick-and-place task, V-JEPA 2-AC requires two sub-goal images in addition to the final goal image and switches among them on a fixed schedule (Assran et al., 2025). Language is an easier goal interface to specify. It is available before the robot acts, and a long-horizon task can be decomposed to sub-instructions. But existing world models predict in a purely visual latent space, which provides no direct comparison between an imagined future and a language goal.

A straightforward fix is to align such a visual latent space with language through additional multimodal training (Liu et al., 2024). We invert this construction. Rather than aligning a world model's latent space with language, we build the world model *inside a space that is already aligned with language*. This design enables predicting task-agnostic visual future, while retaining the ability to score candidate actions against open-vocabulary instructions. Specifically, we start from a frozen video-language embedding model trained to retrieve videos from text. The model comprises a vision encoder that maps a video clip to a token-level latent and a multimodal readout that projects visual latents and text into a shared unit-norm embedding space, where cosine similarity measures semantic agreement. With this base model, we build **Grounded World Model (GWM)** to predict, from the current frame and a candidate action sequence, the token-level latent of the corresponding future video clip. At test time, the frozen readout maps this prediction and the instruction into the shared embedding space. During planning, the predicted future of each candidate action sequence is compared with the task instruction in this space, and the planner selects the candidate action sequence that minimizes their negative cosine similarity (Fig. 1, right). We call the world model *grounded* because its predictions can be grounded to language by this readout, although no language label is ever used to train it.

This construction has four benefits. First, the goal is stated in language, yet training needs only offline and task-agnostic video-action pairs, without even success filtering. Second, the dynamics stay task-agnostic and the cost function is built in zero-shot given the instruction. One model serves all tasks. Third, actions enter as rendered pixels, so no action tokenizer is trained and any robot with a kinematic model and camera parameters can be scored. Fourth, the full stack runs on one local consumer GPU. Our experiments support this design through four key findings:

- **Language grounds intent in imagined futures.** On WISER, a controlled testbed that separates grounding from motor-skill generalization, planning over GWM's predicted futures solves 87% of 288 held-out tasks. In contrast, ten VLAs fine-tuned from VLMs on the same demonstrations average only 22%. Ablation on WISER yields a robust recipe to support real-world data training.
- **The cost supports semantic grounding and contact-accurate action selection.** When trained on real robot data, GWM's cost function successfully guides a cross-entropy method (CEM) (Rubinstein & Kroese, 2004) search to localize four named photos on a tabletop. Furthermore, the same planner generates precise actions to push target cubes, achieving a 51% success rate in moving the target at least 3 cm without disturbing unrelated objects.
- **Zero-shot planning generalizes to open scenes.** In IsaacSim evaluation with realistic rendering, the GWM cost ranks language-blind trajectory proposals and the planner executes the one that matches the instruction. Planning with GWM completes 70/70 trials across 14 tasks requiring referring-expression reasoning, matching a modular planner (94%) grounded by Gemini-Robotics-ER (Gemini Robotics Team et al., 2025), while $\pi_{0.5}$ reaches 53% under a relaxed pick criterion.
- **The system is capable of zero-shot real-robot deployment.** We ask the same GWM planner to operate a Franka arm in an out-of-the-box evaluation with out-of-distribution viewpoint, novel objects, unseen instructions. The system completes 55/60 separately evaluated pick-and-place tasks, comparable to the modular planner's 52/60, with the entire stack running on a single NVIDIA RTX 5090 within 24 GB of GPU memory instead of a remotely hosted frontier VLM.

2 Related Work

World models and goal specification. DINO-WM (Zhou et al., 2025a), V-JEPA 2-AC (Assran et al., 2025), and a rapidly growing family of successors (Terver et al., 2026; Destrade et al., 2025; Wang et al., 2026b; Zhang et al., 2026b; Goswami et al., 2025; Maes et al., 2026; Sobal et al., 2025; Zhang et al., 2026a; Bar et al., 2025) predict future latents of a frozen or jointly learned encoder and plan by minimizing a latent distance to an encoded goal image. These methods are demonstrated in simulation and, for V-JEPA 2-AC and HWM, zero-shot on real Franka arms. The world model

remains task-agnostic because the task enters only through the cost (LeCun et al., 2022; Sobal et al., 2025). Our method keeps this recipe, including a frozen encoder, reward-free offline latent prediction, and test-time optimization, while using language (Lin et al., 2024; Du et al., 2023) to define the cost.

Language as a planning cost. GenRL (Mazzaglia et al., 2024) connects a video-language embedding to a reconstruction-based world model through a learned adapter and trains policies in imagination. FOREWARN (Wu et al., 2025) rolls out a world model and has a fine-tuned VLM verbalize and rank the rollouts. VLMPC (Zhao et al., 2024) and semantic world models (Berg et al., 2025) query a VLM on generated video or VQA-style predictions. LAGO (Barbeau et al., 2026) learns language-to-latent-subgoal mappings from thousands of language-annotated demonstrations per environment. These approaches learn an additional interface to the world model, use a generative VLM to evaluate candidate futures, or both. The GWM cost compares embeddings with a single cosine after the frozen readout, with no language labels for world-model training, no generative VLM queries, and no per-environment adaptation. Related reward models score robot videos against language for policy learning or verification, rather than for world-model planning (Lee et al., 2026; Liang et al., 2026; Kwok et al., 2026; Chen et al., 2026; Kang et al., 2025).

Modular VLM planners and VLAs. TiPToP (Shen et al., 2026) grounds instructions with the proprietary VLM, Gemini-Robotics-ER (Gemini Robotics Team et al., 2025) and open-vocabulary segmentation, then plans with task-and-motion optimization. It shares the geometric processing, grasp sampler, and motion planner with our real-world system and is our main comparison. VLAs fine-tune a pretrained VLM into an end-to-end policy (Brohan et al., 2023; Black et al., 2026; Physical Intelligence et al., 2025; Yang et al., 2025). Growing evidence indicates fine-tuning erodes inherited vision-language alignment, leading to poor generalization (Zhai et al., 2025; Song et al., 2025; Zhang et al., 2025; Li, 2025; Zhou et al., 2025b; Xu et al., 2025; Wang et al., 2026a). The VLA train-test gap on WISER is consistent with this pattern. GWM training leaves the pretrained embedding model's weights unchanged. On WISER, planning with GWM retains most of its training success on the held-out split (92% to 87%) while the VLAs fall from 90% to 22%.

3 Grounded World Model

A latent world model for visuomotor control consists of an observation model that encodes observation o_t into a latent state and a transition model that predicts the latent consequence of a candidate action sequence $a_{t:t+c}$ over horizon c (Zhou et al., 2025a). A planner specifies a goal and a cost that scores predicted futures, then selects the minimum-cost candidate action sequence. In prior works, the encoder is a self-supervised vision model like DINO and the goal is an image o_g, so the cost is a latent distance $\|\hat{z}_{t+c} - \mathrm{enc}(o_g)\|$ between predicted future and encoded goal.

3.1 From a Video-Language Embedding Model to a World Model

To make world models more broadly useful, we specify the goal with language. One could align that purely visual space with language through further multimodal training. We invert the construction and build the world model inside a space already aligned with language, that of a video-language embedding model. Trained for retrieval, it consists of a vision encoder E that maps an image or video clip to a token-level latent, and a multimodal readout B that pools such a latent, optionally with text, into a unit-norm embedding. As a result, cosine similarity between a clip's embedding and a text's embedding measures how well the text describes the clip.

Definition. A *Grounded World Model* comprises the frozen vision encoder E as its observation model and a learned predictor P_θ as its transition model (Fig. 2). It maps the current frame and a candidate action sequence to the token-level latent of the corresponding future video clip. GWM is not language-conditioned. It is language-grounded because it predicts in a visual latent state space from which a fixed pretrained readout B maps imagined futures and language goals into the same embedding space. The readout B is not part of GWM and enters only through the planning cost. Only P_θ is learned, while E and B are never updated, preserving the pretrained video-language alignment and world knowledge for grounding. In practice, we instantiate E and B with Qwen3-VL-Embedding-8B (Li et al., 2026) and WeMM-Embedding (Zhou et al., 2026).

Observations and actions as video. A candidate trajectory $a_{t:t+c}$ is tokenized by a renderer R into $K-1$ robot-only frames, a scheme we call Rendering-based Action Tokenization (RAT). Each

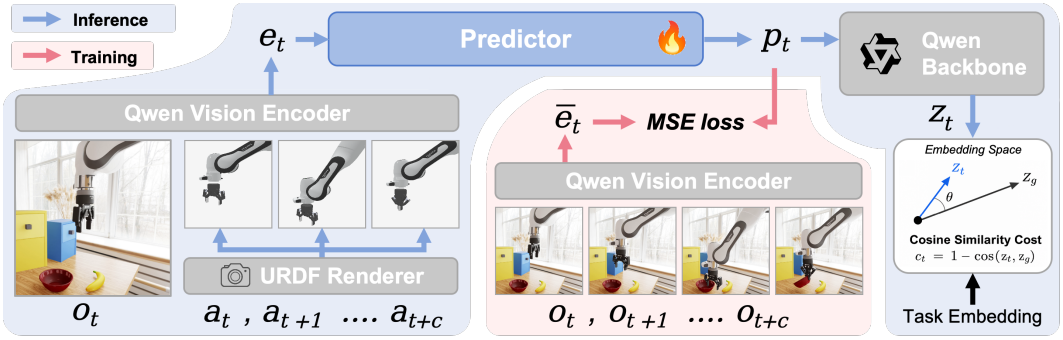


Figure 2: Training GWM and reading out its predictions for planning, shown for one candidate of a bowl pick-up task. *Blue flow*: the candidate trajectory $a_{t:t+c}$ is rendered by R into robot-only frames and encoded together with the current observation into e_t by the frozen vision encoder E; the learned predictor P_θ outputs the imagined future latent p_t, which the frozen readout B projects to z_t for comparison with the instruction embedding z_g, where $1 - \cos$ ranks candidates as Eq. 2 does. *Pink flow*: during training the ground-truth future clip is encoded by the same E into the target $\bar{e}_t$ of Eq. 1.

frame renders the robot mesh at one future joint state from the robot's URDF, using the scoring camera's parameters, namely its intrinsics and extrinsics, so the rendered arm appears where the real arm would appear in o_t. Everything except the robot is left black, since how the scene responds is what P_θ has to predict. Prepending the current frame gives a K-frame condition clip, encoded as $e_t = E(o_t, R(a_{t:t+c}))$. We use K=6. Actions enter as pixels, requiring no new tokenizer parameters, and any robot with a kinematic model and camera parameters can get its action encoded.

Predictor. P_θ is trained so that its output for the condition clip matches the latent that E assigns to the real K frames' future clip, $p_t = P_\theta(e_t) \approx \bar{e}_t = E(o_{t:t+c})$. Training requires only a dataset $\mathcal{D}$ of offline video-action trajectories. Training minimizes the latent-space error of this prediction,

$$\theta^* = \arg\min_\theta \mathbb{E}_{(o_{t:t+c},\, a_{t:t+c})\sim\mathcal{D}} \left\| P_\theta\big(E(o_t, R(a_{t:t+c}))\big) - E(o_{t:t+c}) \right\|_2^2. \tag{1}$$

The loss asks P_θ only to learn the transitions observed in the offline data. Semantic comparison with the goal is left to the fixed readout B. We predict the token-level latent rather than the pooled embedding $B(\bar{e}_t)$ because the token-level target supervises every token and keeps the spatial and temporal structure of the clip. Regressing the pooled embedding directly with the same predictor fits the training tasks but falls to 61% on the held-out split (§4).

3.2 Grounding Predicted Futures for Planning

Grounding readout. Because p_t is predicted in E's latent space, the fixed readout B treats it as an encoded video and maps the imagined future of the candidate action sequence to $z_t = B(p_t)$. The instruction ℓ defines the goal and is embedded by the same fixed readout as $z_g = B(\ell, E(o_t))$, with a fixed system prompt prepended to ℓ and the current frame o_t providing visual context. The embedding model was pretrained to align videos with their textual descriptions, while Equation 1 trains the predictor to approximate the latent encoding of the real future video. Action scoring reuses the model's pretrained text-to-video retrieval capability, so the planner searches for the action whose imagined future best matches the instruction. We call the composition of GWM, the fixed readout, and negative cosine similarity the **GWM cost**. Once the embeddings are computed, comparison requires one inner product per candidate.

Planning with GWM. A basic planner obtains N candidates $a^1, \ldots, a^N$ from a trajectory proposal mechanism $q(a \mid o_t)$ and executes the one of minimum GWM cost,

$$\begin{aligned} \mathcal{C}(a^n \mid o_t, \ell) &= -\big\langle B\big(P_\theta(E(o_t, R(a^n_{t:t+c})))\big),\ B(\ell, E(o_t))\big\rangle, \\ n^* &= \arg\min_{n\in\{1,\ldots,N\}} \mathcal{C}(a^n \mid o_t, \ell), \qquad a^* = a^{n^*}, \end{aligned} \tag{2}$$

with unit-normalized embeddings, so the inner product is cosine similarity and the cost is its negative. GWM defines a task-agnostic space of imagined futures, while each language goal induces a different

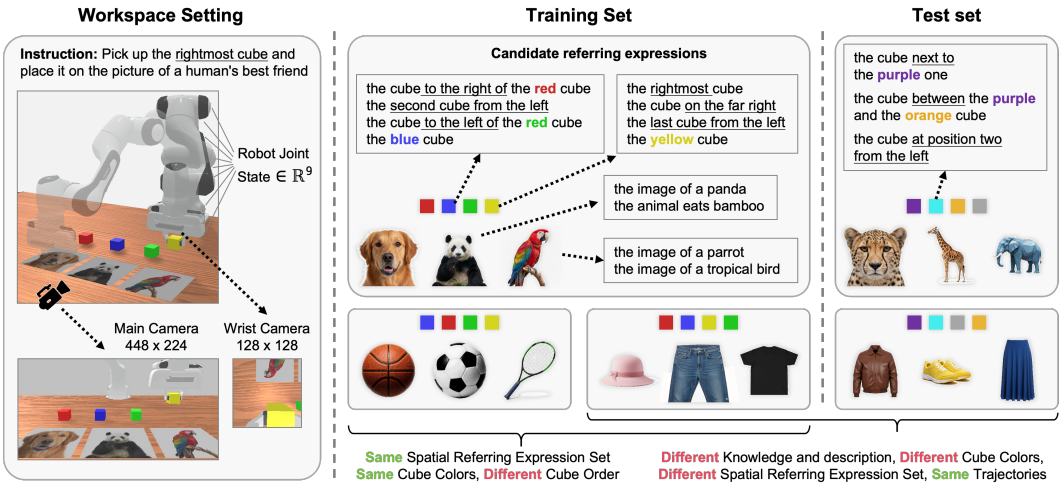


Figure 3: The WISER testbed. Each observation combines the instruction ℓ, joint positions, gripper state, and camera input o_t. Each of the 24 world-knowledge categories has a training scene and a paired test scene. The two splits share the same 12 motions but have disjoint images, referring expressions, and cube colors. Cube order is randomized across categories, so even a fixed position, such as the second cube from the left, carries different visual and semantic cues across the splits.

cost landscape over that space. Because GWM is agnostic to q, the proposals can be full trajectories or target end-effector poses T^*, which an interpolator $\mathcal{I}$ expands into a joint-space trajectory $a_{t:t+c} = \mathcal{I}(T_t, T^*)$ from the current pose T_t. In the experiments, we use three instances of q. Joint-state retrieval of demonstrated trajectories serves §4. A Gaussian over T^*, refined by CEM (Rubinstein & Kroese, 2004) and expanded by $\mathcal{I}$, serves §5. Structured proposals, where grasp sampling and motion planning give each object candidate trajectories, serve §6 and §7.

4 Preliminary Experiments

Planning with GWM relies on two measurable properties. (P1) The fixed readout B must map clips and instructions so their cosine ranks futures correctly in robot scenarios. (P2) GWM must predict the future latent from the current frame and candidate actions. We isolate and measure both properties on a simulated testbed called WISER before moving to open scenes.

WISER. Open-scene success entangles grounding with perception and action quality. To expose the role of grounding, we build **W**orld-knowledge **I**ntegrated **S**emantic **E**mbodied **R**easoning (WISER), which pairs 24 categories of world knowledge, including food, animals, landmarks, and numbers, with tabletop pick-and-place in ManiSkill (Tao et al., 2025). Each category has training and test splits (Fig. 3). A scene contains four cubes and three printed images, and each instruction names a cube and destination, giving 12 tasks per scene and 288 in each split. The test visuals and instructions are disjoint from training, but the 12 motions are shared, so every test task can be solved by recalling a demonstrated motion. This separates semantic generalization from the need to learn new motor skills.

Setup. We collect data on the training tasks with a scripted planner, producing 1,728 demonstrations, six per task. Appendix B gives the collection procedure, metrics, full protocol and ablation details. We compare planning with GWM to VLAs fine-tuned on the same demonstrations. Planning uses closed-loop receding-horizon control. Every 20 steps, the N=12 candidates are retrieved by joint-state matching. These trajectories are scored with Eq. 2, and the goal query used on this testbed is given in Appendix B.3. Retrieval supplies executable candidates, so planning tests whether the predicted futures identify the correct motion. VLAs must map the same new scenes and instructions to demonstrated actions. The planning hyperparameters for GWM are selected with the Oracle on the training split. The VLAs and planning with GWM are evaluated zero-shot on the test tasks. We report the fraction of completed tasks in Table 1 and Fig. 4. Detailed metrics such as target object grasp and destination reach success are reported in Appendix B.

