ARGUS: AN EVOLVING, AUDITABLE EVALUATOR FOR 3D SCENE EDITING AGENTS VIA EVIDENCE-DRIVEN SKILL INDUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

A 3D scene editing agent can produce a plausible scene without completing the requested edit, so evaluators that accurately check the executed result are increasingly important. Image-based judgment often fails when the target is occluded or the change is small, and an aggregate score or binary verdict discards which conditions were checked, whether a checker existed, and whether the evidence sufficed, which is the information repair needs. We propose ARGUS, an experience-driven framework whose primary objective is accurate evaluation and whose downstream objective is scene improvement. It compiles a task specification, binds each requirement to compatible checkers, and combines state, visual, physical, and preservation evidence; confirmed violations go to repair, missing evidence to additional observation, and missing capability to skill learning. A separate pipeline has an LLM propose executable evaluation and repair programs from reviewed evidence. Admission tests each candidate on labeled counterexamples and on a label-free invariance test that replays it under meaning-preserving transformations such as unit conversion and world-frame rotation, which in a controlled test rejects faulty measurements that single-axis labels admit; only programs that pass both become reusable skills. On 48 edits where every evaluator sees only the instruction and the scenes, ARGUS has the lowest selective risk (error rate among decided conditions), 7.0%, and the highest scene AUROC, 81.9, and it holds conditions it cannot settle with the missing evidence named. Registered skills, together with a fixed relation frame in the binder, raise coverage (the fraction of conditions decided) from 44.2% to 60.0% at the same error rate; the skills alone decide 11 held conditions, 10 correctly. Accurate diagnosis carries over to repair, raising the holistic success rate from 56.8% to 65.1%, and admitted skills are reused without LLM calls and revised when new counterexamples appear.

1 INTRODUCTION

An agent can produce a convincing 3D scene while failing the requested edit: the change may be incomplete, non-target objects may have moved, or physical constraints may be violated. Neither a model's confidence nor an editor's self-reported success establishes that the final scene is correct, so an *evaluator* must check these requirements on the executed result, and its accuracy is a research goal in its own right. LEGO-Eval examines fine-grained instruction alignment, SceneEval and SceneCritic assess semantic and spatial properties, and SceneTeract evaluates agent-conditioned physical executability (Kang et al., 2026; Tam et al., 2026; Sengupta et al., 2026; Maillard et al., 2026). We focus on the chain from a requirement to an executed check, its evidence, and the final decision, which has three problems.

A fixed checker set leaves binding and coverage gaps. A suitable checker may be bound to the wrong requirement, or no checker may exist for a needed property. Either way, the mandatory condition should be reported as unevaluated rather than silently dropped from the verdict.

A final verdict alone erases distinctions needed for the next action. An occluded target, a change too small to measure in images, a tool error, and a supported violation call for different

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.

actions. Collapsing them into a score or binary result hides whether more observation, a different check, or a scene edit is needed, and a mandatory failure should never be offset by improvements elsewhere.

Repeated failures do not by themselves produce trustworthy checking capability. Accurate repair requires a sound check, and with a fixed checker set, repeated repair rarely fixes a failure no checker can detect. An LLM can write the missing check anew on every attempt, but such a program may measure the wrong quantity, overfit one scene, or test a weaker condition, and each attempt adds cost. Only programs that pass counterexample and regression tests should be registered as skills and reused deterministically.

ARGUS links two processes. **Scene execution** observes the actual before/after states, fixes the requirements independently of the tools, checks them, and re-evaluates repair candidates under the same specification. **Experience learning** proposes guarded programs from independently reviewed errors and repair trajectories and registers only those that pass admission. The primary goal is an accurate, continually improving evaluator; repair tests whether its diagnoses are useful. Our contributions are:

1. **Diagnosis of checking gaps.** Multiview observation and state, visual, physical, and preservation checkers select a compatible check path per condition, and condition-level evidence records show what was checked and what remains unevaluated.
2. **Admitted capability growth.** LLM-proposed evaluation and repair programs become skills only after passing labeled counterexamples and a label-free invariance test, and then run without further LLM calls.
3. **Evidence-based repair under a fixed specification.** Repair receives evidence of confirmed violations; candidates are re-evaluated under unchanged requirements and accepted only if they make progress.

