Proceedings of the 4th International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/2026.36903

Vision-Centric World Models for Embodied Robots: Representations, Predictive Interfaces, and Evaluation

Yining Li

College of Computer Science and Technology, Zhejiang University, Hangzhou, China
3230105343@zju.edu.cn

Abstract. Embodied robots need more than a description of the current image: they must estimate how the scene may change under motion, contact, and partial observation. Vision-centric world models provide this predictive layer, but they expose it through different state interfaces. This paper organizes the literature into four families according to the state available to planning: future observations, compact latent states, geometry-structured maps, and persistent entities or relations. The comparison examines how each state is formed, advanced, and queried; where inference and planning costs arise; and which errors matter in physical use. Representative methods show that observation prediction retains interpretable appearance but makes repeated rollout expensive. Latent dynamics reduce that cost while risking the loss of contact-scale variables. Geometric states support pose, clearance, and occupancy queries, although their reliability depends on calibration and timely updates. Entity-relational states preserve object identity and task relations, yet they remain vulnerable to binding errors under occlusion. Evaluation is therefore linked to the exposed state rather than to a single generic score. The resulting framework clarifies where temporal prediction, persistent geometry, and semantic identity complement one another in embodied planning.

Keywords: vision-centric world models, embodied robots, predictive representations, robot planning

1. Introduction

With the rapid development of embodied intelligence, robots are expected to perform complex tasks in dynamic and unstructured environments. Embodied robots must continuously perceive their surroundings, infer latent environmental states, and make decisions through physical interaction. Their environment representations must remain useful as the robot moves, acts, and encounters changes in its surroundings.

Computer vision provides the basic perceptual interface for embodied robots. Object detection, semantic segmentation, depth estimation, and scene understanding can describe what is visible at a particular instant, and modern foundation models have improved their accuracy and transferability. A snapshot-level description, however, does not specify how an object will move, which regions are occluded, or what will happen after a robot action. Manipulation, navigation, and human-robot collaboration require planning over state changes and action outcomes across time.

DOI: 10.54254/2755-2721/2026.36903

Vision world models address this gap by learning an internal, predictive representation of visual environments. World Models established a compact visual-dynamics-and-controller pipeline, PlaNet brought latent dynamics into planning from pixels, and Dreamer-style imagination showed that learned dynamics can support policy improvement while reducing direct interaction with the physical world [1-3]. For embodied robots, prediction alone is insufficient. The retained state must preserve the geometry, object identity, uncertainty, and action-dependent changes needed by the downstream task.

The literature spans several research communities. Video prediction models generate future visual observations. Model-based reinforcement learning advances compact latent states under candidate actions. Neural mapping represents geometry and view-dependent appearance, while object-centric learning factors a scene into persistent entities and relations. Recent work on embodied simulators and world models emphasizes scale, pretraining data, and task transfer [4]. Its temporal or spatial state is often embedded in an end-to-end architecture. Consequently, similar systems may be described as dynamic, spatial, semantic, generative, or action conditioned even when the planner reads the same type of state. Recent surveys examine world models for robot learning, embodied simulators, and links to predictive coding in cognitive and developmental robotics [4-6]. This paper narrows that broad literature to the state interface exposed to planning.

This paper uses a targeted narrative synthesis rather than a systematic or bibliometric review. Sources were identified from recent surveys and reference chaining, then screened for an image-grounded predictive or queryable state and evidence relevant to planning, control, mapping, or evaluation. The selection balances foundational work, recent representative systems, the four planner-facing state carriers, and evaluation standards. Purely reactive policies, static reconstruction without a predictive or query interface, duplicate technical reports, and works lacking sufficient methodological or evaluation detail for comparison were excluded. The resulting corpus supports a technical synthesis, not claims about publication prevalence or exhaustive coverage.

This review uses the planner-facing state as the organizing variable. A vision-centric world model takes images or image-derived features as input, maintains a state grounded in those observations, and supports a query about an unseen time, viewpoint, or candidate action before the outcome is observed. The state must be available to a downstream controller, planner, evaluator, mapper, or policy. A predictive branch used only as a training loss is not treated as a planning interface. These conditions keep direct reactive policies and static scene assets adjacent to the review without conflating them with queryable world models.

Four categories follow from the index used to retrieve state. Observation-space models return future pixels or video tokens. Latent-state models propagate compact features whose addresses do not correspond to stable entities or metric coordinates. Geometry-structured models answer queries at positions, poses, voxels, surfaces, or geometric primitives. Entity-relational models retrieve attributes and relations through persistent object identities or slots. A hybrid system may contain several representations; its category here follows the state actually consumed in the reported downstream task. This distinction is useful for comparison because representation, transition, and planning cost are determined by the exposed state, not by the name of the architecture.

The query address also determines what an error means. In observation space, an error appears as a misplaced pixel, object, or motion pattern. In latent space, it appears as a state that misranks candidate actions. A geometric error changes pose, distance, occupancy, or visibility. An entity error changes identity, attribute, or relation. These failure units lead to different training targets and evaluation measures, even when two models are tested on the same robot task.

Figure 1 summarizes the information flow.