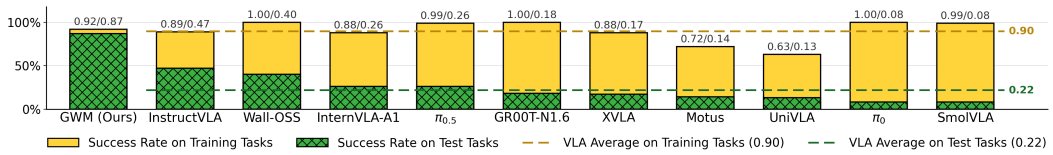


Figure 4: WISER success of planning with GWM and of ten fine-tuned VLAs on training tasks (yellow) and semantically disjoint test tasks (green). Dashed lines show the VLA averages.

The space reads intent from the futures. The Oracle diagnostic replaces each prediction p_t^n with the ground-truth future latent $\bar{e}_t^n$, revealing what the embedding space could support under perfect prediction. It completes 90% of the training tasks and 93% of the 288 test tasks. This supports P1, showing that cosine to the instruction can rank the correct future among twelve candidates across different colors, spatial relations, and world knowledge. The same Oracle falls to 33% in the 2B sibling and to 0% in a CLIP-style video encoder (Bolya et al., 2025).

Table 1: WISER success of GWM and its variants.

Method	Train Success	Test Success
GWM (Ours)	0.92	0.87
GWM w/ action encoder	0.74	0.24
GWM w/ pooled target	0.92	0.61
GWM w/ WEMM	0.92	0.76
GWM w/ xArm6	0.87	0.83
Oracle	0.90	0.93

The future is predictable in this space. Planning with futures predicted by GWM completes 87% of the test tasks, narrowing the Oracle gap to 6 percentage points and supporting P2. Removing P_θ and scoring its input latent e_t directly gives 9%. Predicting pixels with DreamDojo (Gao et al., 2026), re-encoding them with E, and applying the same readout gives 17%. These comparisons support learning a forward predictor in latent space, which avoids synthesizing and re-encoding a future video for each candidate. A third variant keeps the same predictor but regresses the pooled embedding $B(\bar{e}_t)$ directly and scores it without the readout. It matches GWM on the training tasks at 92% but falls to 61% on the held-out split, favoring token-level latent prediction. Furthermore, how actions enter the predictor is equally critical. Replacing RAT with a learned encoder over raw numeric actions yields only 24% at test. In contrast, RAT enables the Panda-trained GWM to predict futures for visually distinct xArm6 trajectory proposals, achieving zero-shot success at 83%.

The recipe is not tied to one embedding model. Replacing Qwen3-VL-Embedding-8B with WEMM (Zhou et al., 2026), a 9B video-language embedding model, and training the same predictor with the same recipe gives 92% on the training tasks and 76% on the test tasks, while the Oracle in that space reaches 94% and 88% (Table 1, Appendix B.4). This result shows that the same recipe also works with another video-language embedding model. Qwen3-VL-Embedding-8B performs better on WISER, so we use it for the subsequent experiments.

Planning in the frozen space retains semantic generalization. Ten VLAs, including the world-action model Motus, are fine-tuned on the same data as GWM. Their average success falls from 90% on training tasks to 22% on test tasks, and the best reaches 47% (Fig. 4). The models still perform perfect pick-and-place motions in the test scenes, but choose the wrong cube or destination. The gap suggests a poor ability to bind instruction to object despite recalling in-distribution motions.

5 Scaling to Real-World Data

We scale the WISER-selected recipe to real-world-scale data. We train a new GWM on DROID (Khazatsky et al., 2024) and MolmoB0T simulation data (Deshpande et al., 2026). The resulting predictor checkpoint, paired with the frozen E and B, is evaluated zero-shot across all subsequent experiments. Here, we define *zero-shot* as deployment in an unseen environment featuring novel viewpoints, new objects, and instructions, without any target-task data, demonstrations, rewards, or fine-tuning. Before integrating GWM into a full manipulation system, we test whether its cost landscape remains semantically grounded and spatially precise despite the shift in training data and visual realism.

Setup. We build separate pointing and pushing scenes in IsaacSim. The pointing task evaluates target-level semantic grounding, while pushing tasks test whether the cost can distinguish trajectories

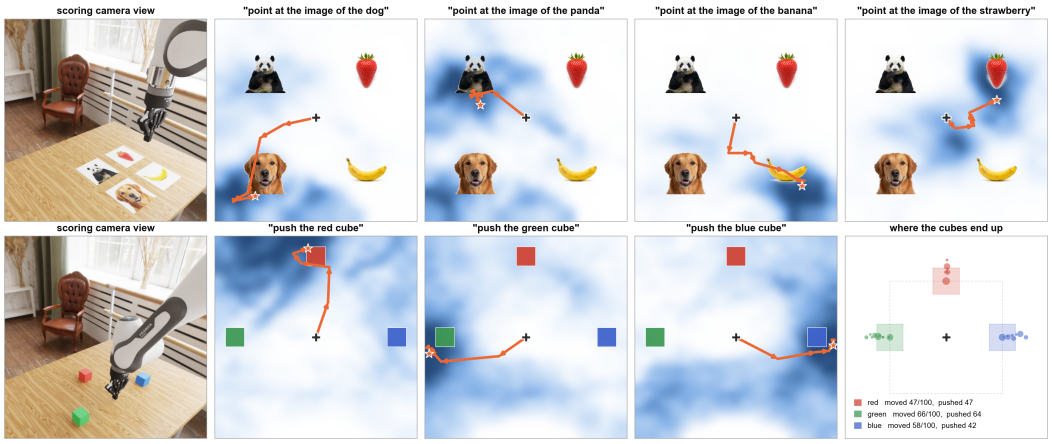


Figure 5: GWM-based scoring landscapes after scaling GWM to real-world data. Each row begins with an external scoring view. *Top*: scores over end-effector hover positions for four pointing instructions. The orange path follows the CEM mean, and the star marks the selected final gripper pose. Darker blue denotes a higher value of the probe's scoring objective. *Bottom*: scores over slide endpoints for three pushing instructions, followed by the named cube's final positions under each instruction. The outcome legend distinguishes "moved" (at least 1 cm) from "pushed" (at least 3 cm).

that make effective contact. We use CEM (Rubinstein & Kroese, 2004) to test whether this cost guides search from an uninformed Gaussian over candidate endpoints. In both, CEM searches over the tabletop endpoint (x, y) of a target pose T^*, which $\mathcal{I}$ expands into a trajectory scored with GWM. Figure 5 shows the scoring landscapes and search traces; Appendix C gives the probe protocols.

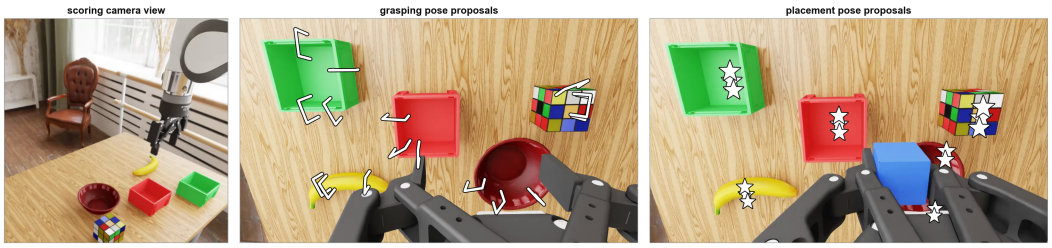
The cost provides robust semantic grounding. In the pointing task, four printed photos of a dog, a panda, a banana, and a strawberry lie on an empty table. The instruction is "point at the image of the {name}." The visualization shows that the lowest-cost region overlaps the named photo. CEM's best sample reaches the target in 4/4 searches. These searches show that the cost function provides a correct semantic signal to guide the search and optimization.

The cost exhibits fine-grained spatial precision. In the pushing task, three cubes surround a narrow closed fingertip blade. For the instruction "push the {color} cube," candidate trajectories slide the fingertip in a straight line to the endpoint (x, y), requiring contact precise enough to move the named cube by at least 3 cm. We run 100 CEM trials per instruction across different seeds and execute every selected trajectory. The cost map ranks the target cube first for all three instructions, and 153/300 selected trajectories move it by at least 3 cm. No unnamed cube moves in any of the 300 trials.

6 Zero-Shot Planning in Simulation

Setup. The preceding probes evaluate the cost landscape within a two-dimensional action space. We now integrate the same GWM and cost formulation into a complete manipulation system, where perception, feasible trajectory generation, and physical execution all play critical roles. Figure 6 illustrates the simulated scene and lists the 14 instructions covering 10 picking and 4 placement tasks. Each task comprises five trials per method, judged by one scripted judge shared by all methods.

Methods. We compare four methods: GWM (ours), TiPToP, $\pi_{0.5}$, and V-JEPA 2-AC. The GWM and TiPToP systems share wrist-depth geometry, RANSAC table fitting, M2T2 grasp sampling, and cuRobo motion planning. *GWM (Ours)*: Our proposal mechanism generates candidate actions for every object without reading the instruction, so the GWM cost supplies the sole semantic selection signal. For picking, the system clusters the wrist depth image into unlabeled objects and samples grasps over the whole scene with M2T2 (Yuan et al., 2023), keeping 16 target poses T^* spread across the objects. cuRobo (Sundaralingam et al., 2023) expands each pose into a collision-free joint-space trajectory $a_{t:t+c} = \mathcal{I}(T_t, T^*)$, which the GWM cost scores from an external view that matches DROID's shoulder-camera perspective. Placement poses follow the same geometry-based principle. The system selects an object by its candidates' aggregate GWM score, then chooses a feasible



target	instruction
banana	"pick up the fruit"
banana	"pick up the yellow object"
banana	"pick up the thing you could eat"
banana	"pick up the object that is neither a toy nor a container"
cube	"pick up the puzzle toy"

target	instruction
cube	"pick up the most colorful object"
cube	"pick up the object closest to the bowl that is not red"
bowl	"pick up the object you would eat a meal from"
bowl	"pick up the object between the cube and the banana"
bowl	"pick up the round container"

target	instruction
red bin	"put the block into the red box"
green bin	"put the block into the green box"
red bin	"put the block into the box that has the color of a tomato"
green bin	"put the block into the box that has the color of grass"

Figure 6: Zero-shot planning in DROID-sim. *Left*: The scoring view, containing a banana, a cube, a bowl, and two colored bins. *Middle*: 16 language-blind grasp candidates, visualized as gripper outlines from the wrist camera at the home pose. *Right*: 16 placement candidates, shown as stars after the block has been grasped. *Bottom*: The 14 instructions and the target.

grasp by M2T2 confidence (Appendix D). Scoring 16 structured candidates per decision keeps open-scene execution within our inference budget. *TiPToP* (Shen et al., 2026): Uses the proprietary frontier VLM Gemini-Robotics-ER (Gemini Robotics Team et al., 2025) and open-vocabulary SegmentAnything (Kirillov et al., 2023) segmentation to produce symbolic goals. $\pi_{0.5}$ (Physical Intelligence et al., 2025): The official DROID joint-position checkpoint, evaluated zero-shot. *V-JEPA 2-AC* (Assran et al., 2025): A DROID-trained world model from the image-goal lineage. It receives a pre-captured goal image and scores all 16 trajectories in the same candidate pools.

Results. Table 2 separates *grounding failures*, where a method selects the wrong object or destination or fails to specify a goal, from *planning failures*, where the grasp slips or no valid plan exists. Planning with GWM completes 70/70 from the scoring view and 65/70 from the opposite view, where all five failures fall on one instruction whose target is small and shadowed from that viewpoint. TiPToP reaches 66/70 (94%). Its failures comprise three grasp slips and one grounding failure, where the VLM failed to produce the structured goal required by cuTAMP (Shen et al., 2024), so planning never begins. Despite their shared components, the two stacks differ in grounding, grasp selection, and placement construction, so this is a system-level comparison. An overview of the two stacks is given in Appendix D.1. $\pi_{0.5}$ reaches 37/70 when picks count if the target ever rises above 0.01 m during the episode, accommodating lifts that are subsequently dropped; Appendix D gives the strict score. V-JEPA 2-AC completes 25/70.

Table 2: Success over 14 tasks and five trials per task, with failures attributed where the method exposes the distinction. $\pi_{0.5}$ is scored under a relaxed pick criterion (Appendix D).

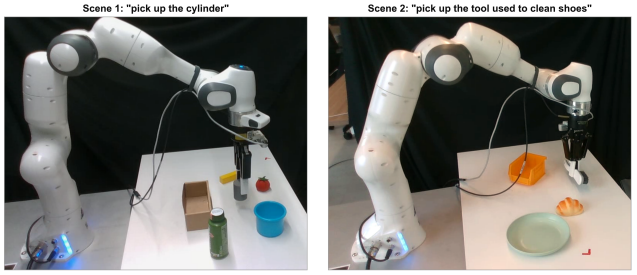
Method	Success	Grounding failures	Planning failures
GWM (Ours)	**70/70**	0	0
Opposite View	65/70	5	0
TiPToP	66/70	1	3
$\pi_{0.5}$	37/70	-	
V-JEPA 2-AC	25/70	45	0

Beyond Pick and Place. To test whether the semantic interface itself is tied to only those skills, we score drawer-pull trajectories in a three-cabinet scene. Planning with GWM selects and opens the instructed drawer in 5/6 tasks. Planning with GWM in new scenes requires only effective trajectory candidates, and no new grounding schema. TiPToP instead requires new symbolic components to be defined manually for a new skill grounding. The evaluation protocol is in Appendix D.2.



7 Real-Robot Experiments

Setup. We deploy the same GWM and planning interface on a physical Franka Panda with a Robotiq 2F-140 gripper, without task-specific adaptation. The hardware system retains the simulation stack's components and planning interface. Both systems use the same robot and wrist-mounted RGB-D camera, with scene geometry reconstructed by FoundationStereo (Wen et al., 2025). Our entire stack



target	instruction
cylinder	"pick up the cylinder"
box	"place it inside the box"
tomato	"pick up the object that can eat"
cup	"place it inside the cup"
yellow block	"pick up the yellow object"
cup	"place it in the container in the sky color"
drink	"pick up the drink"
box	"place it in the container made of paper"
brush	"pick up the tool used to clean shoes"
yellow box	"place it into the yellow box"
croissant	"pick up the croissant"
plate	"place it into the plate"

Figure 7: The two real-robot scenes from the external scoring camera, each captured while executing the pick instruction in its title. The figure also lists the twelve instructions and their object or container referents. Scene 1 was recorded under artificial light in the evening, and scene 2 under daylight.

runs locally on a single RTX 5090, with GPU memory use within 24 GB. TiPToP's grounding stack requires a remotely hosted VLM, whose call takes a median of 6.0 s per decision, while scoring all 16 candidates with GWM takes about 2.5 s. Appendix E gives the full hardware experiment details.

Tasks. We evaluate pick-and-place in two tabletop scenes, shown with their instructions in Fig. 7. The robot plans and executes the pick, re-observes, and then plans and executes the place. Following TiPToP, trajectories are executed open-loop. Pick and place sub-tasks are scored separately. Five trials per instruction give 60 trials per method, comprising 40 in scene 1 and 20 in scene 2. We exclude rounds where a sensor or driver fault prevents planning.

Table 3: Real-robot success over 60 separately scored pick-and-place sub-task trials per method.

Method	Scene 1	Scene 2	Total	Ground. fail.	Plan. fail.
GWM (Ours)	36/40	19/20	**55/60**	2	3
TiPToP	37/40	15/20	52/60	0	8

Results. Planning with GWM completes 55/60 sub-task trials, with 36/40 in scene 1 and 19/20 in scene 2, and TiPToP completes 52/60 (Table 3), so the two systems are comparable while only ours runs locally. TiPToP makes no grounding errors, but all eight failures arise in planning. Three are placement drops in scene 1, where it succeeds on 20/20 picks and 17/20 placements. In scene 2, "place into the yellow box" produces no plan in five trials, while every pick and plate placement succeeds. GWM's five failures comprise three planning and two grounding failures. In both grounding failures, it chooses a distractor close to the target. Successful instructions require function, material, and indirect color to be resolved against novel objects. Examples include "the object that can eat," "made of paper," and "the sky color." Planning with GWM resolves these references by comparing language with imagined futures through the GWM cost, without a semantic detector, open-vocabulary segmenter, or generative VLM query. On scene 1, the system completes 34/40 sub-task trials from an alternative shoulder-height scoring view, compared with 36/40 from the reported view (Appendix F). Hardware videos and code for deploying the system are available in the supplementary material.

8 Conclusion

We built GWM from the frozen vision encoder of a video-language embedding model and a learned predictor that approximates state transitions in its visual latent space. A fixed pretrained readout maps GWM's predicted futures and language goals into the same embedding space, where negative cosine similarity supplies the planning cost. This preserves the separation between task-agnostic dynamics and task-specific goals. The experiments trace this construction from controlled diagnosis to physical deployment. On WISER, planning with predicted futures generalizes to unseen instructions and visual conditions. When trained on real robot data, the same model supports zero-shot manipulation in simulation and on hardware, achieving performance comparable to a frontier-VLM-based planner.

Limitations. When integrating into a planner, the system depends on a module to propose executable trajectory candidates. Thus the quality of proposals matters. The grounding accuracy also depends on the video-language embedding model's ability to recognize robot behavior. Training a more powerful base model and building GWM in its latent space is a natural next step.