2 RELATED WORK

3D generation and scene evaluation. Object-level evaluation begins with T^3Bench, which scores quality and alignment from multiview renders, and Eval3D, which adds geometric and structural consistency, and extends to human preferences and hierarchical evaluation in 3DGen-Bench and Hi3DEval (He et al., 2023; Duggal et al., 2025; Zhang et al., 2025a;b). For indoor scenes, LEGO-Eval decomposes descriptions into atomic constraints grounded by tools, SceneEval scores explicit requirements and implicit ones such as collision, support, and accessibility, and SceneCritic judges layout plausibility against a spatial ontology (Kang et al., 2026; Tam et al., 2026; Sengupta et al., 2026). SceneTeract checks agent-specific navigability, reachability, and clearance with geometry and uses that verifier as a reward for VLM post-training (Maillard et al., 2026); PhyMix steers generation with physical plausibility, and 4DWorldBench includes it as one axis of world-generation evaluation (Wu et al., 2026; Lu et al., 2026). We study how the checking gaps and evidence limits these evaluators leave should determine the next action, and how reviewed failures become new executable checks.

3D scene generation and editing agents. LayoutGPT, Holodeck, and I-Design map language to layouts, spatial constraints, and scene graphs with LLMs, and InstructScene uses a diffusion model with a semantic graph prior (Feng et al., 2023; Yang et al., 2024; Çelen et al., 2024; Lin & Mu, 2024). Agentic generators such as SceneCraft, Scenethesis, SceneWeaver, SAGE, and SceneAssistant revise their own scenes iteratively from renders or semantic and physical critiques (Hu et al., 2024; Ling et al., 2026; Yang et al., 2025; Xia et al., 2026; Luo et al., 2026). Editing agents range from Chat-Edit-3D, which changes appearance and removes or replaces objects, and ReSpace, which adds, removes, and swaps objects, to Vinedresser3D, which edits single assets, and Edit-As-Act, which plans by goal regression over the preconditions and effects of EditLang (Fang et al., 2024; Bucher & Armeni, 2026; Chi et al., 2026; Noh et al., 2026). Our common editor follows the Edit-As-Act format; whether such edits are correct is the subject of this paper.

Learning from experience. ReAct interleaves reasoning and acting, and Self-Refine and CRITIC revise outputs with self-feedback and tool-based critique, all within a single task (Yao et al., 2023;

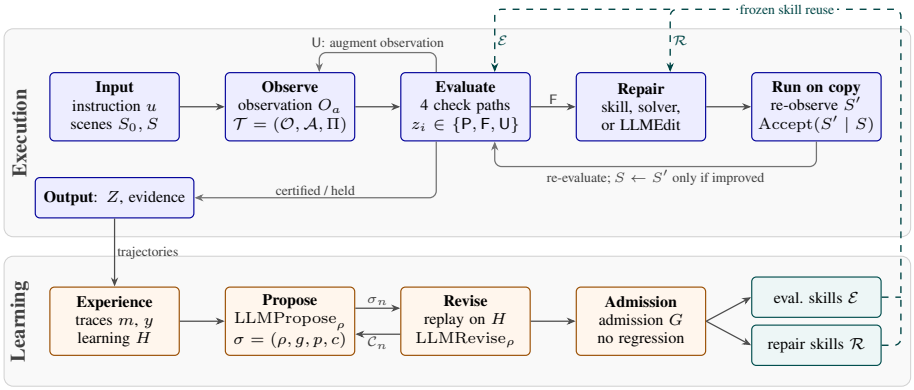

Figure 1: ARGUS overview. Top (scene execution): the evaluator returns pass, fail, or hold for each condition. A hold augments the observation; a failure runs a repair candidate on a copy, re-evaluates it with the same evaluator, and accepts it only if it improves. Bottom (experience learning): an LLM proposes programs from learning H trajectories and independent labels, revises them with counterexamples, and adds only those that pass admission to the library. Dashed lines show frozen skill reuse; evaluation skills only measure, and repair skills write only allowed fields on a copy.

Madaan et al., 2023; Gou et al., 2024). Reflexion, ExpeL, Contextual Experience Replay, and ReasoningBank carry experience across tasks as textual memory of reflections, insights, and strategies (Shinn et al., 2023; Zhao et al., 2024; Liu et al., 2025; Ouyang et al., 2026); Agent Workflow Memory stores induced workflows as text, while Voyager and Agent Skill Induction reuse executable code as skills (Wang et al., 2025b; 2024; 2025a). Program synthesis and repair likewise feed verifier counterexamples, oracle answers, or failing tests into the next candidate (Solar-Lezama et al., 2006; Jha et al., 2010; Orvalho et al., 2025). We bring this to 3D editing: programs derived from experience pass an admission test encoding 3D domain knowledge, are revised with counterexamples, and are stored in two libraries with separate permissions, evaluation skills that only measure and repair skills that write only permitted fields on a copy.

3 METHOD

3.1 OVERVIEW

ARGUS consists of two processes (Figure 1, Algorithm 1). Scene execution observes the actual before/after states, fixes the requirements as a specification, checks them, repairs only confirmed violations, and re-evaluates under the same specification. Experience learning proposes evaluation and repair programs from the trajectories and independent labels of the learning tasks H, revises them with counterexamples, and registers as skills only those that pass admission. Evaluation skills only read the scene, and repair skills write only permitted fields on a copy through the allowed actions of a common editor following the EditLang format of Edit-As-Act (Noh et al., 2026). The LLM only proposes candidate programs; code fixes tolerances, checker binding, and pass/fail.