DOI: 10.54254/2755-2721/2026.36903

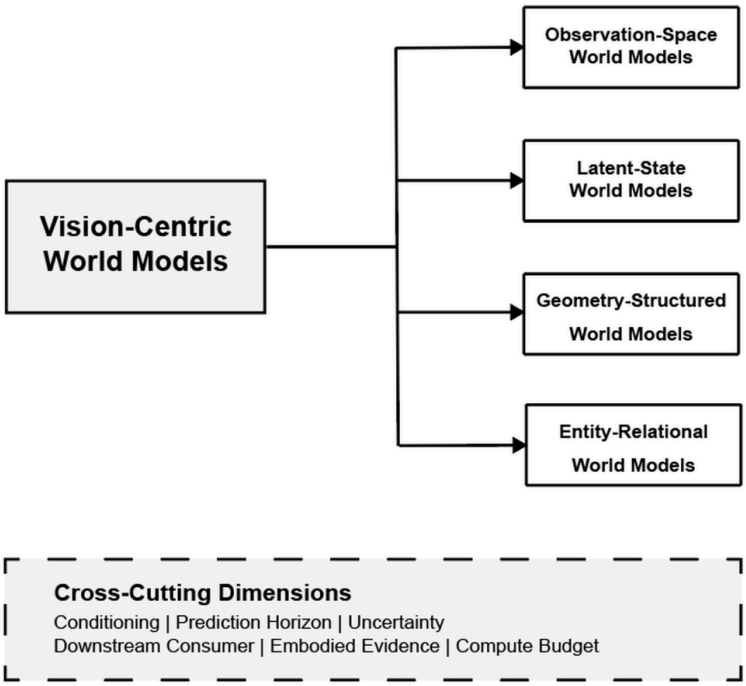

Figure 1. Planner-facing classification of vision-centric world models (picture credit: original)

Sections 2-5 discuss the four categories, followed by a cross-category comparison of representative methods and their evaluation practices.

2. Observation-space world models

Observation-space models expose predicted images or video tokens. Given an observation history and a candidate action sequence, the model estimates a distribution over the next H observations:

$$p(o_{t+1:t+H} \mid o_{1:t},\ a_{t:t+H-1}) \tag{1}$$

where o_t denotes the observation at time t, a_t denotes the action at time t, t is the current time index, and H is the prediction horizon. The planner can inspect the rollout directly or pass it to an existing visual scorer. The same choice retains appearance and produces a high-dimensional search space. Figure 2 shows the basic interface.

This interface changes the role of perception in planning. A conventional perception stack extracts a depth map, object pose, or task feature before control begins. Observation prediction instead keeps the visual carrier and postpones that reduction until a rollout is scored. The arrangement is useful when a pretrained detector, a goal image, or a human-readable cost can already evaluate future frames. It is less convenient when the controller needs a precise contact state or must compare hundreds of action sequences within one control cycle. In that setting, the image generator and the scorer repeat substantial visual computation for every candidate.

DOI: 10.54254/2755-2721/2026.36903

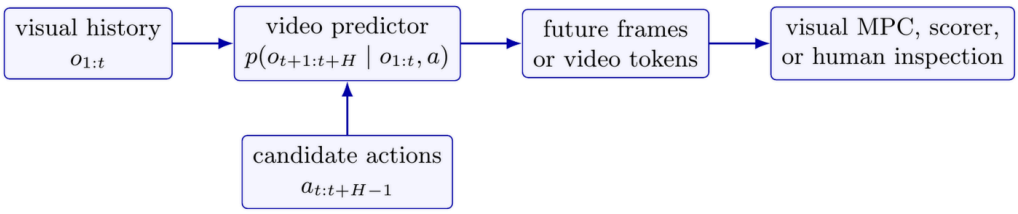

Figure 2. Observation-space rollout and visual planning interface (picture credit: original)

2.1. Direct video prediction

Deep Visual Foresight established that action-conditioned video prediction could be queried inside visual model-predictive control (MPC) [7]. An incorrect object trajectory or missing contact is visible in the rollout, which makes the prediction easier to diagnose than a hidden task state. The same interface is demanding: training data must cover the robot, camera, objects, and contacts that the predictor will encounter, while temporal drift grows when its own output becomes the next input. Short-horizon image evidence is therefore the main planning signal, and new observations are used to correct the rollout before errors dominate.

In visual MPC, the model is normally queried several times rather than trusted for an entire task. A planner samples candidate actions, rolls each sequence forward, compares the predicted frames with a visual goal or object cost, executes a short prefix, and observes the scene again. Replanning limits the time during which model error can accumulate. It also makes camera calibration and action timing part of the predictive interface. If the commanded displacement is expressed in a robot frame while most apparent motion comes from the camera, a model can fit the training video without learning the object response required by control. Multi-camera observations or explicit motion cues help separate these effects, although they raise the input and synchronization cost.

Autoregressive prediction accumulates pose, texture, and identity errors. Efficient transformer predictors make the choice explicit between fully autoregressive and non-autoregressive block prediction [8]. Unified World Models couples video and action diffusion so that generated trajectories and control remain connected [9]. The broader design choice is between a sequential predictor that reuses its previous frame and a space-time generator that revises a complete block. Either design must still align apparent motion with executable robot action.

The prediction mechanism affects what can be varied at test time. Autoregressive models provide a direct temporal factorization and can reuse previously generated frames, but an early displacement becomes part of the next input. Diffusion models revise a space-time block through denoising and can retain several plausible motion patterns. Their additional samples are valuable only when the planner treats them as alternatives rather than averaging them into one visually smooth future. Simulator-style models expose camera and actor interventions more explicitly. They demand stronger scene consistency because a query may change viewpoint, ego motion, and object motion at once. Across these designs, temporal resolution has a practical effect: coarse steps reduce rollout length but can skip the brief contact that changes an object's subsequent motion.

2.2. Learned control interfaces

Internet video offers broad visual variation but rarely includes synchronized actions, force, or contact. Robot trajectories provide those signals with narrower scene coverage. PEWM restricts generation to short primitive-level horizons and links the resulting visual changes to a modular planner [10]. GEM-4D adds geometric correspondence before a generated rollout is converted into

DOI: 10.54254/2755-2721/2026.36903

robot motion [11]. These methods reduce dependence on dense action labels, yet their learned controls still require alignment with camera motion and robot kinematics.

Alignment can be carried out at several levels. A latent action may describe a visible change without specifying which joint command caused it. It can support interactive generation, but a physical robot still needs an inverse map from that change to an executable command. Geometric correspondence provides one route by recovering motion structure before control. This extra interface creates a failure that image metrics cannot reveal: a rollout may look coherent while representing a displacement, velocity, or contact mode that the robot cannot execute.