Ethics statement

Real-robot experiments were conducted by the authors in a controlled workspace with standard safety measures (workspace limits, human supervision, emergency stop). No human subjects, personal data, or crowd workers were involved. The WISER task images are AI-generated to avoid copyright issues. Simulation assets and datasets are used under their respective licenses. The method selects among externally generated candidate trajectories and does not by itself introduce risks beyond those of standard robot manipulation research.

Reproducibility statement

Appendices A-F give the verbatim system prompt, goal-query construction, architecture, corpora, training configuration, and complete experimental protocols and judges. Code, the WISER task suite, and evaluation harnesses will be released as anonymous supplementary material.

AI use statement

Large language models were used to assist with text editing, LaTeX drafting, and analysis scripting. They also assisted with methodological feedback, diagnostic protocol design, and interpretation of results. All experiments, results, and numbers were produced and verified by the authors. No reported experimental measurements or citations were generated by AI. The WISER task images are AI-generated. The authors take full responsibility for the content of this paper.

References

Hadi Alzayer, Wenlong Huang, Haonan Chen, Christopher Luey, Lvmin Zhang, Maneesh Agrawala, Gordon Wetzstein, Li Fei-Fei, Yilun Du, Jiajun Wu, and Jia-Bin Huang. Masked visual actions for unified world modeling, 2026. URL https://arxiv.org/abs/2607.19343.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15791–15801. IEEE, 2025.

Samuel Barbeau, Simon Roy, Giovanni Beltrame, Christian Desrosiers, and Nicolas Thome. Latent goal prediction from language for model-based planning. *arXiv preprint arXiv:2606.20627*, 2026.

Jacob Berg, Chuning Zhu, Yanda Bao, Ishan Durugkar, and Abhishek Gupta. Semantic world models. *arXiv preprint arXiv:2510.19818*, 2025.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model, 2025. URL https://arxiv.org/abs/2512.13030.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Daniel Li, Piotr Dollár, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network, 2025. URL https://arxiv.org/abs/2504.13181.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023. URL https://arxiv.org/abs/2307.15818.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions, 2025. URL https://arxiv.org/abs/2505.06111.

Remi Cadene, Simon Alibert, Francesco Capuano, Michel Aractingi, Adil Zouitine, Pepijn Kooijmans, Jade Choghari, Martino Russi, Caroline Pascal, Steven Palma, Mustafa Shukor, Jess Moss, Alexander Soare, Dana Aubakirova, Quentin Lhoest, Quentin Gallouédec, and Thomas Wolf. Lerobot: An open-source library for end-to-end robot learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=CiZMMAFQR3.

Junhao Cai, Zetao Cai, Jiafei Cao, Yilun Chen, Zeyu He, Lei Jiang, Hang Li, Hengjie Li, Yang Li, Yufei Liu, Yanan Lu, Qi Lv, Haoxiang Ma, Jiangmiao Pang, Yu Qiao, Zherui Qiu, Yanqing Shen, Xu Shi, Yang Tian, Bolun Wang, Hanqing Wang, Jiaheng Wang, Tai Wang, Xueyuan Wei, Chao Wu, Yiman Xie, Boyang Xing, Yuqiang Yang, Yuyin Yang, Qiaojun Yu, Feng Yuan, Jia Zeng, Jingjing Zhang, Shenghan Zhang, Shi Zhang, Zhuoma Zhaxi, Bowen Zhou, Yuanzhen Zhou, Yunsong Zhou, Hongrui Zhu, Yangkun Zhu, and Yuchen Zhu. Internvla-a1: Unifying understanding, generation and action for robotic manipulation, 2026. URL https://arxiv.org/abs/2601.02456.

Shirui Chen, Cole Harrison, Ying-Chun Lee, Angela Jin Yang, Zhongzheng Ren, Lillian J. Ratliff, Jiafei Duan, Dieter Fox, and Ranjay Krishna. Topreward: Token probabilities as hidden zero-shot rewards for robotics, 2026. URL https://arxiv.org/abs/2602.19313.

Abhay Deshpande, Maya Guru, Rose Hendrix, Snehal Jauhri, Ainaz Eftekhar, Rohun Tripathi, Max Argus, Jordi Salvador, Haoquan Fang, Matthew Wallingford, et al. Molmob0t: Large-scale simulation enables zero-shot manipulation. *arXiv preprint arXiv:2603.16861*, 2026.

Matthieu Destrade, Oumayma Bounou, Quentin Le Lidec, Jean Ponce, and Yann LeCun. Value-guided action planning with jepa world models, 2025. URL https://arxiv.org/abs/2601.00844.

Kamil Dreczkowski, Pietro Vitiello, Vitalis Vosylius, and Edward Johns. Learning a thousand tasks in a day. *Science Robotics*, 10(108), November 2025. ISSN 2470-9476. doi: 10.1126/scirobotics.adv7594. URL http://dx.doi.org/10.1126/scirobotics.adv7594.

Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation, 2023. URL https://arxiv.org/abs/2302.00111.

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, Qianli Ma, Seungjun Nah, Loic Magne, Jiannan Xiang, Yuqi Xie, Ruijie Zheng, Dantong Niu, You Liang Tan, K. R. Zentner, George Kurian, Suneel Indupuru, Pooya Jannaty, Jinwei Gu, Jun Zhang, Jitendra Malik, Pieter Abbeel, Ming-Yu Liu, Yuke Zhu, Joel Jang, and Linxi "Jim" Fan. Dreamdojo: A generalist robot world model from large-scale human videos, 2026. URL https://arxiv.org/abs/2602.06949.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world, 2025. URL https://arxiv.org/abs/2503.20020.

Raktim Gautam Goswami, Amir Bar, David Fan, Tsung-Yen Yang, Gaoyue Zhou, Prashanth Krishnamurthy, Michael Rabbat, Farshad Khorrami, and Yann LeCun. World models can leverage human videos for dexterous manipulation, 2025. URL https://arxiv.org/abs/2512.13644.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution, 2018. URL https://arxiv.org/abs/1809.01999.

Hao Su's Lab. MPlib: A lightweight motion planning library, 2024. URL https://github.com/haosulab/MPlib. GitHub repository.

Keller Jordan. Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/, 2024. Accessed: 2026-03-03.

Gi-Cheon Kang, Junghyun Kim, Kyuhwan Shim, Jun Ki Lee, and Byoung-Tak Zhang. Clip-rt: Learning language-conditioned robotic policies from natural language supervision, 2025. URL https://arxiv.org/abs/2411.00508.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. URL https://arxiv.org/abs/2502.19645.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023. URL https://arxiv.org/abs/2304.02643.

Jacky Kwok, Xilun Zhang, Mengdi Xu, Yuejiang Liu, Azalia Mirhoseini, Chelsea Finn, and Marco Pavone. Scaling verification can be more effective than scaling policy learning for vision-language-action alignment, 2026. URL https://arxiv.org/abs/2602.12281.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Tony Lee, Andrew Wagenmaker, Karl Pertsch, Percy Liang, Sergey Levine, and Chelsea Finn. Roboreward: General-purpose vision-language reward models for robotics, 2026. URL https://arxiv.org/abs/2601.00675.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking, 2026. URL https://arxiv.org/abs/2601.04720.

Quanyi Li. Task reconstruction and extrapolation for π_0 using text latent, 2025. URL https://arxiv.org/abs/2505.03500.

Zhiqi Li, Guo Chen, Shilong Liu, Shihao Wang, Vibashan VS, Yishen Ji, Shiyi Lan, Hao Zhang, Yilin Zhao, Subhashree Radhakrishnan, Nadine Chang, Karan Sapra, Amala Sanjay Deshmukh, Tuomas Rintamaki, Matthieu Le, Ilia Karmanov, Lukas Voegtle, Philipp Fischer, De-An Huang, Timo Roman, Tong Lu, Jose M. Alvarez, Bryan Catanzaro, Jan Kautz, Andrew Tao, Guilin Liu, and Zhiding Yu. Eagle 2: Building post-training data strategies from scratch for frontier vision-language models, 2025. URL https://arxiv.org/abs/2501.14818.

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S. Huang, Luke Zettlemoyer, Dieter Fox, Yu Xiang, Anqi Li, Andreea Bobu, Abhishek Gupta, Stephen Tu, Erdem Biyik, and Jesse Zhang. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons, 2026. URL https://arxiv.org/abs/2603.02115.

Jessy Lin, Yuqing Du, Olivia Watkins, Danijar Hafner, Pieter Abbeel, Dan Klein, and Anca Dragan. Learning to model the world with language, 2024. URL https://arxiv.org/abs/2308.01399.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection, 2024. URL https://arxiv.org/abs/2303.05499.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

Pietro Mazzaglia, Tim Verbelen, Bart Dhoedt, Aaron Courville, and Sai Rajeswar. Genrl: Multimodal-foundation world models for generalization in embodied agents. *Advances in neural information processing systems*, 37:27529–27555, 2024.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z.

Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Sabela Ramos, Sertan Girgin, Léonard Hussenot, Damien Vincent, Hanna Yakubovich, Daniel Toyama, Anita Gergely, Piotr Stanczyk, Raphael Marinier, Jeremiah Harmsen, Olivier Pietquin, and Nikola Momchev. Rlds: an ecosystem to generate, share and use datasets in reinforcement learning, 2021.

Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*, volume 133. Springer, 2004.

William Shen, Caelan Garrett, Nishanth Kumar, Ankit Goyal, Tucker Hermans, Leslie Pack Kaelbling, Tomás Lozano-Pérez, and Fabio Ramos. Differentiable gpu-parallelized task and motion planning. *arXiv preprint arXiv:2411.11833*, 2024.

William Shen, Nishanth Kumar, Sahit Chintalapudi, Jie Wang, Christopher Watson, Edward Hu, Jing Cao, Dinesh Jayaraman, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Tiptop: A modular open-vocabulary planning system for robotic manipulation. *arXiv preprint arXiv:2603.09971*, 2026.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics, 2025. URL https://arxiv.org/abs/2506.01844.

Uladzislau Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim GJ Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. *Advances in Neural Information Processing Systems*, 38:43905–43941, 2025.

Wenxuan Song, Ziyang Zhou, Han Zhao, Jiayi Chen, Pengxiang Ding, Haodong Yan, Yuxin Huang, Feilong Tang, Donglin Wang, and Haoang Li. Reconvla: Reconstructive vision-language-action model as effective robot perceiver. *arXiv preprint arXiv:2508.10333*, 2025.

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. Curobo: Parallelized collision-free robot motion generation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8112–8119. IEEE, 2023.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai, 2025. URL https://arxiv.org/abs/2410.00425.

Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What drives success in physical planning with joint-embedding predictive world models?, 2026. URL https://arxiv.org/abs/2512.24497.

Guodong Wang, Chenkai Zhang, Qingjie Liu, Jinjin Zhang, Jiancheng Cai, Junjie Liu, and Xinmin Liu. Libero-x: Robustness litmus for vision-language-action models, 2026a. URL https://arxiv.org/abs/2602.06556.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim G. J. Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning, 2026b. URL https://arxiv.org/abs/2603.12231.

Bowen Wen, Matthew Trepte, Joseph Aribido, Jan Kautz, Orazio Gallo, and Stan Birchfield. Foundationstereo: Zero-shot stereo matching, 2025. URL https://arxiv.org/abs/2501.09898.

Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: Vlm-in-the-loop policy steering via latent alignment. *arXiv preprint arXiv:2502.01828*, 2025.

Kechun Xu, Zhenjie Zhu, Anzhe Chen, Shuqi Zhao, Qing Huang, Yifei Yang, Haojian Lu, Rong Xiong, Masayoshi Tomizuka, and Yue Wang. Seeing to act, prompting to specify: A bayesian factorization of vision language action policy, 2025. URL https://arxiv.org/abs/2512.11218.

Shuai Yang, Hao Li, Yilun Chen, Bin Wang, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation, 2025. URL https://arxiv.org/abs/2507.17520.

Wentao Yuan, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. M2t2: Multi-task masked transformer for object-centric pick and place. *arXiv preprint arXiv:2311.00926*, 2023.

Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, Ethan Yin, Hao Wang, Hugo Zhou, James Wang, Lights Shi, Lucy Liang, Make Wang, Qian Wang, Roy Gan, Ryan Yu, Shalfun Li, Starrick Liu, Sylas Chen, Vincent Chen, and Zach Xu. Igniting vlms toward the embodied space, 2025. URL https://arxiv.org/abs/2509.11766.

Chuheng Zhang, Rushuai Yang, Xiaoyu Chen, Kaixin Wang, Li Zhao, Yi Chen, and Jiang Bian. How do vlas effectively inherit from vlms?, 2025. URL https://arxiv.org/abs/2511.06619.

Wancong Zhang, Basile Terver, Artem Zholus, Soham Chitnis, Harsh Sutaria, Mido Assran, Randall Balestriero, Amir Bar, Adrien Bardes, Yann LeCun, et al. Hierarchical planning with latent world models. *arXiv preprint arXiv:2604.03208*, 2026a.

Zeyu Zhang, Danning Li, Ian Reid, and Richard Hartley. Geoworld: Geometric world models, 2026b. URL https://arxiv.org/abs/2602.23058.

Wentao Zhao, Jiaming Chen, Ziyu Meng, Donghui Mao, Ran Song, and Wei Zhang. Vlmpc: Vision-language model predictive control for robotic manipulation, 2024. URL https://arxiv.org/abs/2407.09829.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025. URL https://arxiv.org/abs/2510.10274.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning, 2025a. URL https://arxiv.org/abs/2411.04983.

Junjie Zhou, Ke Mei, Lei Li, Tianyi Wang, Fengyun Rao, and Jing Lyu. WeMM-Embedding: WeChat multi-modal embedding technical report, 2026. URL https://arxiv.org/abs/2608.24053.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization, 2025b. URL https://arxiv.org/abs/2510.03827.

A Model and Training Details

Throughout the appendix, the *score* of a candidate means the cosine similarity used in Eq. 2, which is the negative of the planning cost. Candidates are ranked by score and never thresholded.

Architecture. With Qwen3-VL-Embedding-8B, the predictor P_θ is a standard transformer operating on the 1,620-token latent of a K=6-frame clip. The WEMM variant uses 336 tokens per clip with the same 4,096-dimensional token embeddings. Its configuration and training hyperparameters are in Table 4. The latent is the token-level internal visual representation of the embedding model, not the pooled vector, so e_t, p_t, and $\bar{e}_t$ share the same shape and p_t can be fed to the fixed readout B without projection layers. Fig. 8 contrasts RAT with the learned action-encoder variant ablated in Appendix B.

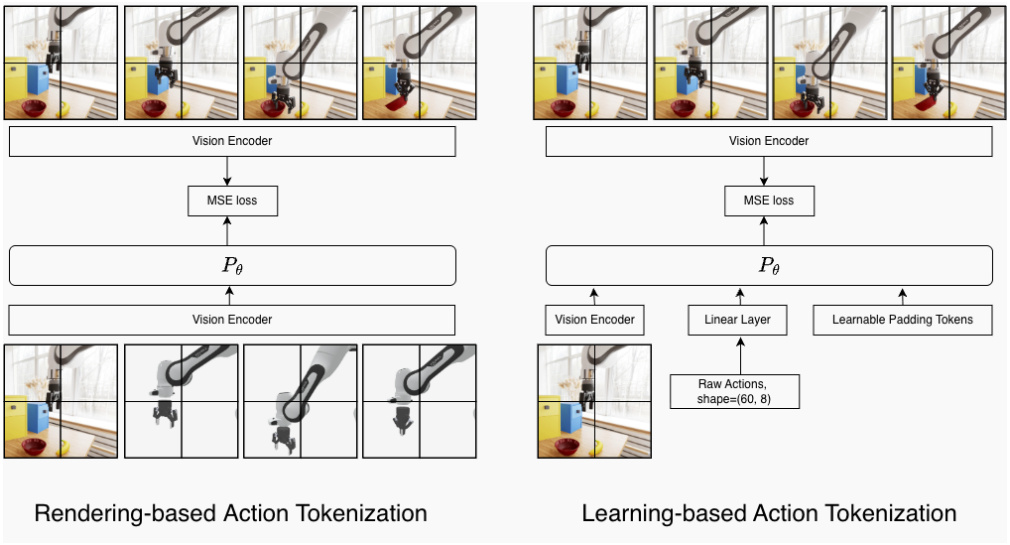


Figure 8: GWM (left) and its raw-action-conditioned variant (right). With RAT, actions are rendered into robot-only frames and read by the same frozen vision encoder as the observation. The variant instead encodes numeric actions with a learnable module. Captured images are exemplary.

WISER training corpus. The model of §4 is trained on the 1,728 WISER demonstrations, with six per training task. Control runs at 20 Hz, and episodes are capped at 120 steps. Training retains only the main-camera stream, discarding the language labels and wrist camera. The prediction horizon is $c = 60$ steps, subsampled to K=6 keyframes. Training runs for three epochs and takes 20 H100 GPU-hours.

Real-world training corpus. The model of §5-§7 is trained on two sources with no language labels: (i) the DROID manipulation corpus (Khazatsky et al., 2024) (74,604 episodes re-released at 15 fps), and (ii) the Franka pick-and-place subset of the MolmoB0T simulation corpus (Deshpande et al., 2026) (∼35k episodes with shoulder- and side-view streams). From DROID we keep the external camera streams for which the calibration could be recovered. Wrist cameras are excluded throughout. Each training example is a K=6-frame window spanning roughly three seconds, with log-uniform time-scale augmentation for robustness to different prediction horizons and color jitter on the full-RGB frames ($p = 0.5$). We hold out 2% of episodes for validation. Training runs 34k steps at 384 windows per step, about ten epochs, on 12 H100 GPUs.