3.2 OBSERVATION AND FIXED TASK SPECIFICATION

From the instruction u, the pre-edit scene S_0, the executed result S, and the camera set V_a of observation round a, the Observe adapter builds

$$O_a = \text{Observe}(S_0, S; V_a) = (X_0, X, \Gamma, \mathcal{I}_a, \mu_a). \tag{1}$$

X_0, X are object transforms, assets, geometry, and appearance; Γ is the before/after object correspondence; $\mathcal{I}_a$ are calibrated renders; and μ_a records units, frames, omissions (unobserved items), and hashes. Unobserved fields stay unknown rather than estimated, and extra observation changes only O_a, never S. The specification is a fixed criterion of edit correctness compiled from the pre-edit observation O_0^{src} and policy Π (tolerances, preservation, physics):

$$\mathcal{T} = \text{Compile}(u, O_0^{\text{src}}; \Pi) = (\mathcal{O}, \mathcal{A}, \Pi), \quad o_i = (q_i, B_i, Q_i, f_i, \Theta_i). \tag{2}$$

$\mathcal{O}$ is the set of mandatory conditions. Each condition o_i has a predicate q_i, object bindings B_i, a quantifier scope Q_i, a frame f_i, and a satisfaction set Θ_i that includes its tolerance. $\mathcal{A}$ lists the

Algorithm 1 Experience learning and scene execution

Require: Instruction u, scenes S_0, S, base checkers $\mathcal{K}_0$, policy Π, tasks H, G disjoint from E

Experience learning ▷ source only; E is never used

for role $\rho \in \{\text{evaluate}, \text{repair}\}$ **do**
  $\sigma \leftarrow \text{LLMPropose}_\rho(M_H, \text{allowed primitives})$
  while revision budget remains and the source checks or benefit test fail **do**
    Replay on copies from H, collect $\mathcal{C}_n$, and set $\sigma \leftarrow \text{LLMRevise}_\rho(\sigma, \mathcal{C}_n, M_H)$
  end while
  Apply Equation (17) on G; if it passes, register σ in $\mathcal{E}$ or $\mathcal{R}$
end for
Freeze $\mathcal{E}$ and $\mathcal{R}$ before evaluation

Scene execution ▷ after the specification and libraries are frozen

Observe; compile and freeze $\mathcal{T} = (\mathcal{O}, \mathcal{A}, \Pi)$; $D \leftarrow \text{Evaluate}$ on four paths
while budget remains and D is not certified **do**
  Augment the observation for conditions with missing evidence and re-evaluate
  if no confirmed violation or permitted repair remains **then break**
  $\alpha \leftarrow \text{ProposeEdit}$ (a repair skill, else the solver, else LLMEdit)
  $S' \leftarrow \text{ExecuteCopy}(S, \alpha; \mathcal{A})$; re-evaluate under $\mathcal{T}$; if Eq. (11), $(S, D) \leftarrow (S', D')$
end while
Lock and return the verdict Z, condition-level evidence, and reasons for holds

editable objects and properties. Ambiguous meanings stay unresolved, and the specification is never changed to fit an output. A relational selector such as "the chair to the right of the table" is resolved only when f_i maps the dataset's left, right, front, and behind to ground-plane axes. This mapping, the *relation frame*, is a fixed binder component set once on training tasks, not an LLM-proposed skill; the binder picks a target only when exactly one candidate fits and otherwise holds the condition.

3.3 Evaluator: four check paths and evidence authority

For each condition, only compatible tools among the base checkers $\mathcal{K}_0$ and evaluation skills $\mathcal{E}$ are kept:

$$\mathcal{K}_i = \{k \in \mathcal{K}_0 \cup \mathcal{E} : \text{Compatible}(o_i, k; \Pi) = 1\}. \tag{3}$$

Compatibility checks predicate meaning, argument types, quantifier scope, units, frame, and decision authority. An evaluation skill $k \in \mathcal{E}$ is compatible only when the instruction's amount has a single reading, so that Θ_i is unique (the *amount guard*). Let $\mathcal{K}_i^{\text{ready}}(O_a)$ be the checkers whose inputs are present and adequate in the current observation; the next step is

$$\begin{cases} k \in \mathcal{K}_i^{\text{ready}}(O_a), & \mathcal{K}_i^{\text{ready}}(O_a) \neq \varnothing, \\ O_{a+1} = \text{Observe}(S_0, S; V_{a+1}), & \mathcal{K}_i \neq \varnothing,\ \mathcal{K}_i^{\text{ready}}(O_a) = \varnothing, \\ \sigma \leftarrow \text{LLMPropose}_{\text{evaluate}}(M_H), & \mathcal{K}_i = \varnothing. \end{cases} \tag{4}$$