Pixel similarity and Fréchet Video Distance describe appearance [12]. Robot use also depends on object persistence, depth consistency, action response, and error growth over time. Object tracks and segmentation reveal identity failures that a perceptual score may miss. Multiple sampled futures can express ambiguity, but the sample count changes both coverage and latency. Observation-space models therefore suit short, inspectable rollouts; long-horizon control usually benefits from a more compact or structured state.

Evaluation should consequently pair visual measures with a task-facing test. Frame quality indicates whether the generator preserves texture and broad motion, while object trajectories show whether the commanded action moves the intended instance. Depth or correspondence checks reveal geometric drift, and horizon curves show when repeated prediction stops being useful. Closed-loop success then includes the effect of replanning and the visual cost. Reporting rollout time beside these measures matters for embodied use: a high-fidelity future that arrives after the control deadline cannot serve the same role as a lower-resolution prediction updated at every step.

3. Latent-state world models

Latent-state models encode visual history into a compact state and learn how actions advance that state. A decoder may reconstruct observations, but planning consumes the propagated latent, a reward prediction, or a learned value. Figure 3 illustrates this separation between perception and imagination.

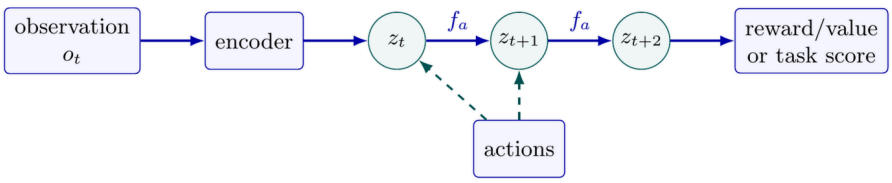

Figure 3. Latent imagination and task scoring (picture credit: original)

The latent is usually formed from the current image together with a recurrent summary of earlier observations. This history matters in partially observed scenes. Two frames can look alike even though an object has different velocity, a drawer was previously opened, or a tool is hidden behind the robot hand. During training, the encoder can use the previous state, previous action, and current observation to form an observation-corrected posterior. During imagined planning, the future observation is unavailable, so the model predicts a prior from the current state and action and feeds that prior into the next transition. A one-step mismatch between the observation-corrected posterior and the self-propagated prior is therefore reused at every imagined step. Over a long rollout, missing velocity, contact, or occluded-object information can accumulate into a state that ranks actions incorrectly. This posterior-prior gap is a core long-horizon problem for latent models whose training states are corrected by observations but whose planning states are propagated without them.

DOI: 10.54254/2755-2721/2026.36903

3.1. Recurrent state-space models and imagination

The development of latent world models can be read in three stages: compact visual dynamics, online latent planning, and imagined policy learning. World Models couples visual processing, recurrent memory, and control in a compressed state [1]. PlaNet brings learned latent dynamics directly into online planning from pixels [2]. The current image is encoded, candidate actions advance the latent, predicted returns rank the candidates, and the agent executes a short prefix before observing again. This arrangement uses fresh observations to correct the state, but it pays the search cost at every decision.

Dreamer moved repeated search from execution time to training time. Its flow is data collection, posterior state inference, imagined latent trajectories, actor-critic updates, and then direct policy execution [3]. The resulting actor is cheaper to run than online search, although it inherits errors from model-generated training trajectories. A temporally hierarchical deep-active-inference controller applies slow and fast latent states to real-world robot control, where interaction time and control frequency make that trade-off visible [13]. As latent objectives become more task directed, dependence on a pixel decoder can decrease. Diagnostic evidence decreases with it: a failed plan may be visible only as a misranked value or action. Recent work shifts the question from how imagination is used to what the latent should retain. TD-MPC2 uses a decoder-free latent model and local trajectory optimization across continuous-control domains [14]. Hierarchical Planning with Latent World Models uses multiple temporal scales and macro-actions to reduce long-horizon search cost [15]. V-JEPA 2-AC adds action conditioning to a predictive feature space pretrained from large-scale video [16]. The practical choice is which task variables, transferable cues, and uncertainties must survive compression.

Task-oriented objectives and pretrained feature objectives make different compromises. A task-oriented latent may discard texture that does not affect value, which is helpful when appearance changes but dynamics do not. TD-MPC2 emphasizes decision-relevant prediction, hierarchical latent planning preserves variables across temporal scales, and V-JEPA 2-AC retains broad visual cues that can transfer across observations [14-16]. None guarantees contact-state fidelity: two states that are semantically similar can still require different forces or approach directions. Cross-embodiment training can broaden the action and scene distribution, while simulation can vary appearance and physical parameters. Neither source removes embodiment differences from the transition model.

Shared datasets do not remove embodiment differences. Camera placement, control rate, gripper geometry, and action parameterization alter the transition that a common latent must learn. Conditioning on embodiment information can separate these modes, while a normalized action space makes transfer easier to implement. Domain randomization covers selected changes in rendering and physics, but the latent should still expose uncertainty when a test scene falls outside those variations. Otherwise, a compact state can produce confident imagined rewards from observations that the encoder has not represented reliably.

3.2. Compression, aliasing, and uncertainty

Compact rollouts reduce planning cost. Compression can also map visually distinct situations to a similar latent when the training objective assigns them the same target. Thin obstacles, contact state, or a partly occluded tool may disappear from the representation. Depth, optical flow, segmentation, and reward objectives can preserve selected variables, while object slots add addressable components between a global latent and full images.

DOI: 10.54254/2755-2721/2026.36903

Aliasing is especially damaging when the omitted variable changes the ordering of actions. A small obstacle may have little effect on reconstruction loss but can invalidate an otherwise high-value trajectory. Auxiliary prediction heads encourage the state to retain selected geometry or motion, yet each head reflects a prior choice about what will matter. Another option is to keep a richer latent during planning and compress only the part used by the value model. This raises computation, but it gives the controller access to features that were not anticipated by a single reward target.