Rendering. The same robot-only renderer produces the RAT frames for training data and inference, and given future joint states the robot mesh is rendered on a black background with the scoring camera's intrinsics and extrinsics. Only a kinematic model (URDF) and the camera calibration are needed, which is what makes RAT embodiment-agnostic. Concurrent work applies a related pixel-space action interface to a fine-tuned video generator under the name masked visual actions (Alzayer et al., 2026), expressing an action as a partially revealed trajectory in the frame.

Table 4: Transformer configuration and training hyperparameters of GWM.

Hyperparameter	**Value**
Architecture	
Hidden dimension (d_{model})	4096
FFN intermediate dimension (d_{ffn})	8192
Attention head dimension (d_{head})	128
Number of layers	5
Number of attention heads	32
Number of KV heads (GQA)	8
Input / Output dimension	4096
Input sequence length (Qwen / WEMM)	1620 / 336
Positional encoding	2D RoPE
Normalization	RMSNorm ($\epsilon = 10^{-5}$)
FFN activation	SwiGLU
Precision	bfloat16
Training	
Optimizer	Muon (Jordan, 2024) + Adam (Kingma & Ba, 2017)
Learning rate (Muon, hidden weights)	0.01
Learning rate (Adam, embed/head)	5×10^{-5}
Adam β	(0.9, 0.95)
Weight decay	0.01
LR scheduler	Cosine annealing
Min learning rate	10^{-6}
Epochs (WISER model)	3
Epochs (real-world model)	10
Gradient clipping	1.0
Loss function	MSE

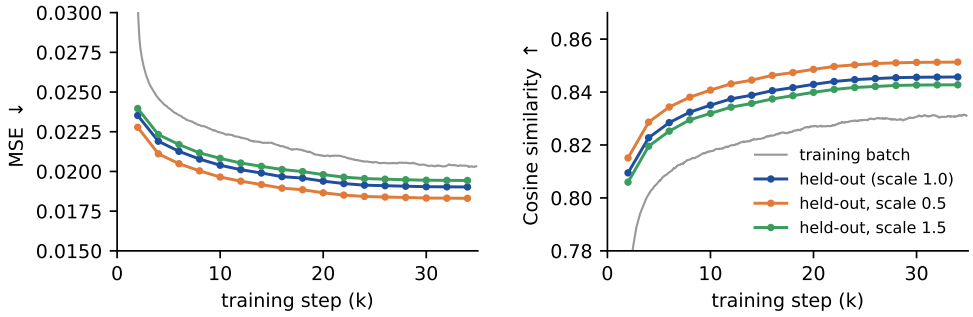


Figure 9: Held-out MSE and cosine similarity between predicted and ground-truth future latents over the real-world training run, evaluated every 2k steps on 256 held-out windows at time scales 1.0, 0.5, and 1.5. The gray curve is the training-batch value smoothed over 200 steps.

Training curves. Fig. 9 shows the real-world run. Held-out MSE falls from 0.0235 (step 2k) to 0.0190 (step 34k) and held-out cosine similarity between the predicted and ground-truth future latents rises from 0.809 to 0.846, improving monotonically at all 17 evaluation points, so the prediction steadily approaches the representation of the real future video, which is the property the planning cost relies on (§3.1). Held-out windows resampled at time scales 0.5 and 1.5 reach cosine 0.851 and 0.843, so the model predicts both faster and slower motion than the nominal window. Held-out cosine slightly exceeds the training-batch cosine of 0.830, whose batches carry color jitter and random time scales; these different augmentation conditions limit direct comparison of the two values.

Goal query. To obtain z_g, we feed a multimodal prompt into the embedding model. A retrieval-oriented system prompt s precedes the task instruction ℓ: *"Retrieve the video which can best finish*

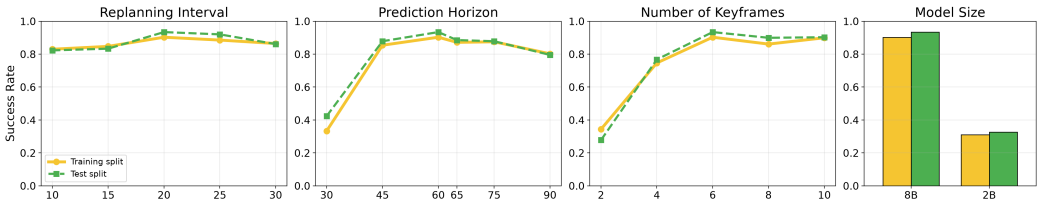


Figure 10: Oracle sweeps of the replanning interval, prediction horizon, and keyframe count on the WISER training and test splits, and the Oracle in the 8B and 2B embedding spaces. The training split selects a replanning interval of 20, $c = 60$, and $K = 6$, and the test split selects the same optimum in each sweep.

the manipulation task specified by the user, given the layout of the workspace and the current frame observation." This system prompt steers the model to produce embeddings that align task descriptions with future visual outcomes and discloses no task-specific information. The main text absorbs s into ℓ and writes $z_g = B(\ell, E(o_t))$; this appendix writes $B(s, \ell, E(o_t))$ where the prompt matters.

Verbatim instructions in the open-scene experiments. In the IsaacSim evaluation (§6) and on the real robot (§7), the instruction is used *verbatim*, giving $z_g = B(s, \ell, E(o_t))$ with the current frame from the scoring view and the same system prompt, and none of the sub-task decomposition or softmax normalization the preliminary experiments use (Appendix B.3). On the robot, pick and place are sequential sub-tasks, each receiving one grounding call to score its candidate trajectories.

B The WISER Testbed and Full Preliminary Results

B.1 Testbed Description

WISER (**W**orld-knowledge **I**ntegrated **S**emantic **E**mbodied **R**easoning) is described in §4 and Fig. 3. This section gives the collection protocol, the metrics, and the results held out of the main text.

The testbed is built in ManiSkill (Tao et al., 2025). Demonstrations are collected on training tasks only, with a Franka Panda driven by a privileged motion planner (Hao Su's Lab, 2024) at 20 Hz (simulation at 100 Hz). Six demonstrations per task with randomized initial states give 1,728 trajectories. We record the main camera (224×448), wrist camera (128×128), joint positions, gripper state, instruction, and actions. Episodes are capped at 120 steps (6 s). The dataset is released in LeRobot (Cadene et al., 2026) and RLDS (Ramos et al., 2021) formats. Task images are AI-generated to avoid copyright issues. The full task gallery is in Appendix G.

Metrics. *Grasp*: the correct cube is picked. *Reach*: the tool center point reaches the destination with near-zero speed, even if the grasp failed. *Success* = *Grasp* $\times$ *Reach*. Each task is evaluated once and metrics are averaged over the 288 tasks of a split.

B.2 Planning Hyperparameters

The Oracle evaluates whether the fixed readout can rank real future clips under each planning configuration, so it lets us select these hyperparameters independently of the trained predictor. We select them on the training split and report the test split alongside as a robustness check. Fig. 10 sweeps the replanning interval over $\{10, 15, 20, 25, 30\}$, the prediction horizon over $\{30, 45, 60, 65, 75, 90\}$, and the keyframe count over $\{2, 4, 6, 8, 10\}$, one at a time with the other two held at their defaults. The training split selects a replanning interval of 20, $c = 60$, and $K = 6$, and the test split selects the same optimum in each sweep. On the training split, the optimum sits on a plateau, since an interval of 20 leads 25 by 0.017 and $K = 6$ and $K = 10$ differ by a single episode. Short horizons and only 2-4 keyframes give lower success, consistent with the need for enough of the future motion to distinguish the intended outcome.

B.3 Goal Query and Prompt Decomposition

On this testbed the instruction ℓ follows the template *"Pick up the {X} and place it onto the {Y}"* and is split by pattern matching into ℓ_{pick} = *"Pick up the {X} from the table"* and ℓ_{place} = *"Place the grasped object to the {Y} on the table"*. Each sub-task prompt is embedded with the system prompt, the initial observation o_0, and the current observation o_t, giving $z_g^{\text{pick}} = B(s, \ell_{\text{pick}}, E(o_0), E(o_t))$ and $z_g^{\text{place}} = B(s, \ell_{\text{place}}, E(o_0), E(o_t))$. Candidates are scored with Eq. 2 against the active sub-goal, and the gripper's contact state selects which sub-goal is active, z_g^{pick} before the cube is grasped and z_g^{place} after.

Table 5: Prompt decomposition for GWM (Ours) and InstructVLA on WISER.

Prompt	Training Set			Test Set		
	Grasp	Reach	Success	Grasp	Reach	Success
GWM ℓ_{pick} + ℓ_{place}	0.97	0.95	0.92	0.99	0.88	0.87
GWM ℓ	0.93	0.88	0.82	0.92	0.76	0.73
InstructVLA ℓ_{pick} + ℓ_{place}	0.98	0.53	0.52	0.80	0.30	0.30
InstructVLA ℓ	0.98	0.92	0.89	0.79	0.51	0.47

Decomposing the instruction this way raises the test success of planning with GWM from 0.73 to 0.87, supporting the use of atomic sub-task queries for this testbed. Applying the same decomposition to the best VLA, InstructVLA, *drops* its performance on both splits (train $0.89 \rightarrow 0.52$, test $0.47 \rightarrow 0.30$), showing sensitivity to a change in prompt structure after fine-tuning (Table 5).

B.4 Embedding-Model Ablation

Table 6: The same Oracle planner built in different embedding spaces (WISER test split).

Embedding space	Grasp	Reach	Success
Qwen3-VL-Embedding-8B	1.00	0.93	**0.93**
WEMM-Embedding-9B (Zhou et al., 2026)	0.94	0.90	0.88
Qwen3-VL-Embedding-2B	0.67	0.40	0.33
Perception Encoder (Bolya et al., 2025)	-	-	0.00

Which spaces can host a grounded world model? With ground-truth futures supplied by the Oracle, the only variable is the embedding space. The 8B model reaches 93%, WEMM-Embedding-9B reaches 88%, and the 2B sibling drops to 33%. A GWM trained in the WEMM space with the unchanged recipe reaches 92% on the training split and 76% on the test split (Table 1), demonstrating that the method also works with another video-language embedding model. We use Qwen3-VL-Embedding-8B in the subsequent experiments because it gives higher WISER test success. The CLIP-style contrastive Perception Encoder completes 0% of the tasks under the same Oracle protocol (Table 6). These comparisons indicate that the embedding space's ability to rank robot futures is an important consideration when choosing a model.

B.5 The Form and Role of the Predictor

Three variants change whether P_θ is there at all, what space it predicts in, and what it predicts, holding the demonstrations, the twelve candidates, the embedding model, and the cost fixed.

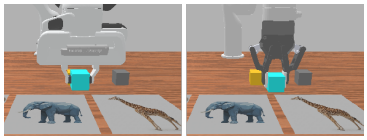
No prediction. Scoring the condition latent $e_t = E(o_t, R(a_{t:t+c}))$ directly, without mapping it forward, gives 8% training and 9% test success, close to the $1/12 = 8.3\%$ rate of picking one of the twelve candidates at random. The actions are not invisible to the encoder here, since the condition clip contains the rendered trajectory. What it does not contain is the scene's response. The rendered frames carry the robot alone, so the cube's new position, which is what the instruction is about, exists only in the predicted future.

Prediction in pixels. DreamDojo (Gao et al., 2026) is an action-conditioned video world model. We train it on the same demonstrations, generate each candidate's future video, re-encode that video with the same frozen embedding model, and score it with the same cosine, comparing a pixel-prediction pipeline with GWM's latent prediction under the same proposal and scoring protocol. It reaches 15% training and 17% test success. The generated videos are visually plausible but do not follow the commanded actions, so their re-encoded latents barely separate the twelve candidates. The substitution is also the expensive direction at planning time, since every candidate requires synthesizing a video and running the vision encoder over it, where GWM requires one forward pass of P_θ on a condition latent the encoder has already produced.

Prediction of the pooled embedding. The pooled-target variant keeps the 747M predictor and its training recipe and changes only the target and the head. The backbone output is mean-pooled over the 1,620 tokens and mapped by the same linear head to the pooled embedding $B(\bar{e}_t)$ that the frozen readout assigns to the real future clip. Training minimizes MSE against this unit-norm pooled target, and planning scores the predicted vector against z_g directly, so the readout is never applied to a prediction. It matches GWM on the training split at 92% and falls to 61% on the test split, with 0.91 Grasp and 0.66 Reach, compared with GWM's 0.99 and 0.88 (Table 8). The larger drop in Reach is consistent with the motivation for retaining spatial and temporal structure in the target, although this comparison does not isolate why the pooled target generalizes less well. The token-level target supervises the layout across the six keyframes, whereas the pooled target compresses the clip into one vector. Together, the three variants support predicting token-level future latents, which preserves the fixed readout interface and avoids video synthesis during planning. Table 8 gives every variant with all three metrics. The twelve candidates, the embedding model, and the cosine comparison are shared across these variants.

The no-prediction and pixel-prediction variants also show similar secondary metrics. Their test *Reach* is 0.44 for both, below GWM's 0.88, and their *Grasp* is 0.26 and 0.28, close to the one-in-four chance of selecting the referred cube. Executable proposals alone therefore do not ensure that the selected motion satisfies either part of the instruction.

B.6 Action Tokenization and Cross-Embodiment Generalization



The learned-action-encoder variant (Fig. 8) replaces RAT with a module that embeds raw numeric actions. It exhibits the same train-test gap as the VLAs (0.74 train, 0.24 test), while RAT's rendered actions align with the observation through the same vision encoder and generalize. RAT also enables zero-shot cross-embodiment transfer. We collect the 12 unique trajectories on an xArm6 (recording only joint positions and gripper state), render them with the xArm URDF, and reuse the Panda-trained GWM unchanged. Despite the different action space, kinematics, and appearance, planning succeeds on 87% of training and 83% of test tasks (Table 1).

B.7 Data Efficiency

The reduced-data variant $\frac{1}{2}\mathcal{D}$ uses 144 tasks from 12 of the 24 categories, with one demonstration per task; the fraction denotes category coverage rather than half of the full corpus's 1,728 trajectories. Planning with this GWM reaches 78% training and 72% test success, against 92% and 87% for the full corpus (Table 8). The loss is concentrated in *Reach* (0.74 test, against 0.88), while test *Grasp* is essentially unchanged (0.98 against 0.99), showing that the reduced corpus mainly affects placement success in this comparison. This result complements recent evidence on the data efficiency of retrieval-based planners (Dreczkowski et al., 2025); GWM learns to predict outcomes while action generation remains with the proposal mechanism.

B.8 Baseline Implementation Details

Table 7 summarizes the evaluated VLAs. The replanning interval balances feedback with rollout stability. Most baselines use 20 steps (1 s); Motus, GR00T-N1.6, and UniVLA use 40 to limit error accumulation, and InstructVLA uses 16 (Table 7). Across all models we follow the official fine-tuning setups (full-parameter vs. action-head vs. LoRA) and train until closed-loop performance plateaus on

the training scenes before zero-shot evaluation on the test scenes. We additionally tested OpenVLA-OFT (Kim et al., 2025), which overfits a single category (12 tasks) to 100% but degrades to 24% training success on the full set, so it is excluded. For GR00T we report the official GR00T-N1.6, which outperformed the LeRobot GR00T-N1.5 port, and exclude its wrist camera input, which hurt performance.

Table 7: Configuration of the evaluated VLAs.

Model	c	**Replan Interval**	**Dataloader**	**Implementation**	**Action**
Motus	48	40	LeRobot	Official	Absolute
XVLA	40	20	LeRobot	LeRobot	Absolute
GR00T-N1.6	40	40	LeRobot	Official	Relative
InstructVLA	16	16	RLDS	Official	Absolute
SmolVLA	20	20	LeRobot	LeRobot	Absolute
Wall-OSS	20	20	LeRobot	LeRobot	Absolute
$\pi_{0.5}$	20	20	LeRobot	LeRobot	Absolute
π_0	20	20	LeRobot	LeRobot	Absolute
InternVLA-A1	20	20	LeRobot	Official	Relative
UniVLA	40	40	RLDS	Official	Absolute

B.9 Full WISER Results

Table 8: Full WISER results for the ten fine-tuned VLAs of Fig. 4 and for the GWM variants and diagnostics, with Grasp and Reach alongside Success. Trainable models use the full demonstration corpus except the reduced-data variant, whose category coverage and demonstration count are specified in Appendix B.7. The GWM rows use $N{=}12$ retrieved candidates and the same cosine comparison. They use the same frozen embedding model except for the WEMM variant. GWM (Ours) is repeated from Table 1 for reference.

Method	H100 GPU Hours	Training Set			Test Set		
		Grasp	Reach	Success	Grasp	Reach	Success
GWM (Ours)	20	0.97	0.95	0.92	0.99	0.88	0.87
Fine-tuned VLAs							
InstructVLA (Yang et al., 2025)	70	0.98	0.92	0.89	0.79	0.51	0.47
SmolVLA (Shukor et al., 2025)	75	0.99	1.00	0.99	0.29	0.31	0.08
Wall-OSS (Zhai et al., 2025)	80	1.00	1.00	1.00	0.68	0.50	0.40
GR00T-N1.6 (NVIDIA et al., 2025)	100	1.00	1.00	1.00	0.72	0.18	0.18
InternVLA-A1 (Cai et al., 2026)	100	1.00	0.91	0.88	0.63	0.40	0.26
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	100	1.00	0.99	0.99	0.70	0.38	0.26
π_0 (Black et al., 2026)	100	1.00	1.00	1.00	0.47	0.14	0.08
XVLA (Zheng et al., 2025)	100	1.00	0.88	0.88	0.44	0.17	0.17
UniVLA (Bu et al., 2025)	120	0.79	0.62	0.63	0.38	0.18	0.13
Motus (Bi et al., 2025)	300	0.78	0.72	0.72	0.34	0.14	0.14
VLA Average	-	*0.95*	*0.90*	*0.90*	*0.54*	*0.29*	*0.22*
GWM variants and diagnostics							
GWM w/ action encoder	20	0.91	0.77	0.74	0.47	0.42	0.24
GWM w/ pooled target	-	0.97	0.94	0.92	0.91	0.66	0.61
GWM w/ WEMM	-	0.94	0.96	0.92	0.92	0.81	0.76
GWM w/ xArm6	-	0.96	0.91	0.87	0.97	0.86	0.83
GWM w/ $\frac{1}{2}\mathcal{D}$	20	0.97	0.81	0.78	0.98	0.74	0.72
No prediction	-	0.27	0.41	0.08	0.26	0.44	0.09
DreamDojo (video WM) (Gao et al., 2026)	24	0.22	0.41	0.15	0.28	0.44	0.17
Oracle	-	0.97	0.92	0.90	1.00	0.93	0.93

Fig. 4 in §4 summarizes the fine-tuned VLAs by their Success, and Table 8 gives the full Grasp/Reach/Success triples and the training cost of every model. All VLAs lose substantial success

on the test tasks despite these requiring only demonstrated motions. Some (SmolVLA, π_0) reach a near-perfect training success rate yet score below the $1/12 = 8.3\%$ chance rate of random trajectory retrieval at test. The consistent failure mode is grasping the wrong cube or placing onto a random image. The two best baselines, Wall-OSS and InstructVLA, are pretrained with auxiliary embodied-VQA tasks that help retain VLM knowledge. Even so, Appendix B.10 shows InstructVLA's base VLM localizes the correct destination in 81% of test scenarios before fine-tuning, while the fine-tuned policy reaches it in 51%. All VLAs show partial generalization at the cube-picking stage (average test Grasp 0.54), which requires recognizing unseen colors and referring expressions but no world knowledge. GR00T-N1.6 and InternVLA-A1 act in a relative action space and the other baselines in an absolute one, and the train-test gap appears in both groups. The baselines cover several VLA recipes (latent-action pretraining, knowledge insulation, mixture-of-transformers, VQA auxiliary tasks, video-action joint training), so the drop is observed across architectures. The training performance of planning with GWM (92%) sits below the best VLAs and close to the Oracle's 90%. The Oracle provides a reference for grounding with real future clips, rather than a strict upper bound; predicted futures score slightly higher on training tasks and lower on test tasks (87% versus 93%).

B.10 Visual Grounding of the Base VLM

To rule out that WISER's visuals are simply unrecognizable to pretrained VLMs, we probe the base model of the best VLA before fine-tuning. For each of the 24 test configurations we capture the initial observation and ask Eagle-2B (Li et al., 2025) which of the three destination images (left, middle, right) best matches the place referring expression. The base VLM localizes the correct destination in 81% of the 288 test scenarios, while after fine-tuning on OXE and the WISER training data, InstructVLA's reaching success is 51%. This shows that many WISER references are recognizable to the base VLM, while comparing localization with executed reaching does not isolate the effect of fine-tuning.

C Real-Data Probe Protocols

Shared CEM setup. The pointing and pushing probes in §5 use separate IsaacSim scenes and the same real-data GWM checkpoint as the subsequent system evaluations. A two-dimensional sample specifies the (x, y) endpoint of a target pose T^*, with a fixed height z_0 and downward orientation. The interpolator $\mathcal{I}$ expands each endpoint into a 31-waypoint trajectory from the home pose. Both probes refine 30 CEM samples over five iterations. CEM refines the endpoints using the scores of their predicted futures, and Fig. 5 visualizes the scoring landscapes.

C.1 Pointing

Four printed photos depict a dog, a panda, a banana, and a strawberry, and the instruction is "point at the image of the {name}." We report two distinct checks, whether the spatial map's lowest-cost position lies on the named photo (4/4) and whether CEM's best sample reaches it (4/4). Fig. 5 shows the cost maps and the paths of the Gaussian means.

C.2 Pushing

Three 4.7 cm cubes surround a gripper with a 2.4 cm-wide closed fingertip blade. For "push the {color} cube," the candidate slides the fingertip in a straight line to its sampled endpoint. We run 100 CEM trials per instruction with different seeds and execute each selected trajectory. A push succeeds when the named cube moves at least 3 cm, giving 153/300 successful pushes. All 153/153 successful pushes lie within $45°$ of the ideal sliding direction, with median angular errors of $3.6°$, $4.4°$, and $5.3°$ for the three instructions.

D IsaacSim Protocol

Scene and tasks. The evaluation uses one IsaacSim tabletop scene with a Franka arm, a banana, a cube, and a bowl for the pick tasks, plus a red and a green bin for the place tasks. Each task has five trials with a 90 s episode cap. Per-task results for every arm are in Table 9.

Table 9: Per-task results in the IsaacSim evaluation (successes out of 5 trials). Cam2 is the shipped scoring view used in §6. Cam1 is the symmetric opposite view. Fusion averages the two views' scores per candidate (Appendix F). The $\pi_{0.5}$ column uses the relaxed pick criterion of this appendix.

Task (referring expression)	Target	GWM cam2	GWM cam1	GWM fusion	TiPToP	$\pi_{0.5}$	V-JEPA 2-AC
"pick up the fruit"	banana	5/5	5/5	5/5	4/5	0/5	0/5
"…the yellow object"	banana	5/5	0/5	5/5	5/5	5/5	0/5
"…the thing you could eat"	banana	5/5	5/5	5/5	5/5	1/5	0/5
"…neither a toy nor a container"	banana	5/5	5/5	5/5	5/5	2/5	0/5
"…the puzzle toy"	cube	5/5	5/5	5/5	4/5	5/5	0/5
"…the most colorful object"	cube	5/5	5/5	5/5	4/5	5/5	0/5
"…closest to the bowl that is not red"	cube	5/5	5/5	5/5	4/5	4/5	0/5
"…you would eat a meal from"	bowl	5/5	5/5	5/5	5/5	3/5	5/5
"…between the cube and the banana"	bowl	5/5	5/5	5/5	5/5	0/5	5/5
"…the round container"	bowl	5/5	5/5	5/5	5/5	5/5	5/5
"put the block into the red box"	red bin	5/5	5/5	5/5	5/5	4/5	0/5
"…the green box"	green bin	5/5	5/5	5/5	5/5	3/5	5/5
"…the box that has the color of a tomato"	red bin	5/5	5/5	5/5	5/5	0/5	0/5
"…the box that has the color of grass"	green bin	5/5	5/5	5/5	5/5	0/5	5/5
Total		**70/70**	65/70	**70/70**	66/70	37/70	25/70

Judge. The same scripted judge evaluates every arm and was never retuned; the additional relaxed pick criterion reported for $\pi_{0.5}$ is defined below. Pick: the target's mesh center is at least 0.15 m above its settled height at episode end. Place: the block starts held in the gripper and succeeds when its mesh center is within 0.05 m (xy) of the named bin's center, with its height inside the bin's band, judged from the last pre-reset snapshot.

Proposal pipeline. Both systems lift wrist depth to a point cloud and fit the table plane with RANSAC. The GWM system then groups points above the surface into anonymous objects with DBSCAN; TiPToP's object segmentation is described in Appendix D.1. A grasp-synthesis network (Yuan et al., 2023) proposes grasps over the whole scene, and confidence-weighted sampling keeps 16 of them spread across the objects, so every object receives candidates regardless of the instruction. cuRobo (Sundaralingam et al., 2023) expands each grasp into a collision-free joint-space trajectory. A candidate is a sequence of joint waypoints plus gripper events.

Selection. Stage 1 picks the object whose candidates have the best aggregate score (mean over the object's candidates in simulation). Stage 2 picks the grasp within that object by the grasp network's confidence, subject to a feasibility gate. Placement candidates are built in the same way from geometry-based target poses on each perceived destination (Appendix D.1).

Fairness and caveats. The language-goal methods receive byte-identical instructions and scenes, and the goal-image baseline uses the corresponding scenes and supplied goal images. $\pi_{0.5}$ is the official DROID joint-position checkpoint, run zero-shot and closed-loop with no simulation fine-tuning. The comparison is on task outcome, not compute. This evaluation covers one scene, one layout, and 14 instructions at 5 trials each. We read the margins as "grounding is reliable in this scene", not as transferable percentages. The scoring view was chosen after inspecting both cameras (Appendix F).

$\pi_{0.5}$ under relaxed criteria. Under the official judge $\pi_{0.5}$ scores 7/70 (pick 0/50, place 7/20). Per-step diagnostics show $\pi_{0.5}$'s pick failures are grasp *retention*, not grounding, since it reaches for the correct object and produces real lifts that are then dropped, ending near table height. Under progressively relaxed pick criteria, end-state height above 0.10/0.05/0.01 m gives 0/2/5 of 50, and "ever lifted above 0.01 m during the episode" gives 30 of 50 trials (cube tasks 14/15). Combining the most relaxed pick criterion with the official place judge yields 37/70, 53%, the rate reported in §6. Its place failures on indirect color references ("the color of a tomato / grass", 0/10) carry the block toward a bin but drop it on the table, while direct color references mostly succeed (7/10).

Goal-image world-model baseline (V-JEPA 2-AC). V-JEPA 2-AC (Assran et al., 2025) plans toward a goal image with numeric end-effector actions. We supply both and score all trajectories in the same

16-candidate pools, with the same judges and two-stage selection rule. Its evaluation differs from the language-goal systems' in the following ways, stated explicitly. *Goals*: each task receives one goal image, the final scoring-camera frame of the target object's highest-confidence successful trajectory. All 16 candidate trajectories remain available for selection. *Actions*: each candidate's joint-waypoint plan is converted by forward kinematics into end-effector states at the model's training cadence (one action per 0.267 s). The resulting per-step deltas lie within the model's action cap. *Selection*: the candidate's actions are rolled out autoregressively in latent space from the current frame, the energy is the L1 distance between the predicted final frame's features and the goal image's features, and the object is chosen by the same two-stage rule as planning with GWM. The plain per-candidate argmin instead gives 10/70. The pick and place judges are unchanged from the main protocol.

D.1 The GWM and TiPToP Stacks Compared

Both stacks use wrist-depth geometry, RANSAC table fitting, DBSCAN point-cloud cleanup, convex hulls, M2T2 grasp sampling (Yuan et al., 2023), and cuRobo motion planning (Sundaralingam et al., 2023). These are shared components. The differences lie in how objects are identified, how grasps are selected, and how placement targets and plans are constructed. The comparison in §6 and §7 therefore evaluates complete systems.

Perception. Both stacks fit the table plane with RANSAC and use DBSCAN to clean its point cloud. TiPToP obtains object regions from VLM bounding boxes and segmentation masks. The GWM stack instead uses DBSCAN to group above-table points into anonymous objects. Both construct convex hulls for object geometry; the distinction is whether object identification requires semantic labels.

Grasp choice. Within the object that the GWM cost selects, the GWM stack executes the grasp with the highest M2T2 confidence, after a geometric feasibility check at the closing configuration that rejects grasps whose finger pads would meet too thin a slab or land off the object's center. The GWM cost selects the object by semantic agreement, and this second stage delegates grasp robustness to the grasp network. TiPToP selects a feasible grasp as part of its cuTAMP symbolic task plan, rather than applying this confidence-based selection after choosing an object.

Placement target. The GWM stack measures the object in hand from the point cloud, reads the landing surface of every perceived cluster from the raw points, and tells a hollow container from a solid block by the depth of its interior below the rim and by the rim enclosing that interior all around. A hollow destination receives candidates on its inner floor. Each candidate is a two-segment cuRobo plan, an approach above the rim followed by a constrained straight descent, with the destination disabled as an obstacle during the descent only and the closed-finger overhang subtracted from the clearance. Multiple offsets per destination are planned independently, and each successful plan becomes a candidate. TiPToP represents the destination by the convex hull of its mask, so an open box becomes a solid block whose inside cannot be told from its top, a limitation its documentation states. Placement is then a stability constraint on the shrunken bounding box of that hull, solved by cuTAMP with the destination kept as a collision obstacle throughout. A small open container can therefore admit no placement pose for TiPToP while the GWM stack lands inside it, which is consistent with the five no-plan outcomes on "place into the yellow box" in §7.

Execution. In simulation a GWM plan ends holding the object inside the destination, whereas a TiPToP plan releases it and returns home, and the judge accepts both end states. On hardware both systems release the object and return to the home pose.

D.2 Drawer Selection

Three single-drawer cabinets, a large muted-red one, a medium yellow one yawed 25°, and a small blue one, stand behind a block, a bowl, and a banana on the table. Six candidate trajectories are built with the same inverse-kinematics machinery, three drawer pulls, each comprising an approach, a knob grasp, and a pull along the slide axis, and three top-down grasps of the table objects that serve as distractors. Each drawer is referred to by two instructions of the form "open the drawer of the {attribute} cabinet", one naming its color (red, yellow, blue) and the other a size or position attribute (largest, middle, smallest). Each instruction is averaged over five phrasings. Planning with GWM selects the instructed drawer pull in 5/6 tasks, and replaying the selected pull in the simulator opens that drawer in each of them. The miss is "the smallest cabinet", where the pull of the middle

cabinet scores 0.022 above the pull of the instructed one; margins here are differences in cosine score between two candidate trajectories under the same instruction.

E REAL-ROBOT PROTOCOL

Rig. The rig uses a Franka Panda with a Robotiq 2F-140 gripper. A wrist-mounted RGB-D camera supplies the images from which FoundationStereo (Wen et al., 2025) reconstructs the scene geometry, and one external RGB camera provides the scoring view with its frame cropped to the workspace. The gripper is a Robotiq 2F-140, whereas most DROID episodes use the Robotiq 2F-85; the renderer simply uses the rig's own URDF, and no adaptation was needed.

Pipeline. The GWM pipeline follows Appendix D using the real wrist point cloud. The pipeline splits 16 candidates evenly across the discovered objects. Selection is two-stage with the object aggregate taken as the mean of its top-2 candidate scores, followed by the grasp-confidence choice and a feasibility gate. Because each sub-task receives one grounding call, the robot plans and executes the pick open-loop, re-observes, then plans and executes the place. The gripper state routes which sub-task is active. Trajectories are replayed at reduced speed with an acceleration cap.

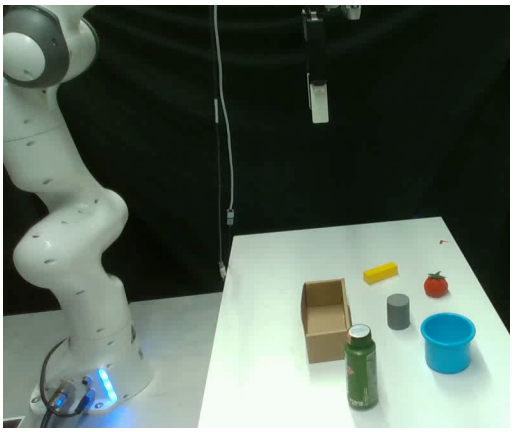
Scoring efficiency. Scoring 16 candidates takes about 4 s sequentially and 2.5 s in parallel. TiPToP's remote VLM call takes a median of 6.0 s per decision and ranges from 4.7 to 10.0 s over the 20 recorded scene-1 runs. The whole system (embedding model, predictor, perception, planner) runs on a single RTX 5090, with GPU memory use within 24 GB.

Success criteria and anomaly rule. Each pick or place sub-task is labeled separately by a human operator. The denominator of 60 is the total number of evaluated sub-tasks, comprising 30 picks and 30 placements across the two scenes, with five trials per instruction. A pick succeeds if the named object is grasped and held. A place succeeds if the carried object ends inside or on the named destination. Rounds in which the system never planned due to a sensor or driver fault are excluded for both methods. Every failure is attributed to *grounding* (wrong object or destination selected) or *planning* (no valid proposal, drop or unstable grasp during execution, no plan produced).

Table 10: Per-task real-robot results. Each row reports successes out of 5 trials of one pick or place sub-task; totals count the two stages separately.

Scene	Task	GWM best view	GWM shoulder view	TiPToP
1	pick the cylinder	5/5	5/5	5/5
1	place it inside the box	4/5	5/5	5/5
1	pick the object that can eat	5/5	5/5	5/5
1	place it inside the cup	4/5	3/5	5/5
1	pick the yellow object	5/5	4/5	5/5
1	place it in the container in the sky color	5/5	3/5	3/5
1	pick the drink	4/5	5/5	5/5
1	place it in the container made of paper	4/5	4/5	4/5
2	pick the shoe-cleaning tool	5/5	-	5/5
2	place it into the yellow box	4/5	-	0/5
2	pick the croissant	5/5	-	5/5
2	place it onto the plate	5/5	-	5/5
	Total	55/60	34/40 (scene 1)	52/60

Results. Table 10 lists every task. In scene 1 TiPToP scores 20/20 on the pick sub-tasks and 17/20 on the place sub-tasks. In scene 2, TiPToP's picks succeed while "place into the yellow box" produced no plan in all five trials (planning failures). The shoulder-view column is the camera-placement comparison of Appendix F, with the same system, same tasks, and original camera position (two excluded anomaly rounds). A supplementary video shows complete runs of both scenes.