Stochastic states, ensembles, and transition disagreement represent alternative futures. Their probabilities need to correspond to robot events such as collision or failed grasp if a controller uses them for risk-sensitive action selection. Long-horizon tests are also important because a low one-step error can grow after repeated latent transitions. Compared with observation-space prediction, latent imagination offers faster search and weaker direct interpretability. The trade-off depends on the control horizon and on the smallest task-relevant structure that the encoder must retain.

One-step latent loss, reconstruction quality, and control return answer different questions. The first measures local consistency in the learned space; the second checks whether visual information can be decoded; the third tests whether the retained variables are sufficient for decisions. Horizon-wise reward or feature error exposes drift that an aggregate return can hide. Planning time and the number of imagined transitions indicate how much search the compact state actually enables. Latent models are most attractive when many candidates must be compared quickly and the training objectives cover the variables that distinguish safe actions from visually similar failures.

4. Geometry-structured world models

Geometry-structured models index state by coordinates, poses, voxels, surfaces, or geometric primitives. Their inputs may be posed red-green-blue (RGB) images or red-green-blue plus depth (RGB-D) observations. They answer where a surface lies, whether space is occupied, or what should be visible from a query viewpoint. Figure 4 shows the retained map and its two common query paths.

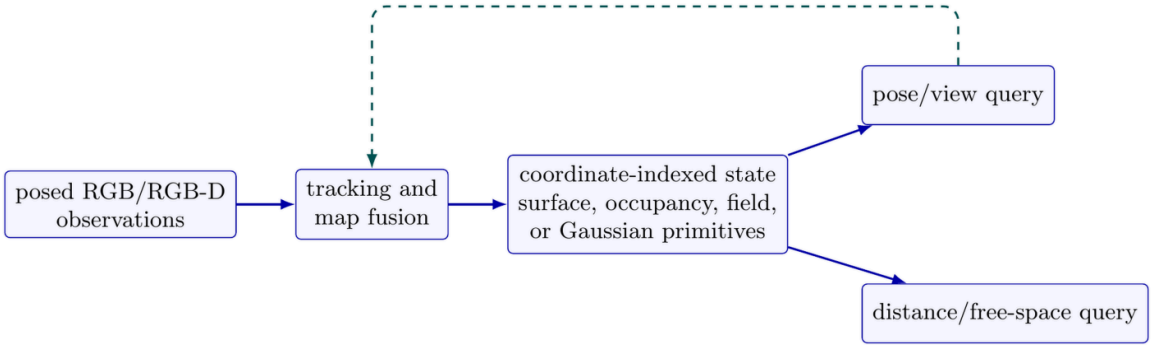

Figure 4. Geometry-structured mapping and query interface (picture credit: original)

Unlike a global latent, a geometric state has an explicit address system. A query at a camera pose or three-dimensional coordinate should refer to the same part of the environment after the robot moves. This property supports viewpoint selection, collision checking, and spatial memory, but it also makes coordinate errors persistent. Every map update depends on calibration and pose estimation. If a moving object is fused as part of the static scene, later observations may leave

DOI: 10.54254/2755-2721/2026.36903

duplicate surfaces or erase free space. Geometry-structured world models therefore combine representation learning with a data-association and reference-frame problem.

4.1. Neural fields, Gaussians, and online mapping

Neural radiance fields represent color and density as a continuous function of position and viewing direction [17]. Novel-view rendering supports viewpoint planning and occlusion reasoning, but the original optimization is slow and assumes a mainly static scene. Dynamic extensions can explain change through deformation or moving primitives. Their spatial continuity is useful, although rendering quality alone does not establish collision accuracy.

Canonical deformation models explain change by mapping observations at different times to a shared reference. They work well when the deformation is smooth and correspondences remain recoverable. Topology changes, newly revealed surfaces, and independently moving objects are harder to express with one field. Explicit dynamic Gaussians offer local primitives that can move, split, or be updated, which makes the representation easier to inspect. Their geometry still depends on how opacity and scale are converted into a surface. A planner that only needs a rendered view can tolerate a different error profile from one that requires a conservative clearance estimate.

3-D Gaussian Splatting replaces volumetric sampling with explicit anisotropic primitives and fast rasterization [18]. Its rendering speed supports interactive updates, while geometric use still depends on surface regularity and view coverage. The difference between rendering and planning queries is important here. View synthesis integrates color and opacity along rays, so several imperfect primitives may still produce a convincing image. Collision checking asks whether a particular volume is free, and a small unsupported gap can change the answer. Depth supervision, surface regularization, or conversion to occupancy can make Gaussian maps more useful for control. These operations add assumptions about thickness and free space that are not measured by appearance scores alone.

Online neural simultaneous localization and mapping (SLAM) connects learned scene representations to robot state estimation. The same feedback structure appears across neural mapping systems: the current map supports pose estimation, and the estimated pose determines where new observations enter the map. Pose error and map error are therefore coupled. A sharp novel view can coexist with incorrect metric scale or an outdated surface.

Tracking and mapping form a feedback loop. The current map predicts what the camera should observe, and the discrepancy is used to update pose. The estimated pose then determines where new pixels or depth samples enter the map. Local tracking failure can thus contaminate later updates even after the image becomes clear again. Keyframes, multiscale features, and joint optimization improve recovery, but they consume time that competes with the robot control loop. For planning, map-update latency is as relevant as final reconstruction quality because the robot acts while the estimate is still changing.

Signed-distance and occupancy representations make geometry easier to query. iSDF updates a continuous signed-distance field from posed depth and returns collision-relevant distances [19]. Fine voxels improve local detail but increase memory. Thin structures and unobserved free space remain difficult. Map timestamps, confidence, and separate treatment of static surfaces and moving objects help a planner avoid stale geometry.