(a) Reported view
36/40 successful sub-task trials



(b) Shoulder-height view
34/40 successful sub-task trials

Figure 11: Two scoring views of hardware scene 1, shown at the start of separate "pick up the cylinder" trials. Success counts cover all eight scene-1 pick and place instructions, with five trials each. Each evaluation uses one camera.

F Scoring View and Multi-View Fusion

The main-text system comparisons score each candidate from one external camera at a time, using cam2 in IsaacSim and the reported view on the robot. This appendix compares individual scoring views and reports a two-view fusion experiment in IsaacSim. The scoring view is a first-order factor. RAT renders the candidate with the scoring camera's calibration, so what the embedding model can read depends on what the camera can see. We evaluate whether averaging scores across views can recover errors caused by limited visibility from one camera.

Simulation. From the shipped view (cam2), planning with GWM selects correctly on all 14 tasks (70/70). From the symmetric opposite view (cam1), the same system scores 65/70, and the five failures are all on "the yellow object", where the banana appears small, distant, and shadowed by the gripper from that viewpoint. The selection error repeats deterministically across trials and is consistent with reduced target visibility from this view. Averaging the two views' scores per candidate corrects this error and yields 70/70 (Table 9, fusion column).

Real robot. Figure 11 shows the two scoring views used for the scene-1 hardware experiments. The same system completes 36/40 sub-task trials from the reported view and 34/40 from the shoulder-height view, with the same task set evaluated separately from each view. Table 10 gives the per-task results.

Multi-view fusion. Because RAT renders per view, the same candidate can be scored from several cameras and the scores averaged, with no retraining and no new parameters. The two-view fusion renders each candidate under both camera calibrations, scores each rendering against the instruction, and averages the two scores before object aggregation. Table 11 reports the object-selection margin, the aggregate score of the selected object minus the runner-up, for each view and their fusion. Because fusion precedes object aggregation, its object margin need not equal the mean of the single-view margins. On "yellow", cam1 selects the wrong object with a margin of $+0.002$, whereas fusion selects the correct object with a margin of $+0.053$. The two individual views select correctly on 13/14 and 14/14 tasks, while fusion selects correctly on all 14, with a minimum object-selection margin of $+0.015$. Fusion recovers the weaker view's error while preserving all correct selections from the stronger view. It does not consistently increase the margin over the better individual view. These results demonstrate multi-view fusion as a way to improve robustness to the choice of scoring camera.

Table 11: Object-selection margins per scoring view. The margin is the aggregate cosine of the best object minus the second best, and ✗ marks a wrong selection.

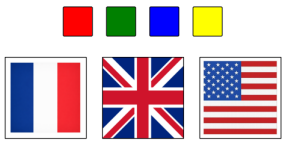
Task	cam2	cam1	fusion
fruit	$+0.104$	$+0.014$	$+0.057$
yellow	$+0.105$	$+0.002$ ✗	$+0.053$
eat	$+0.091$	$+0.018$	$+0.056$
negation	$+0.094$	$+0.019$	$+0.065$
puzzle	$+0.052$	$+0.076$	$+0.069$
colorful	$+0.054$	$+0.057$	$+0.062$
nearbowl	$+0.023$	$+0.016$	$+0.029$
eatfrom	$+0.028$	$+0.029$	$+0.031$
between	$+0.015$	$+0.005$	$+0.015$
container	$+0.018$	$+0.050$	$+0.042$
place_red	$+0.020$	$+0.039$	$+0.028$
place_green	$+0.084$	$+0.054$	$+0.072$
place_tomato	$+0.007$	$+0.029$	$+0.016$
place_grass	$+0.063$	$+0.041$	$+0.057$

G WISER TASK GALLERY

All training and test tasks of WISER are shown below. The images are AI-generated to avoid copyright issues.

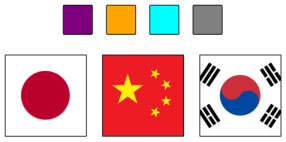
Category-0-country

Training



1. Pick up the red cube and place it onto the national flag of France.
2. Pick up the cube on the far left and place it onto the country that London belongs to.
3. Pick up the red cube and place it onto the national flag of the United States.
4. Pick up the cube to the right of the red cube and place it onto the country that Paris belongs to.
5. Pick up the green cube and place it onto the national flag of the United Kingdom.
6. Pick up the cube between the red and blue cubes and place it onto the country that New York belongs to.
7. Pick up the blue cube and place it onto the national flag of France.
8. Pick up the cube between the green and yellow cubes and place it onto the country that London belongs to.
9. Pick up the blue cube and place it onto the national flag of the United States.
10. Pick up the rightmost cube and place it onto the country that Paris belongs to.
11. Pick up the yellow cube and place it onto the national flag of the United Kingdom.
12. Pick up the last cube from the left and place it onto the country that New York belongs to.

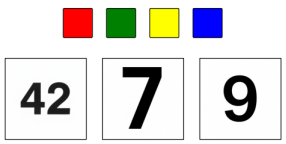
Test



1. Pick up the purple cube and place it onto the national flag of Japan.
2. Pick up the cube directly to the left of orange and place it onto the country that Beijing belongs to.
3. Pick up the purple cube and place it onto the national flag of South Korea.
4. Pick up the cube immediately next to the purple one and place it onto the country that Toyko belongs to.
5. Pick up the orange cube and place it onto the national flag of China.
6. Pick up the cube at position two from the left and place it onto the country that Seoul belongs to.
7. Pick up the cyan cube and place it onto the national flag of Japan.
8. Pick up the cube sandwiched between orange and grey and place it onto the country that Beijing belongs to.
9. Pick up the cyan cube and place it onto the national flag of South Korea.
10. Pick up the cube on the most right side and place it onto the country that Toyko belongs to.
11. Pick up the grey cube and place it onto the national flag of China.
12. Pick up the cube directly to the right of cyan and place it onto the country that Seoul belongs to.

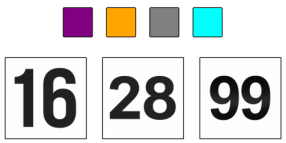
Category-1-number

Training



1. Pick up the cube on the far left and place it onto the image of number 42.
2. Pick up the red cube and place it onto the image of a prime number.
3. Pick up the leftmost cube and place it onto the image of number 9.
4. Pick up the green cube and place it onto the number of the answer of universe according to The Hitchhiker's Guide to the Galaxy.
5. Pick up the cube between the red and yellow cubes and place it onto the image of number 7.
6. Pick up the green cube and place it onto the result of 10 minus 1.
7. Pick up the cube between the green and blue cubes and place it onto the image of number 42.
8. Pick up the yellow cube and place it onto the image of a prime number.
9. Pick up the cube to the left of the blue cube and place it onto the image of number 9.
10. Pick up the blue cube and place it onto the number of the answer of universe according to The Hitchhiker's Guide to the Galaxy.
11. Pick up the last cube from the left and place it onto the image of number 7.
12. Pick up the blue cube and place it onto the result of 10 minus 1.

Test



1. Pick up the cube directly to the left of orange and place it onto the image of number 16.
2. Pick up the purple cube and place it onto the results of 4 multiplies 7.
3. Pick up the cube on the left side and place it onto the image of number 99.
4. Pick up the orange cube and place it onto the image of 2 raised to the power of 4.
5. Pick up the cube at position two from the left and place it onto the image of number 28.
6. Pick up the orange cube and place it onto the image of the figure close most to 100.
7. Pick up the cube sandwiched between orange and cyan and place it onto the image of number 16.
8. Pick up the grey cube and place it onto the results of 4 multiplies 7.
9. Pick up the cube directly to the left of cyan and place it onto the image of number 99.
10. Pick up the cyan cube and place it onto the image of 2 raised to the power of 4.
11. Pick up the cube directly to the right of grey and place it onto the image of number 28.
12. Pick up the cyan cube and place it onto the image of the figure close most to 100.

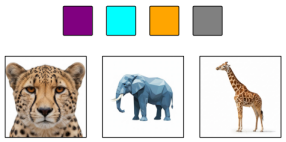
Category-2-animal

Training



1. Pick up the red cube and place it onto the image of a dog.
2. Pick up the leftmost cube and place it onto the animal eats bamboo.
3. Pick up the red cube and place it onto the image of a parrot.
4. Pick up the cube between the red and green cubes and place it onto the image of man's best friend.
5. Pick up the blue cube and place it onto the image of a panda.
6. Pick up the second cube from the left and place it onto the image of a tropical bird.
7. Pick up the green cube and place it onto the image of a dog.
8. Pick up the cube to the left of the yellow cube and place it onto the animal eats bamboo.
9. Pick up the green cube and place it onto the image of a parrot.
10. Pick up the last cube from the left and place it onto the image of man's best friend.
11. Pick up the yellow cube and place it onto the image of a panda.
12. Pick up the cube on the far right and place it onto the image of a tropical bird.

Test



1. Pick up the purple cube and place it onto the image of a cheetah.
2. Pick up the cube on the left side and place it onto the animal with a long nose.
3. Pick up the purple cube and place it onto the image of a giraffe.
4. Pick up the cube at position two from the left and place it onto the fastest land animal.
5. Pick up the cyan cube and place it onto the image of an elephant.
6. Pick up the cube sandwiched between purple and orange and place it onto the animal with a long neck.
7. Pick up the orange cube and place it onto the image of a cheetah.
8. Pick up the cube directly to the left of grey and place it onto the animal with a long nose.
9. Pick up the orange cube and place it onto the image of a giraffe.
10. Pick up the cube directly to the right of orange and place it onto the fastest land animal.
11. Pick up the grey cube and place it onto the image of an elephant.
12. Pick up the cube to the right of the orange cube and place it onto the animal with a long neck.

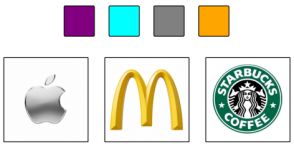
Category-3-logo

Training



1. Pick up the leftmost cube and place it onto the logo of Ferrari.
2. Pick up the red cube and place it onto the professional basketball league of North America.
3. Pick up the first cube from the left and place it onto the logo of the Olympics.
4. Pick up the blue cube and place it onto the famous hyper car producer.
5. Pick up the second cube from the left and place it onto the logo of the NBA.
6. Pick up the blue cube and place it onto the international sporting event happened every 4 years.
7. Pick up the cube to the left of the green cube and place it onto the logo of Ferrari.
8. Pick up the yellow cube and place it onto the professional basketball league of North America.
9. Pick up the second cube from the right and place it onto the logo of the Olympics.
10. Pick up the green cube and place it onto the famous hyper car producer.
11. Pick up the cube on the far right and place it onto the logo of the NBA.
12. Pick up the green cube and place it onto the international sporting event happened every 4 years.

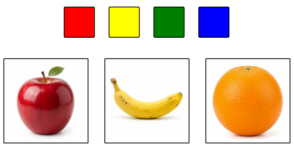
Test



1. Pick up the cube on the left side and place it onto the logo of Apple.
2. Pick up the purple cube and place it onto the fast food chain known for the golden arches.
3. Pick up the cube at position one from the left and place it onto the logo of Starbucks.
4. Pick up the cyan cube and place it onto the brand making IPhone and Macbook.
5. Pick up the cube sandwiched between purple and grey and place it onto the logo of McDonald's.
6. Pick up the cyan cube and place it onto the coffeehouse chain with the green mermaid logo.
7. Pick up the cube directly to the left of orange and place it onto the logo of Apple.
8. Pick up the grey cube and place it onto the fast food chain known for the golden arches.
9. Pick up the cube on the left side of orange and place it onto the logo of Starbucks.
10. Pick up the orange cube and place it onto the brand making IPhone and Macbook.
11. Pick up the cube to the right of the grey cube and place it onto the logo of McDonald's.
12. Pick up the orange cube and place it onto the coffeehouse chain with the green mermaid logo.

Category-4-fruit

Training



1. Pick up the red cube and place it onto the image of an apple.
2. Pick up the first cube from the left and place it onto the image of a yellow curved fruit.
3. Pick up the red cube and place it onto the image of an orange.
4. Pick up the second cube from the left and place it onto the image of a red and delicious fruit.
5. Pick up the yellow cube and place it onto the image of a banana.
6. Pick up the cube to the right of the red cube and place it onto the image of a round citrus fruit.
7. Pick up the green cube and place it onto the image of an apple.
8. Pick up the second cube from the right and place it onto the image of a yellow curved fruit.
9. Pick up the green cube and place it onto the image of an orange.
10. Pick up the cube on the far right and place it onto the image of a red and delicious fruit.
11. Pick up the blue cube and place it onto the image of a banana.
12. Pick up the rightmost cube and place it onto the image of a round citrus fruit.

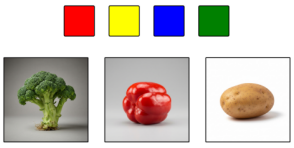
Test



1. Pick up the purple cube and place it onto the image of grapes.
2. Pick up the cube at position one from the left and place it onto the image of a tropical fruit with spiky leaves.
3. Pick up the purple cube and place it onto the image of a strawberry.
4. Pick up the cube sandwiched between purple and orange and place it onto the image of a bunch of small purple fruit.
5. Pick up the grey cube and place it onto the image of a pineapple.
6. Pick up the cube immediately next to the purple one and place it onto the image of a red berry with seeds on the outside.
7. Pick up the orange cube and place it onto the image of grapes.
8. Pick up the cube on the left side of cyan and place it onto the image of a tropical fruit with spiky leaves.
9. Pick up the orange cube and place it onto the image of a strawberry.
10. Pick up the cube to the right of the orange cube and place it onto the image of a bunch of small purple fruit.
11. Pick up the cyan cube and place it onto the image of a pineapple.
12. Pick up the cube on the most right side and place it onto the image of a red berry with seeds on the outside.

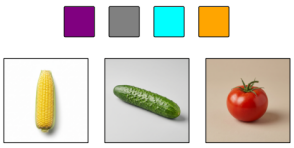
Category-5-vegetable

Training



1. Pick up the first cube from the left and place it onto the image of broccoli.
2. Pick up the red cube and place it onto the image of a colorful bell pepper.
3. Pick up the cube on the far left and place it onto the image of a potato.
4. Pick up the yellow cube and place it onto the image of a green vegetable looking like a small tree.
5. Pick up the cube to the right of the red cube and place it onto the image of a pepper.
6. Pick up the yellow cube and place it onto the image of the vegetable used to make french fries.
7. Pick up the second cube from the right and place it onto the image of broccoli.
8. Pick up the blue cube and place it onto the image of a colorful bell pepper.
9. Pick up the cube between the yellow and green cubes and place it onto the image of a potato.
10. Pick up the green cube and place it onto the image of a green vegetable looking like a small tree.
11. Pick up the rightmost cube and place it onto the image of a pepper.
12. Pick up the green cube and place it onto the image of the vegetable used to make french fries.

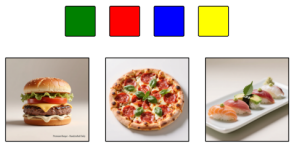
Test



1. Pick up the cube at position one from the left and place it onto the image of corn.
2. Pick up the purple cube and place it onto the image of a long green vegetable.
3. Pick up the cube directly to the left of grey and place it onto the image of a tomato.
4. Pick up the grey cube and place it onto the image of a yellow vegetable ear.
5. Pick up the cube immediately next to the purple one and place it onto the image of a cucumber.
6. Pick up the grey cube and place it onto the image of a red round vegetable.
7. Pick up the cube on the left side of orange and place it onto the image of corn.
8. Pick up the cyan cube and place it onto the image of a long green vegetable.
9. Pick up the cube sandwiched between grey and orange and place it onto the image of a tomato.
10. Pick up the orange cube and place it onto the image of a yellow vegetable ear.
11. Pick up the cube on the most right side and place it onto the image of a cucumber.
12. Pick up the orange cube and place it onto the image of a red round vegetable.

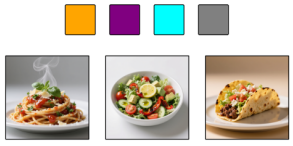
Category-6-food

Training



1. Pick up the green cube and place it onto the image of a burger.
2. Pick up the cube on the far left and place it onto the image of a round dish with cheese and toppings.
3. Pick up the green cube and place it onto the image of sushi.
4. Pick up the cube to the right of the green cube and place it onto the image of a sandwich with a meat patty.
5. Pick up the red cube and place it onto the image of a pizza.
6. Pick up the cube between the green and blue cubes and place it onto the image of a Japanese dish with rice and fish.
7. Pick up the blue cube and place it onto the image of a burger.
8. Pick up the cube between the red and yellow cubes and place it onto the image of a round dish with cheese and toppings.
9. Pick up the blue cube and place it onto the image of sushi.
10. Pick up the rightmost cube and place it onto the image of a sandwich with a meat patty.
11. Pick up the yellow cube and place it onto the image of a pizza.
12. Pick up the last cube from the left and place it onto the image of a Japanese dish with rice and fish.

Test



1. Pick up the orange cube and place it onto the image of pasta.
2. Pick up the cube directly to the left of purple and place it onto the image of a mix of vegetables.
3. Pick up the orange cube and place it onto the image of a taco.
4. Pick up the cube immediately next to the orange one and place it onto the image of an Italian noodle dish.
5. Pick up the purple cube and place it onto the image of a salad.
6. Pick up the cube at position two from the left and place it onto the image of a Mexican dish with a folded tortilla.
7. Pick up the cyan cube and place it onto the image of pasta.
8. Pick up the cube sandwiched between purple and grey and place it onto the image of a mix of vegetables.
9. Pick up the cyan cube and place it onto the image of a taco.
10. Pick up the cube on the most right side and place it onto the image of an Italian noodle dish.
11. Pick up the grey cube and place it onto the image of a salad.
12. Pick up the cube directly to the right of cyan and place it onto the image of a Mexican dish with a folded tortilla.

Category-7-electronics

Training



1. Pick up the cube on the far left and place it onto the image of a laptop.
2. Pick up the green cube and place it onto the image of a pointing device.
3. Pick up the leftmost cube and place it onto the image of a phone.
4. Pick up the red cube and place it onto the image of a portable computer.
5. Pick up the cube between the green and yellow cubes and place it onto the image of a computer mouse.
6. Pick up the red cube and place it onto the image of a mobile smartphone.
7. Pick up the cube between the red and blue cubes and place it onto the image of a laptop.
8. Pick up the yellow cube and place it onto the image of a pointing device.
9. Pick up the cube to the left of the blue cube and place it onto the image of a phone.
10. Pick up the blue cube and place it onto the image of a portable computer.
11. Pick up the last cube from the left and place it onto the image of a computer mouse.
12. Pick up the blue cube and place it onto the image of a mobile smartphone.

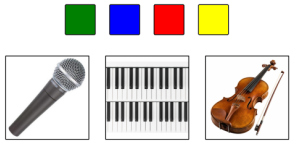
Test



1. Pick up the cube directly to the left of purple and place it onto the image of a camera.
2. Pick up the orange cube and place it onto the image of a device for listening to audio.
3. Pick up the cube on the left side and place it onto the image of a watch.
4. Pick up the purple cube and place it onto the image of a device for taking photos.
5. Pick up the cube at position two from the left and place it onto the image of headphones.
6. Pick up the purple cube and place it onto the image of a wrist timepiece.
7. Pick up the cube sandwiched between purple and cyan and place it onto the image of a camera.
8. Pick up the grey cube and place it onto the image of a device for listening to audio.
9. Pick up the cube directly to the left of cyan and place it onto the image of a watch.
10. Pick up the cyan cube and place it onto the image of a device for taking photos.
11. Pick up the cube directly to the right of grey and place it onto the image of headphones.
12. Pick up the cyan cube and place it onto the image of a wrist timepiece.

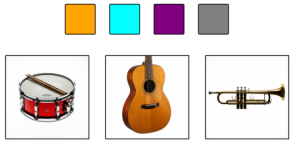
Category-8-instruments

Training



1. Pick up the green cube and place it onto the image of a microphone.
2. Pick up the leftmost cube and place it onto the image of an instrument with black and white keys.
3. Pick up the green cube and place it onto the image of a violin.
4. Pick up the cube between the green and red cubes and place it onto the image of an instrument for singing or recording.
5. Pick up the blue cube and place it onto the image of a piano keyboard.
6. Pick up the second cube from the left and place it onto the image of an instrument played with a bow.
7. Pick up the red cube and place it onto the image of a microphone.
8. Pick up the cube to the left of the yellow cube and place it onto the image of an instrument with black and white keys.
9. Pick up the red cube and place it onto the image of a violin.
10. Pick up the last cube from the left and place it onto the image of an instrument for singing or recording.
11. Pick up the yellow cube and place it onto the image of a piano keyboard.
12. Pick up the cube on the far right and place it onto the image of an instrument played with a bow.

Test



1. Pick up the orange cube and place it onto the image of a drum.
2. Pick up the cube on the left side and place it onto the image of a six-stringed instrument.
3. Pick up the orange cube and place it onto the image of a trumpet.
4. Pick up the cube at position two from the left and place it onto the image of a percussion instrument played with sticks.
5. Pick up the cyan cube and place it onto the image of a guitar.
6. Pick up the cube sandwiched between orange and purple and place it onto the image of a brass instrument with three valves.
7. Pick up the purple cube and place it onto the image of a drum.
8. Pick up the cube directly to the left of grey and place it onto the image of a six-stringed instrument.
9. Pick up the purple cube and place it onto the image of a trumpet.
10. Pick up the cube directly to the right of purple and place it onto the image of a percussion instrument played with sticks.
11. Pick up the grey cube and place it onto the image of a guitar.
12. Pick up the cube to the right of the purple cube and place it onto the image of a brass instrument with three valves.

Category-9-furniture

Training



1. Pick up the leftmost cube and place it onto the image of a chair.
2. Pick up the green cube and place it onto the long comfortable seat.
3. Pick up the first cube from the left and place it onto the image of a table.
4. Pick up the blue cube and place it onto the furniture for sitting.
5. Pick up the second cube from the left and place it onto the image of a sofa.
6. Pick up the blue cube and place it onto the furniture with a flat top.
7. Pick up the cube to the left of the red cube and place it onto the image of a chair.
8. Pick up the yellow cube and place it onto the long comfortable seat.
9. Pick up the second cube from the right and place it onto the image of a table.
10. Pick up the red cube and place it onto the furniture for sitting.
11. Pick up the cube on the far right and place it onto the image of a sofa.
12. Pick up the red cube and place it onto the furniture with a flat top.

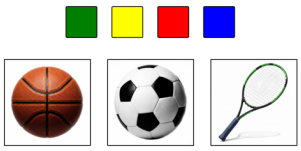
Test



1. Pick up the cube on the left side and place it onto the image of a bed.
2. Pick up the orange cube and place it onto the furniture for storage.
3. Pick up the cube at position one from the left and place it onto the image of a toilet.
4. Pick up the cyan cube and place it onto the furniture for sleeping.
5. Pick up the cube sandwiched between orange and grey and place it onto the image of a cabinet.
6. Pick up the cyan cube and place it onto the object usually in washroom.
7. Pick up the cube directly to the left of purple and place it onto the image of a bed.
8. Pick up the grey cube and place it onto the furniture for storage.
9. Pick up the cube on the left side of purple and place it onto the image of a toilet.
10. Pick up the purple cube and place it onto the furniture for sleeping.
11. Pick up the cube to the right of the grey cube and place it onto the image of a cabinet.
12. Pick up the purple cube and place it onto the object usually in washroom.

Category-10-sports

Training



1. Pick up the green cube and place it onto the image of a basketball.
2. Pick up the first cube from the left and place it onto the image of a ball used in soccer matches.
3. Pick up the green cube and place it onto the image of a tennis racket.
4. Pick up the second cube from the left and place it onto the image of an orange ball used in professional basketball.
5. Pick up the yellow cube and place it onto the image of a soccer ball.
6. Pick up the cube to the right of the green cube and place it onto the image of the equipment for playing tennis.
7. Pick up the red cube and place it onto the image of a basketball.
8. Pick up the second cube from the right and place it onto the image of a ball used in soccer matches.
9. Pick up the red cube and place it onto the image of a tennis racket.
10. Pick up the cube on the far right and place it onto the image of an orange ball used in professional basketball.
11. Pick up the blue cube and place it onto the image of a soccer ball.
12. Pick up the rightmost cube and place it onto the image of the equipment for playing tennis.

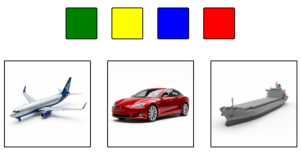
Test



1. Pick up the orange cube and place it onto the image of a boxing glove.
2. Pick up the cube at position one from the left and place it onto the image of an oval ball used in American football.
3. Pick up the orange cube and place it onto the image of a ping-pong racket.
4. Pick up the cube sandwiched between orange and purple and place it onto the image of a hand-wearing equipment.
5. Pick up the grey cube and place it onto the image of a football (NOT soccer!).
6. Pick up the cube immediately next to the orange one and place it onto the image of a equipment used for table tennis.
7. Pick up the purple cube and place it onto the image of a boxing glove.
8. Pick up the cube on the left side of cyan and place it onto the image of an oval ball used in American football.
9. Pick up the purple cube and place it onto the image of a ping-pong racket.
10. Pick up the cube to the right of the purple cube and place it onto the image of a hand-wearing equipment.
11. Pick up the cyan cube and place it onto the image of a football (NOT soccer!).
12. Pick up the cube on the most right side and place it onto the image of a equipment used for table tennis.

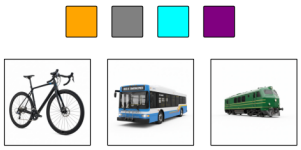
Category-11-vehicles

Training



1. Pick up the first cube from the left and place it onto the image of an airplane.
2. Pick up the green cube and place it onto the image of a four-wheeled vehicle for roads.
3. Pick up the cube on the far left and place it onto the image of a ship.
4. Pick up the yellow cube and place it onto the image of a vehicle that flies in the sky.
5. Pick up the cube to the right of the green cube and place it onto the image of a car.
6. Pick up the yellow cube and place it onto the image of a vehicle that travels on water.
7. Pick up the second cube from the right and place it onto the image of an airplane.
8. Pick up the blue cube and place it onto the image of a four-wheeled vehicle for roads.
9. Pick up the cube between the yellow and red cubes and place it onto the image of a ship.
10. Pick up the red cube and place it onto the image of a vehicle that flies in the sky.
11. Pick up the rightmost cube and place it onto the image of a car.
12. Pick up the red cube and place it onto the image of a vehicle that travels on water.

Test



1. Pick up the cube at position one from the left and place it onto the image of a bicycle.
2. Pick up the orange cube and place it onto the image of a large public transport vehicle.
3. Pick up the cube directly to the left of grey and place it onto the image of a green train.
4. Pick up the grey cube and place it onto the image of a two-wheeled pedal vehicle.
5. Pick up the cube immediately next to the orange one and place it onto the image of a bus.
6. Pick up the grey cube and place it onto the image of a vehicle that runs on tracks.
7. Pick up the cube on the left side of purple and place it onto the image of a bicycle.
8. Pick up the cyan cube and place it onto the image of a large public transport vehicle.
9. Pick up the cube sandwiched between grey and purple and place it onto the image of a green train.
10. Pick up the purple cube and place it onto the image of a two-wheeled pedal vehicle.
11. Pick up the cube on the most right side and place it onto the image of a bus.
12. Pick up the purple cube and place it onto the image of a vehicle that runs on tracks.

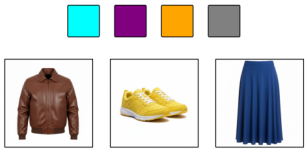
Category-12-clothing

Training



1. Pick up the blue cube and place it onto the image of a pink hat.
2. Pick up the cube on the far left and place it onto the image of pants made of denim.
3. Pick up the blue cube and place it onto the image of a t-shirt.
4. Pick up the cube to the right of the blue cube and place it onto the image of an accessory worn on the head.
5. Pick up the red cube and place it onto the image of a pair of jeans.
6. Pick up the cube between the blue and green cubes and place it onto the image of a short-sleeved casual top.
7. Pick up the green cube and place it onto the image of a pink hat.
8. Pick up the cube between the red and yellow cubes and place it onto the image of pants made of denim.
9. Pick up the green cube and place it onto the image of a t-shirt.
10. Pick up the rightmost cube and place it onto the image of an accessory worn on the head.
11. Pick up the yellow cube and place it onto the image of a pair of jeans.
12. Pick up the last cube from the left and place it onto the image of a short-sleeved casual top.

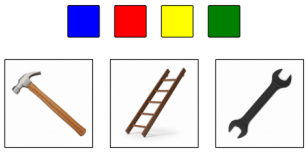
Test



1. Pick up the cyan cube and place it onto the image of a jacket.
2. Pick up the cube directly to the left of purple and place it onto the image of yellow footwear.
3. Pick up the cyan cube and place it onto the image of a skirt.
4. Pick up the cube immediately next to the cyan one and place it onto the image of outerwear for the upper body.
5. Pick up the purple cube and place it onto the image of a pair of shoes.
6. Pick up the cube at position two from the left and place it onto the one-piece garment that girls love.
7. Pick up the orange cube and place it onto the image of a jacket.
8. Pick up the cube sandwiched between purple and grey and place it onto the image of yellow footwear.
9. Pick up the orange cube and place it onto the image of a skirt.
10. Pick up the cube on the most right side and place it onto the image of outerwear for the upper body.
11. Pick up the grey cube and place it onto the image of a pair of shoes.
12. Pick up the cube directly to the right of orange and place it onto the one-piece garment that girls love.

Category-13-tools

Training



1. Pick up the cube on the far left and place it onto the image of a hammer.
2. Pick up the blue cube and place it onto the image of a tool used to climb.
3. Pick up the leftmost cube and place it onto the image of a wrench.
4. Pick up the red cube and place it onto the image of a tool used to drive nails.
5. Pick up the cube between the blue and yellow cubes and place it onto the image of a ladder.
6. Pick up the red cube and place it onto the image of a tool used to turn nuts and bolts.
7. Pick up the cube between the red and green cubes and place it onto the image of a hammer.
8. Pick up the yellow cube and place it onto the image of a tool used to climb.
9. Pick up the cube to the left of the green cube and place it onto the image of a wrench.
10. Pick up the green cube and place it onto the image of a tool used to drive nails.
11. Pick up the last cube from the left and place it onto the image of a ladder.
12. Pick up the green cube and place it onto the image of a tool used to turn nuts and bolts.

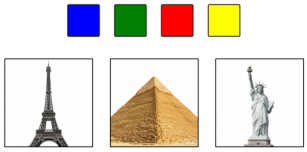
Test



1. Pick up the cube directly to the left of purple and place it onto the image of a broom.
2. Pick up the cyan cube and place it onto the image of a power tool for making holes.
3. Pick up the cube on the left side and place it onto the image of pliers.
4. Pick up the purple cube and place it onto the image of a tool used to clean floor.
5. Pick up the cube at position two from the left and place it onto the image of a drill.
6. Pick up the purple cube and place it onto the image of a tool for gripping objects.
7. Pick up the cube sandwiched between purple and orange and place it onto the image of a broom.
8. Pick up the grey cube and place it onto the image of a power tool for making holes.
9. Pick up the cube directly to the left of orange and place it onto the image of pliers.
10. Pick up the orange cube and place it onto the image of a tool used to clean floor.
11. Pick up the cube directly to the right of grey and place it onto the image of a drill.
12. Pick up the orange cube and place it onto the image of a tool for gripping objects.

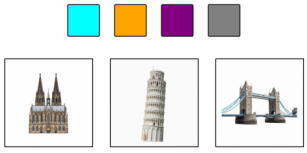
Category-14-landscape

Training



1. Pick up the blue cube and place it onto the image of the Eiffel Tower.
2. Pick up the leftmost cube and place it onto the image of the ancient structures in Giza.
3. Pick up the blue cube and place it onto the image of the Statue of Liberty.
4. Pick up the cube between the blue and red cubes and place it onto the image of the famous iron tower in Paris.
5. Pick up the green cube and place it onto the image of the Pyramids.
6. Pick up the second cube from the left and place it onto the image of the famous statue in New York harbor.
7. Pick up the red cube and place it onto the image of the Eiffel Tower.
8. Pick up the cube to the left of the yellow cube and place it onto the image of the ancient structures in Giza.
9. Pick up the red cube and place it onto the image of the Statue of Liberty.
10. Pick up the last cube from the left and place it onto the image of the famous iron tower in Paris.
11. Pick up the yellow cube and place it onto the image of the Pyramids.
12. Pick up the cube on the far right and place it onto the image of the famous statue in New York harbor.

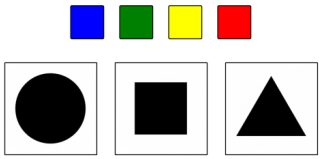
Test



1. Pick up the cyan cube and place it onto the image of the cologne church.
2. Pick up the cube on the left side and place it onto the image of the leaning tower in Italy.
3. Pick up the cyan cube and place it onto the image of Twoer Bridge.
4. Pick up the cube at position two from the left and place it onto the image of the building in German.
5. Pick up the orange cube and place it onto the image of the pisa tower.
6. Pick up the cube sandwiched between cyan and purple and place it onto the image of the famous building in London.
7. Pick up the purple cube and place it onto the image of the cologne church.
8. Pick up the cube directly to the left of grey and place it onto the image of the leaning tower in Italy.
9. Pick up the purple cube and place it onto the image of Twoer Bridge.
10. Pick up the cube directly to the right of purple and place it onto the image of the building in German.
11. Pick up the grey cube and place it onto the image of the pisa tower.
12. Pick up the cube to the right of the purple cube and place it onto the image of the famous building in London.

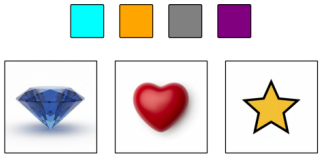
Category-15-shapes

Training



1. Pick up the leftmost cube and place it onto the image of a circle.
2. Pick up the blue cube and place it onto the image of a shape with four equal sides.
3. Pick up the first cube from the left and place it onto the image of a triangle.
4. Pick up the green cube and place it onto the image of a round shape with no corners.
5. Pick up the second cube from the left and place it onto the image of a square.
6. Pick up the green cube and place it onto the image of a shape with three sides.
7. Pick up the cube to the left of the red cube and place it onto the image of a circle.
8. Pick up the yellow cube and place it onto the image of a shape with four equal sides.
9. Pick up the second cube from the right and place it onto the image of a triangle.
10. Pick up the red cube and place it onto the image of a round shape with no corners.
11. Pick up the cube on the far right and place it onto the image of a square.
12. Pick up the red cube and place it onto the image of a shape with three sides.