Signed distance gives a local notion of clearance, while occupancy represents whether a cell is free, occupied, or unknown. The distinction matters around unseen regions. Treating an unobserved cell as free encourages shorter but unsafe paths; treating it as occupied encourages exploration or conservative motion. Continuous fields can query arbitrary coordinates and provide gradients for

DOI: 10.54254/2755-2721/2026.36903

trajectory optimization. Voxel grids provide direct neighborhood operations and predictable memory access. Resolution then sets the smallest obstacle and passage that the planner can represent.

4.2. Future occupancy and semantic maps

OccSim autoregressively generates long-horizon occupancy maps from an initial state and future ego-actions [20]. The internal model may use discrete tokens, yet the planning interface is a coordinate-indexed future occupancy volume. This interface supports direct free-space and collision queries. Its accuracy is bounded by voxel resolution, rare-class imbalance, and motion uncertainty.

Separating ego motion from scene flow reduces an otherwise ambiguous source of change. A static wall moves across the sensor frame when the robot turns, whereas a person can move within the world frame. Forecasting both with one undifferentiated update makes it harder to maintain map consistency. Indoor robot scenes add close-range occlusion, narrow passages, and movable objects, so occupancy errors near the robot can matter more than errors over a large visible volume. Horizon-specific intersection-over-union and collision checks provide different views of this behavior.

Language features can also be attached to coordinates. Neural-field systems can attach geometry, semantics, and dynamics to coordinate-indexed representations [21]. Such maps aid navigation and object search, but spatial fusion may merge similar instances or leave semantic features at a previous position after an object moves. Coordinate-indexed semantics remain geometry structured because the query address is a location rather than a persistent object identity.

Semantic maps give planning a vocabulary beyond surface shape. A command can be grounded to regions associated with a category or language description, after which metric planning supplies a route or viewpoint. The fusion process averages evidence across frames, which stabilizes noisy features but may blur adjacent instances. Confidence, observation time, and instance-aware updates reduce stale labels. When the task asks for "the mug seen earlier" rather than any mug-shaped region, however, a location-indexed feature is insufficient by itself. That query motivates the persistent identities discussed in the next section.

5. Entity-relational world models

Entity-relational models organize state by persistent objects, slots, or graph nodes. Each entity carries attributes, and edges encode interactions or task relations. This factorization supports identity-sensitive queries under viewpoint change and occlusion. Figure 5 shows how visual evidence updates a small relational memory.

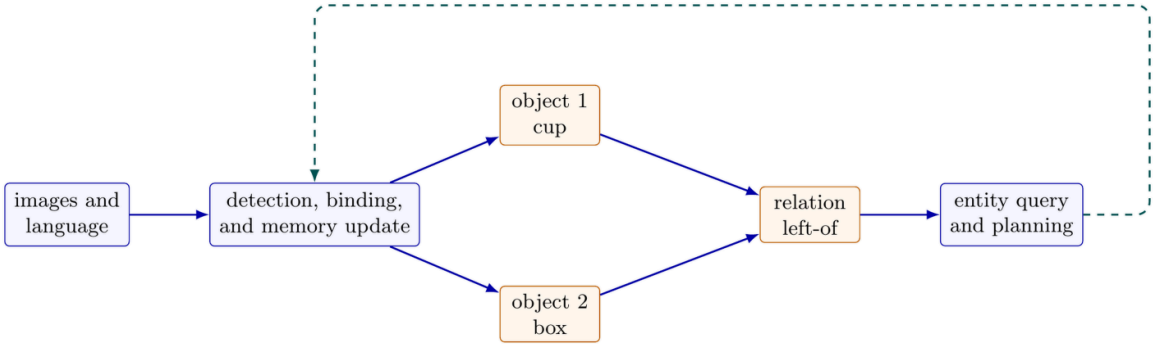

Figure 5. Entity binding, relational memory, and planning interface (picture credit: original)

DOI: 10.54254/2755-2721/2026.36903

The state of an entity may contain appearance, position, motion, category, uncertainty, or an action-relevant attribute such as whether a container is open. Relations can describe distance, support, containment, contact, or a learned interaction. Because the planner addresses a node rather than a pixel or coordinate, evidence observed from different views can update one record. The representation is compact when only a few objects matter. Its usefulness depends on maintaining that address through missed detections, close interactions, and changes in the number of visible objects.

5.1. Object-centric dynamics

Object-centric dynamics developed from modular scene representations through learned slot rollouts and then task-conditioned embodied systems. ROOTS learns modular object-centric three-dimensional representations from partial observations [22]. FOCUS learns an object-centric world model and uses its structured state to guide exploration in robot manipulation [23]. These works make the persistent unit of prediction explicit rather than treating the scene as one undifferentiated feature vector.

The next stage places more responsibility on the perceptual interface. SlotFormer rolls learned object slots forward and tests whether they support long-horizon video reasoning and planning [24]. WorldDP couples an object-centric high-level world model to a low-level diffusion policy for multi-stage manipulation [25]. The progression moves from stable visual decomposition to entity states exposed directly to embodied planning and execution.

Action conditioning has several meanings in an object state. Appending a global command to every slot is useful when the action changes the camera or the whole scene, but it can entangle unaffected objects with the command. Assigning the action only to a selected slot preserves locality when perception already identifies a manipulation target; an identity error then sends the action to the wrong state. A third option represents the command as an interaction between the end effector and scene objects. That form suits contact-rich or multi-object effects, provided that end-effector pose, object binding, and relation updates remain reliable. Language can specify intent at a coarse level, but object dynamics still need visual or action evidence to determine the resulting motion.

Object factorization makes contact and relation queries explicit. It also introduces a binding problem. Occlusion, non-rigid motion, and an incorrect number of slots can switch identities or merge instances. A predicted object may remain visually plausible while its identity no longer matches the physical object that the planner intends to manipulate. Identity-switch rate, object-state error, relation accuracy, persistence through occlusion, and task success target these failures more directly than whole-frame reconstruction.