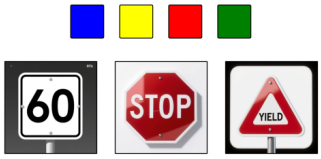
Test



1. Pick up the cube on the left side and place it onto the image of a diamond.
2. Pick up the cyan cube and place it onto the image of a symbol of love.
3. Pick up the cube at position one from the left and place it onto the image of a star.
4. Pick up the orange cube and place it onto the image of a rhombus shape.
5. Pick up the cube sandwiched between cyan and grey and place it onto the image of a heart.
6. Pick up the orange cube and place it onto the image of a shape with five points.
7. Pick up the cube directly to the left of purple and place it onto the image of a diamond.
8. Pick up the grey cube and place it onto the image of a symbol of love.
9. Pick up the cube on the left side of purple and place it onto the image of a star.
10. Pick up the purple cube and place it onto the image of a rhombus shape.
11. Pick up the cube to the right of the grey cube and place it onto the image of a heart.
12. Pick up the purple cube and place it onto the image of a shape with five points.

Category-16-traffic_sign

Training



1. Pick up the blue cube and place it onto the image of a speed limit sign.
2. Pick up the first cube from the left and place it onto the red octagon sign.
3. Pick up the blue cube and place it onto the image of a yield sign.
4. Pick up the second cube from the left and place it onto the sign showing max speed.
5. Pick up the yellow cube and place it onto the image of a stop sign.
6. Pick up the cube to the right of the blue cube and place it onto the inverted triangle sign.
7. Pick up the red cube and place it onto the image of a speed limit sign.
8. Pick up the second cube from the right and place it onto the red octagon sign.
9. Pick up the red cube and place it onto the image of a yield sign.
10. Pick up the cube on the far right and place it onto the sign showing max speed.
11. Pick up the green cube and place it onto the image of a stop sign.
12. Pick up the rightmost cube and place it onto the inverted triangle sign.

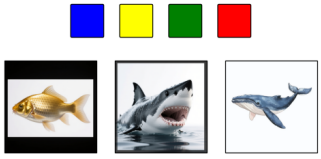
Test



1. Pick up the cyan cube and place it onto the image of a no entry sign.
2. Pick up the cube at position one from the left and place it onto the warning sign of pedestrian crossing.
3. Pick up the cyan cube and place it onto the image of a traffic light.
4. Pick up the cube sandwiched between cyan and purple and place it onto the red circle with white bar.
5. Pick up the grey cube and place it onto the image of crosswalk warning sign.
6. Pick up the cube immediately next to the cyan one and place it onto the signal with red yellow green.
7. Pick up the purple cube and place it onto the image of a no entry sign.
8. Pick up the cube on the left side of orange and place it onto the warning sign of pedestrian crossing.
9. Pick up the purple cube and place it onto the image of a traffic light.
10. Pick up the cube to the right of the purple cube and place it onto the red circle with white bar.
11. Pick up the orange cube and place it onto the image of crosswalk warning sign.
12. Pick up the cube on the most right side and place it onto the signal with red yellow green.

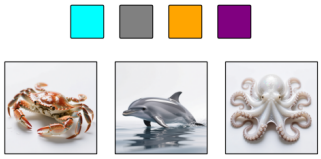
Category-17-animal_extra

Training



1. Pick up the first cube from the left and place it onto the image of a golden fish.
2. Pick up the blue cube and place it onto the predator of the ocean.
3. Pick up the cube on the far left and place it onto the image of a whale.
4. Pick up the yellow cube and place it onto the popular aquarium pet.
5. Pick up the cube to the right of the blue cube and place it onto the image of a shark.
6. Pick up the yellow cube and place it onto the largest mammal in the sea.
7. Pick up the second cube from the right and place it onto the image of a golden fish.
8. Pick up the green cube and place it onto the predator of the ocean.
9. Pick up the cube between the yellow and red cubes and place it onto the image of a whale.
10. Pick up the red cube and place it onto the popular aquarium pet.
11. Pick up the rightmost cube and place it onto the image of a shark.
12. Pick up the red cube and place it onto the largest mammal in the sea.

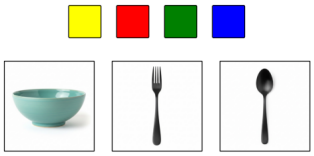
Test



1. Pick up the cube at position one from the left and place it onto the image of a crab.
2. Pick up the cyan cube and place it onto the intelligent marine mammal.
3. Pick up the cube directly to the left of grey and place it onto the image of an octopus.
4. Pick up the grey cube and place it onto the sea creature with claws.
5. Pick up the cube immediately next to the cyan one and place it onto the image of a dolphin.
6. Pick up the grey cube and place it onto the sea creature with 8 arms.
7. Pick up the cube on the left side of purple and place it onto the image of a crab.
8. Pick up the orange cube and place it onto the intelligent marine mammal.
9. Pick up the cube sandwiched between grey and purple and place it onto the image of an octopus.
10. Pick up the purple cube and place it onto the sea creature with claws.
11. Pick up the cube on the most right side and place it onto the image of a dolphin.
12. Pick up the purple cube and place it onto the sea creature with 8 arms.

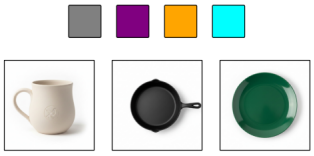
Category-18-kitchenware

Training



1. Pick up the yellow cube and place it onto the image of a bowl.
2. Pick up the cube on the far left and place it onto the image of a utensil with prongs.
3. Pick up the yellow cube and place it onto the image of a spoon.
4. Pick up the cube to the right of the yellow cube and place it onto the image of a container for soup.
5. Pick up the red cube and place it onto the image of a fork.
6. Pick up the cube between the yellow and green cubes and place it onto the image of a utensil for eating soup.
7. Pick up the green cube and place it onto the image of a bowl.
8. Pick up the cube between the red and blue cubes and place it onto the image of a utensil with prongs.
9. Pick up the green cube and place it onto the image of a spoon.
10. Pick up the rightmost cube and place it onto the image of a container for soup.
11. Pick up the blue cube and place it onto the image of a fork.
12. Pick up the last cube from the left and place it onto the image of a utensil for eating soup.

Test



1. Pick up the grey cube and place it onto the image of a cup.
2. Pick up the cube directly to the left of purple and place it onto the image of cookware for frying.
3. Pick up the grey cube and place it onto the image of a plate.
4. Pick up the cube immediately next to the grey one and place it onto the image of a container for drinking.
5. Pick up the purple cube and place it onto the image of a pan.
6. Pick up the cube at position two from the left and place it onto the image of a flat dish for food.
7. Pick up the orange cube and place it onto the image of a cup.
8. Pick up the cube sandwiched between purple and cyan and place it onto the image of cookware for frying.
9. Pick up the orange cube and place it onto the image of a plate.
10. Pick up the cube on the most right side and place it onto the image of a container for drinking.
11. Pick up the cyan cube and place it onto the image of a pan.
12. Pick up the cube directly to the right of orange and place it onto the image of a flat dish for food.

Category-19-weather

Training



1. Pick up the cube on the far left and place it onto the image of a cloud.
2. Pick up the yellow cube and place it onto the image of the symbol for snowy weather.
3. Pick up the leftmost cube and place it onto the image of the sun.
4. Pick up the red cube and place it onto the image of the symbol for cloudy weather.
5. Pick up the cube between the yellow and blue cubes and place it onto the image of a snowflake.
6. Pick up the red cube and place it onto the image of the symbol for sunny weather.
7. Pick up the cube between the red and green cubes and place it onto the image of a cloud.
8. Pick up the blue cube and place it onto the image of the symbol for snowy weather.
9. Pick up the cube to the left of the green cube and place it onto the image of the sun.
10. Pick up the green cube and place it onto the image of the symbol for cloudy weather.
11. Pick up the last cube from the left and place it onto the image of a snowflake.
12. Pick up the green cube and place it onto the image of the symbol for sunny weather.

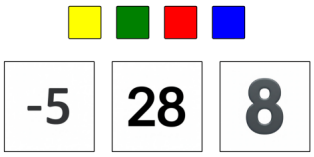
Test



1. Pick up the cube directly to the left of purple and place it onto the image of a lightning bolt.
2. Pick up the grey cube and place it onto the image of the symbol for rainy weather.
3. Pick up the cube on the left side and place it onto the image of a wind icon.
4. Pick up the purple cube and place it onto the image of the symbol for stormy weather.
5. Pick up the cube at position two from the left and place it onto the image of a rain cloud.
6. Pick up the purple cube and place it onto the image of the symbol for windy weather.
7. Pick up the cube sandwiched between purple and orange and place it onto the image of a lightning bolt.
8. Pick up the cyan cube and place it onto the image of the symbol for rainy weather.
9. Pick up the cube directly to the left of orange and place it onto the image of a wind icon.
10. Pick up the orange cube and place it onto the image of the symbol for stormy weather.
11. Pick up the cube directly to the right of cyan and place it onto the image of a rain cloud.
12. Pick up the orange cube and place it onto the image of the symbol for windy weather.

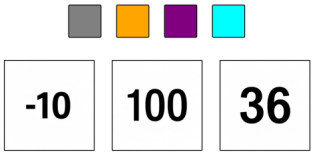
Category-20-num_extra

Training



1. Pick up the yellow cube and place it onto the image of number -5.
2. Pick up the leftmost cube and place it onto the result of 4 multiplies 7.
3. Pick up the yellow cube and place it onto the image of number 8.
4. Pick up the cube between the yellow and red cubes and place it onto the image of result of 1 minus 6.
5. Pick up the green cube and place it onto the image of number 28.
6. Pick up the second cube from the left and place it onto the result of 2^3.
7. Pick up the red cube and place it onto the image of number -5.
8. Pick up the cube to the left of the blue cube and place it onto the result of 4 multiplies 7.
9. Pick up the red cube and place it onto the image of number 8.
10. Pick up the last cube from the left and place it onto the image of result of 1 minus 6.
11. Pick up the blue cube and place it onto the image of number 28.
12. Pick up the cube on the far right and place it onto the result of 2^3.

Test



1. Pick up the grey cube and place it onto the image of number -10.
2. Pick up the cube on the left side and place it onto the results of 10 x 10.
3. Pick up the grey cube and place it onto the image of number 36.
4. Pick up the cube at position two from the left and place it onto the image of a number smaller than 0.
5. Pick up the orange cube and place it onto the image of number 100.
6. Pick up the cube sandwiched between grey and purple and place it onto the image of resul of 4 multiplies 9.
7. Pick up the purple cube and place it onto the image of number -10.
8. Pick up the cube directly to the left of cyan and place it onto the results of 10 x 10.
9. Pick up the purple cube and place it onto the image of number 36.
10. Pick up the cube directly to the right of purple and place it onto the image of a number smaller than 0.
11. Pick up the cyan cube and place it onto the image of number 100.
12. Pick up the cube to the right of the purple cube and place it onto the image of resul of 4 multiplies 9.

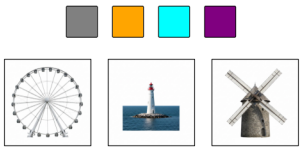
Category-21-building

Training



1. Pick up the leftmost cube and place it onto the image of a bridge.
2. Pick up the yellow cube and place it onto the place where people live.
3. Pick up the first cube from the left and place it onto the image of a skyscraper.
4. Pick up the green cube and place it onto the structure built to span a physical obstacle.
5. Pick up the second cube from the left and place it onto the image of a residential house.
6. Pick up the green cube and place it onto the very tall building in a city.
7. Pick up the cube to the left of the red cube and place it onto the image of a bridge.
8. Pick up the blue cube and place it onto the place where people live.
9. Pick up the second cube from the right and place it onto the image of a skyscraper.
10. Pick up the red cube and place it onto the structure built to span a physical obstacle.
11. Pick up the cube on the far right and place it onto the image of a residential house.
12. Pick up the red cube and place it onto the very tall building in a city.

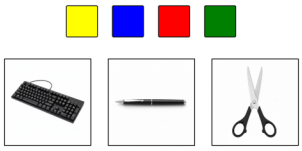
Test



1. Pick up the cube on the left side and place it onto the image of a Ferris wheel.
2. Pick up the grey cube and place it onto the tower with a bright light to guide ships.
3. Pick up the cube at position one from the left and place it onto the image of a windmill.
4. Pick up the orange cube and place it onto the image of a rotating big wheel.
5. Pick up the cube sandwiched between grey and cyan and place it onto the image of a lighthouse.
6. Pick up the orange cube and place it onto the building with sails that turn in the wind.
7. Pick up the cube directly to the left of purple and place it onto the image of a Ferris wheel.
8. Pick up the cyan cube and place it onto the tower with a bright light to guide ships.
9. Pick up the cube on the left side of purple and place it onto the image of a windmill.
10. Pick up the purple cube and place it onto the image of a rotating big wheel.
11. Pick up the cube to the right of the cyan cube and place it onto the image of a lighthouse.
12. Pick up the purple cube and place it onto the building with sails that turn in the wind.

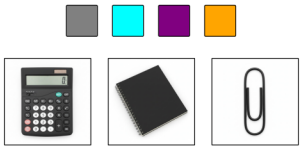
Category-22-office

Training



1. Pick up the yellow cube and place it onto the image of a keyboard.
2. Pick up the first cube from the left and place it onto the image of a tool for writing with ink.
3. Pick up the yellow cube and place it onto the image of a pair of scissors.
4. Pick up the second cube from the left and place it onto the image of a tool used for computer input.
5. Pick up the blue cube and place it onto the image of a pen.
6. Pick up the cube to the right of the yellow cube and place it onto the image of a tool for cutting paper.
7. Pick up the red cube and place it onto the image of a keyboard.
8. Pick up the second cube from the right and place it onto the image of a tool for writing with ink.
9. Pick up the red cube and place it onto the image of a pair of scissors.
10. Pick up the cube on the far right and place it onto the image of a tool used for computer input.
11. Pick up the green cube and place it onto the image of a pen.
12. Pick up the rightmost cube and place it onto the image of a tool for cutting paper.

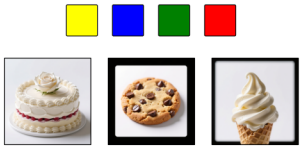
Test



1. Pick up the grey cube and place it onto the image of a calculator.
2. Pick up the cube at position one from the left and place it onto the image of a tool for writing down things.
3. Pick up the grey cube and place it onto the image of a paperclip.
4. Pick up the cube sandwiched between grey and purple and place it onto the image of a device for math operations.
5. Pick up the cyan cube and place it onto the image of a notebook.
6. Pick up the cube immediately next to the grey one and place it onto the image of a wire for holding papers together.
7. Pick up the purple cube and place it onto the image of a calculator.
8. Pick up the cube on the left side of orange and place it onto the image of a tool for writing down things.
9. Pick up the purple cube and place it onto the image of a paperclip.
10. Pick up the cube to the right of the purple cube and place it onto the image of a device for math operations.
11. Pick up the orange cube and place it onto the image of a notebook.
12. Pick up the cube on the most right side and place it onto the image of a wire for holding papers together.

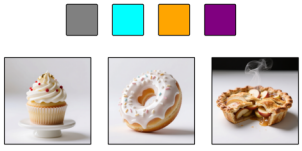
Category-23-food_extra

Training



1. Pick up the first cube from the left and place it onto the image of a cake.
2. Pick up the yellow cube and place it onto the small flat baked sweet.
3. Pick up the cube on the far left and place it onto the image of ice cream.
4. Pick up the blue cube and place it onto the birthday dessert.
5. Pick up the cube to the right of the yellow cube and place it onto the image of a cookie.
6. Pick up the blue cube and place it onto the frozen dairy dessert.
7. Pick up the second cube from the right and place it onto the image of a cake.
8. Pick up the green cube and place it onto the small flat baked sweet.
9. Pick up the cube between the blue and red cubes and place it onto the image of ice cream.
10. Pick up the red cube and place it onto the birthday dessert.
11. Pick up the rightmost cube and place it onto the image of a cookie.
12. Pick up the red cube and place it onto the frozen dairy dessert.

Test



1. Pick up the cube at position one from the left and place it onto the image of a cupcake.
2. Pick up the grey cube and place it onto the ring-shaped fried dough.
3. Pick up the cube directly to the left of cyan and place it onto the image of a pie.
4. Pick up the cyan cube and place it onto the small cake for one person.
5. Pick up the cube immediately next to the grey one and place it onto the image of a donut.
6. Pick up the cyan cube and place it onto the fruit filled pastry.
7. Pick up the cube on the left side of purple and place it onto the image of a cupcake.
8. Pick up the orange cube and place it onto the ring-shaped fried dough.
9. Pick up the cube sandwiched between cyan and purple and place it onto the image of a pie.
10. Pick up the purple cube and place it onto the small cake for one person.
11. Pick up the cube on the most right side and place it onto the image of a donut.
12. Pick up the purple cube and place it onto the fruit filled pastry.