Birth, disappearance, and reappearance make binding harder than frame-wise segmentation. A newly visible object may require a new node, whereas a temporarily occluded object should retain its previous address. Similar instances create another ambiguity: appearance alone may not distinguish two identical blocks after they cross paths. Motion continuity and geometric position provide evidence, but both can be uncertain during contact. A relational memory should therefore carry confidence or alternative associations instead of committing every observation to one identity. For example, after two identical blocks cross under occlusion, the memory can retain two weighted assignments between the old identities and the new detections. If the same action is safe under both assignments, the planner can act immediately. If one assignment makes the target reachable and the other makes it a distractor, the planner should first move the camera to obtain a disambiguating view.

DOI: 10.54254/2755-2721/2026.36903

5.2. Scene graphs and the boundary with policies

ConceptGraphs builds an open-vocabulary three-dimensional scene graph whose nodes and relations are available to robot perception and planning [26]. Perceptual slots remain outside the present definition when they are evaluated only for decomposition or tracking and are not rolled out or queried for a decision. Direct vision-language-action policies sit on the other side of the same boundary: they may contain predictive features, yet an external planner cannot necessarily query them. WorldVLA links action generation with an autoregressive world state [27]. It enters the entity category only when planning retrieves persistent entity or relation indices from that state.

A scene graph joins identity with explicit spatial and semantic relations. Language-enabled scene graphs place object nodes in three-dimensional space, enabling queries such as which object is on a support surface or near a target region. Graph edges can be updated more cheaply than regenerating a complete observation, but they depend on the underlying object poses and associations. Relations can also become stale at different rates. Category and appearance may remain stable, while contact, containment, and reachability change after one action.

The boundary with policies is determined by access, not by network size. A vision-language-action model may contain internal features that encode objects and future effects. If these features are consumed only inside a direct action predictor, an external planner cannot query an alternative object's state or compare candidate futures. WorldVLA moves toward an explicit world-state sequence, but classification still depends on whether persistent state is available to the decision process. This criterion prevents ordinary hidden activations from being counted as entity world models merely because they contain semantic information.

Entity-relational models fit tasks in which object identity and compositional change dominate, including rearrangement, tool use, and instruction following. They are less reliable when the scene has no stable object decomposition or when non-rigid material must be represented by a few slots. Hybrid designs can attach an entity to a geometric pose and use an observation decoder for inspection. The interfaces then need a shared reference: the node called by the planner, the region in the map, and the object shown in a predicted frame must denote the same physical instance. Evaluation should check that consistency in addition to slot reconstruction or final task success.

6. Comparative analysis and evaluation

The four categories expose different planning variables. Table 1 compares their interfaces, computational characteristics, and common errors. An architecture may combine several rows, for example by learning dynamics over object poses stored in a persistent three-dimensional map.

Table 1. Planner-facing state carriers

Category	Exposed state	Typical transition or update	Main advantage	Characteristic limitation
Observation space	Future pixels or video tokens	Sequential, diffusion, or learned-video update	Inspectable appearance; compatible with visual scorers	High rollout cost; visually subtle physical or identity errors
Latent state	Compact feature, reward, or task state	Recurrent, stochastic, token, or feature transition	Fast imagination and action search	Aliasing can remove small obstacles or contact variables

DOI: 10.54254/2755-2721/2026.36903

Table 1. (continued)

Geometry structured	Surface, occupancy, pose, field, or primitives	Tracking, map fusion, deformation, or occupancy flow	Metric view, free-space, and collision queries	Pose drift, stale geometry, and resolution-dependent error
Entity relational	Persistent object states and relations	Binding, object transition, and graph-memory update	Identity-sensitive reasoning and compositional queries	Identity switches, slot mismatch, and outdated relations

Table 2 summarizes within-study evidence reported for representative methods. Each comparison is valid only within the cited study's own benchmark and baseline setup; results from different rows do not form a common leaderboard because sensors, controllers, data, horizons, and success criteria differ.

Table 2. Reported performance evidence for representative methods

Carrier	Method	Evaluation setting	Reported result	Interpretation and limit
Observation	PEWM [10]	Real robot; task-specific OpenVLA fine-tuned on 100 demonstrations per task	Task success (20 trials): 16/20 (80%), 14/20 (70%), and 13/20 (65%); fine-tuned OpenVLA: 12/20 (60%), 10/20 (50%), and 4/20 (20%)	Cup pick, cloth move, and cloth fold, respectively; OpenVLA zero-shot was 0/20 on all three tasks
Observation	GEM-4D [11]	DROID real-world tasks; success judged by a human study	Paper-reported aggregate success: 81% versus 61% for TesserACT (+20 percentage points)	Video rollouts are converted to 6-DoF trajectories; this result is separate from the reported 63-82% RLBench range
Latent	TD-MPC2 [14]	104 online RL tasks in four domains; a separate 80-task multitask suite	One hyperparameter set across 104 tasks; one 317M-parameter agent trained to perform 80 tasks	The 104-task single-task benchmark and the 80-task multitask experiment are distinct evaluations
Latent	HWM [15]	Franka (real); Push-T and Diverse Maze (simulation)	Franka cup pick-and-place: 70% versus 0% for flat planning; reported compute reductions are 3x on Push-T and 4x in one Diverse Maze setting	The real-robot success and simulated compute claims come from separate task suites
Geometry	OccSim [20]	OccSim-generated data for zero-shot 4D semantic occupancy forecasting	Up to 67% of real-data upper-bound performance; about 74% with 5x data; over 3,000 frames and 4 km per rollout	A relative downstream forecasting result, not raw IoU or manipulation success

DOI: 10.54254/2755-2721/2026.36903

Table 2. (continued)

Entity	WorldDP [25]	OGBench-derived manipulation tasks; 50 held-out trajectories per task	Cube-Triple all-three-cube success: 30% versus 12% for the next-best baseline; Scene-Single-Direct average: 74%	All baselines use the same held-out trajectories; success is computed from ground-truth environment states
Entity	ConceptGraphs [26]	CG variant on seven Replica scenes; Amazon Mechanical Turk human evaluation	Mean node precision: 0.71; mean edge precision: 0.88	Measures scene-graph construction quality, not predictive control; CG-D reports different values

The performance evidence shows both progress and fragmentation. Observation-space studies can report task success after converting video to action; latent studies can isolate planning scale or hierarchy; geometry studies mix manipulation success with map or simulator utility; and entity studies measure both structured-state accuracy and multi-stage control. Evaluation should therefore first ask whether the predicted carrier preserves task-relevant variables, then test whether those variables support correct rollout ranking or queries, and finally verify closed-loop behavior. Table 3 aligns carrier-specific tasks, measures, and interpretations under this logic.

Table 3. Evaluation practices by state carrier

Category	Tasks or benchmarks	Measures	Interpretation
Observation space	Robot pushing, interactive video, driving simulation	Learned Perceptual Image Patch Similarity (LPIPS), Fréchet Video Distance (FVD), trajectory error, object persistence, rollout latency	Separates visual plausibility from action-response accuracy
Latent state	Control suites, visual goal planning, physical robot learning	Return, success, latent or feature error, horizon curves, planning time	Tests whether compression preserves decision variables
Geometry structured	RGB-D mapping, occupancy prediction, reconstruction, navigation	Absolute trajectory error (ATE), depth or occupancy error, completion, intersection over union (IoU), collision checks, update latency	Assesses whether map quality supports localization and free-space queries
Entity relational	Object dynamics, tracking through occlusion, language-guided manipulation	Identity-switch count, relation accuracy, slot prediction error, task completion	Reveals faults in maintaining identity and updating relations

The three evaluation levels are complementary rather than interchangeable. Representation-level metrics are inexpensive and diagnostic, but a better reconstruction or feature loss does not guarantee that the planner ranks actions correctly. Predictive-interface tests evaluate rollout ranking, map queries, or relation queries and therefore expose decision-relevant errors, although their metric is specific to the carrier and is difficult to compare across categories. Closed-loop task performance gives the most direct evidence of usefulness, but it also includes the controller, action budget, replanning frequency, and hardware conditions. A strong evaluation therefore uses the first level to locate errors, the second to test the world-model interface, and the third to verify the complete embodied loop.

DOI: 10.54254/2755-2721/2026.36903

Closed-loop results depend on the controller configuration and allowed action budget; changing either can alter the outcome even when the world model is fixed. CALVIN, for instance, records progress across sequences of language-conditioned manipulation tasks [28]. Simulation suites permit repeated trials and access to the underlying state. Tests on physical robots instead bring camera delay, calibration drift, reset procedures, and imperfect contact into the result. Compute should be reported in terms that match the representation. For video diffusion, denoising samples are often the main cost; for latent planners, it is usually the number of candidate rollouts; for maps, update rate and spatial resolution set the workload.

Probabilistic outputs need a separate check. Reliability diagrams compare predicted confidence with observed event frequency, while scalar calibration metrics summarize different aspects of the gap [29]. The relevant event in a robot experiment could be a collision, failed grasp, identity swap, or inconsistency in the map. Treating it as a task event, rather than a pixel label, keeps calibration tied to the decision made by the robot.

The representations solve different parts of the same planning problem. A predicted observation retains appearance for visual scorers and for inspection. Latent rollouts are cheaper to advance many times. A geometric map provides distances, occupancy, and reachability constraints. Entity memories keep track of which object is involved and how it relates to others. In a hybrid model, these components must share addresses and reference frames. If the map and object memory refer to different physical instances, adding a richer decoder or another state branch will not answer the planner's query. The same consistency requirement applies when an object state is rendered back into a predicted observation.

7. Conclusion

The planner-facing state provides a practical way to separate vision-centric world models. Observation-space models predict what the robot may see and preserve visual evidence that can be inspected or passed to an existing scorer. Latent models trade that visibility for compact rollouts and faster action search. Geometry-structured models retain positions, surfaces, occupancy, or clearance in a coordinate system, whereas entity-relational models preserve object identities and the relations needed for compositional tasks. These interfaces make work from video generation, model-based control, neural mapping, and object-centric learning comparable without forcing their results into a single metric.

Each carrier also locates error differently. A plausible video can hide an impossible contact. A compact latent may rank actions incorrectly after discarding a small obstacle. A map may render sharply while its scale or free-space estimate is wrong. An object memory can preserve appearance but switch identities after occlusion. Evaluation should therefore start from the state queried by the planner, measure the variables that state is expected to retain, and then connect those measures to rollout ranking or closed-loop behavior. Final task success remains necessary, but it cannot explain a failure unless controller settings, action budgets, replanning frequency, and computation are reported with it.

The scope is limited to image-grounded models with an explicit predictive or queryable state. Direct reactive policies and static scene representations appear only where their boundaries clarify the definition. Results obtained with different sensors, embodiments, action spaces, and compute budgets are not numerically interchangeable. Recent systems may also change as their implementations and evaluations mature. Within those limits, the comparison shows why no single carrier supplies detailed appearance, efficient long-horizon prediction, metric safety, and stable identity at once. Hybrid systems can combine these properties, but additional modules help only

DOI: 10.54254/2755-2721/2026.36903

when their addresses agree. The object in a relational memory must correspond to the same region in the map and the same instance in a predicted observation. Query rate, uncertainty, and reference-frame consistency are therefore central design constraints for embodied use.

References

[1] Ha, D., & Schmidhuber, J. (2018). Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*.

[2] Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., & Davidson, J. (2019). Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning, Proceedings of Machine Learning Research, 97*, 2555-2565.

[3] Hafner, D., Pasukonis, J., Ba, J., & Lillicrap, T. (2025). Mastering diverse control tasks through world models. *Nature, 640*(8059), 647-653. https://doi.org/10.1038/s41586-025-08744-2

[4] Long, X., Zhao, Q., Zhang, K., Zhang, Z., Wang, D., Liu, Y., et al. (2025). A survey: Learning embodied intelligence from physical simulators and world models. arXiv preprint arXiv:2507.00917.

[5] Hou, B., Li, G., Jia, J., An, T., Guo, X., Leng, S., et al. (2026). World model for robot learning: A comprehensive survey. arXiv preprint arXiv:2605.00080.

[6] Taniguchi, T., Murata, S., Suzuki, M., Ognibene, D., Lanillos, P., Ugur, E., et al. (2023). World models and predictive coding for cognitive and developmental robotics: Frontiers and challenges. *Advanced Robotics, 37*(13), 780-806. https://doi.org/10.1080/01691864.2023.2225232

[7] Finn, C., & Levine, S. (2017). Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2786–2793). https://doi.org/10.1109/ICRA.2017.7989324

[8] Ye, X., & Bilodeau, G.-A. (2023). Video prediction by efficient transformers. *Image and Vision Computing, 130*, Article 104612. https://doi.org/10.1016/j.imavis.2022.104612

[9] Zhu, C., Yu, R., Feng, S., Burchfiel, B., Shah, P., & Gupta, A. (2025). Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. In *Robotics: Science and Systems XXI*. https://doi.org/10.15607/RSS.2025.XXI.015

[10] Sun, Q., Yang, L., Tang, W., Huang, W., Xu, K., Chen, Y., et al. (2025). Learning primitive embodied world models: Towards scalable robotic learning. *arXiv preprint arXiv:2508.20840*.

[11] Zhou, K., Chen, Y., Zhan, F., Hua, H., Chen, G., Chang, X., et al. (2026). GEM-4D: Geometry-enhanced video world models for robot manipulation. *arXiv preprint arXiv:2605.22882*.

[12] Unterthiner, T., van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., & Gelly, S. (2018). Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*.

[13] Fujii, K., & Murata, S. (2026). Real-world robot control by deep active inference with a temporally hierarchical world model. *IEEE Robotics and Automation Letters, 11*(1), 890–897. https://doi.org/10.1109/LRA.2025.3636032

[14] Hansen, N., Su, H., & Wang, X. (2024). TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations (ICLR)*.

[15] Zhang, W., Terver, B., Zholus, A., Chitnis, S., Sutaria, H., Assran, M., et al. (2026). Hierarchical planning with latent world models. *arXiv preprint arXiv:2604.03208*.

[16] Assran, M., Bardes, A., Fan, D., Garrido, Q., Howes, R., Komeili, M., et al. (2025). V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*.

[17] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020). NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision - ECCV 2020, Lecture Notes in Computer Science*, pp. 405–421. https://doi.org/10.1007/978-3-030-58452-8_24

[18] Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics, 42*(4), Article 139, 1–14. https://doi.org/10.1145/3592433

[19] Ortiz, J., Clegg, A., Dong, J., Sucar, E., Novotny, D., Zollhoefer, M., & Mukadam, M. (2022). iSDF: Real-time neural signed distance fields for robot perception. In *Robotics: Science and Systems XVIII*. https://doi.org/10.15607/RSS.2022.XVIII.012

[20] Liu, T., Zhao, S., Pourkeshavarz, M., Li, W., & Rhinehart, N. (2026). OccSim: Multi-kilometer simulation with long-horizon occupancy world models. arXiv preprint arXiv:2603.28887.

[21] Irshad, M. Z., Comi, M., Lin, Y.-C., Heppert, N., Valada, A., Ambrus, R., et al. (2024). Neural fields in robotics: A survey. arXiv preprint arXiv:2410.20220.

[22] Chen, C., Deng, F., & Ahn, S. (2021). ROOTS: Object-centric representation and rendering of 3D scenes. *Journal of Machine Learning Research, 22*(259), 1-36.

DOI: 10.54254/2755-2721/2026.36903

[23] Ferraro, S., Mazzaglia, P., Verbelen, T., & Dhoedt, B. (2023). FOCUS: Object-centric world models for robotics manipulation. In *RSS 2023 Workshop on Interdisciplinary Exploration of Generalizable Manipulation Policy Learning: Paradigms and Debates*, pp. 1-11.
[24] Wu, Z., Dvornik, N., Greff, K., Kipf, T., & Garg, A. (2023). SlotFormer: Unsupervised visual dynamics simulation with object-centric models. In *International Conference on Learning Representations (ICLR)*.
[25] Goswami, R. G., Krishnamurthy, P., LeCun, Y., & Khorrami, F. (2026). Unifying object-centric world models and diffusion policy: A hierarchical framework for multi-stage robotic tasks. arXiv preprint arXiv:2606.08775.
[26] Gu, Q., Kuwajerwala, A., Morin, S., Jatavallabhula, K. M., Sen, B., Agarwal, A., et al. (2024). ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5021-5028. https://doi.org/10.1109/ICRA57147.2024.10610243
[27] Cen, J., Yu, C., Yuan, H., Jiang, Y., Huang, S., Guo, J., et al. (2025). WorldVLA: Towards autoregressive action world model. arXiv preprint arXiv:2506.21539.
[28] Mees, O., Hermann, L., Rosete-Beas, E., & Burgard, W. (2022). CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters, 7*(3), 7327-7334. https://doi.org/10.1109/LRA.2022.3180108
[29] Arrieta-Ibarra, I., Gujral, P., Tannen, J., Tygert, M., & Xu, C. (2022). Metrics of calibration for probabilistic predictions. *Journal of Machine Learning Research, 23*(351), 1-54.