There are four check paths:

$$\begin{aligned} m_j^{\text{state}} &= (t_j - t_j^0,\ R_j (R_j^0)^\top) \text{ compared with } \Theta_i, \\ r_i^{\text{vis}} &= \text{VLM}(q_i, B_i, \mathcal{I}_{a,i}) \text{ matched against } q_i, \\ m_i^{\text{phys}} &= (\widehat{d}_i, \widehat{s}_i, \omega_i) \text{ compared with penetration and contact tolerances}, \\ m^{\text{keep}} &= \{d_\ell(X_\ell, X_\ell^0) : \ell \notin \mathcal{A}\} \text{ compared with } \epsilon_\ell. \end{aligned} \tag{5}$$

The state path compares the translation and relative rotation of object j with Θ_i (requested value and tolerance). The visual path asks a VLM to judge target, appearance, and remaining semantics in selected views. The physical path compares the penetration estimate $\widehat{d}_i$ and support evidence $\widehat{s}_i$ with their limits and records the inspected region ω_i. The preservation path compares the before/after difference of every field outside $\mathcal{A}$ with its tolerance ϵ_ℓ.

An evidence record e holds the condition, checker and version, observation, assumptions, result, and scene and specification hashes. It supports a decision only if

$$\eta_i(e) = \mathbf{1}[\text{Compatible} \wedge \text{Fresh} \wedge \text{Observed} \wedge \text{Adequate}], \tag{6}$$

that is, the checker fits the condition, the hashes match the current scene and specification, and the inputs are present and sufficient (formal definitions in Appendix E). Missing, insufficient, or

conflicting evidence yields unknown U, not pass or fail. Condition decisions $z_i \in \{\mathsf{P}, \mathsf{F}, \mathsf{U}\}$ are aggregated as

$$Z(S;\mathcal{T}) = \begin{cases} \text{certified}, & z_i = \mathsf{P} \text{ for all } i, \\ \text{failed}, & z_i = \mathsf{F} \text{ for some } i, \\ \text{held}, & z_i = \mathsf{U} \text{ for some } i. \end{cases} \tag{7}$$

A mandatory failure is never offset by other passes. The reason for an unknown stays in the evidence record instead of a score and selects the next action: a wrong binding is rematched, an evidence gap triggers more observation, a capability gap triggers evaluation-skill learning, and a confirmed violation goes to repair.

3.4 Repair candidates and acceptance

A repair request contains only confirmed violations:

$$R_{\text{rep}} = \big(\{o_i : z_i = \mathsf{F}\},\ \{B_i : z_i = \mathsf{F}\},\ E_{\text{fail}},\ \mathcal{A}\big), \quad E_{\text{fail}} = \{e : \eta_i(e) = 1,\ d_e = \mathsf{F}\}. \tag{8}$$

Let $\mathcal{R}^{\text{ready}}(R_{\text{rep}})$ be the repair skills whose guards hold, and $\text{Solvable}(R_{\text{rep}}) = 1$ if the violation can be written as constraints on the variables in $\mathcal{A}$. Candidates are produced in the order

$$\alpha = \text{ProposeEdit}(R_{\text{rep}};\mathcal{R}) = \begin{cases} \sigma(R_{\text{rep}}), & \mathcal{R}^{\text{ready}}(R_{\text{rep}}) \neq \varnothing, \\ \text{Solver}(R_{\text{rep}}), & \mathcal{R}^{\text{ready}}(R_{\text{rep}}) = \varnothing,\ \text{Solvable}(R_{\text{rep}}) = 1, \\ \text{LLMEdit}(R_{\text{rep}}), & \mathcal{R}^{\text{ready}}(R_{\text{rep}}) = \varnothing,\ \text{Solvable}(R_{\text{rep}}) = 0. \end{cases} \tag{9}$$

A candidate runs only on a copy of the scene, and the content hash h of the specification and policy must be unchanged:

$$S' = \text{ExecuteCopy}(S, \alpha; \mathcal{A}), \qquad h(S_0, \mathcal{T}, \Pi)_{\text{after}} = h(S_0, \mathcal{T}, \Pi)_{\text{before}}. \tag{10}$$

The re-observed S' is checked against the same S_0 by the frozen evaluator. With P, F, U the passed, failed, and held condition sets (primed for S'), only conservative progress is accepted:

$$\text{Accept}(S' \mid S) = \text{Allowed}(S, S'; \mathcal{A}) \wedge (P \subsetneq P') \wedge (F' \subseteq F) \wedge (U' \subseteq U). \tag{11}$$

At least one condition must newly pass, no pass may be lost, and no failure or hold may be added; acceptance is per-condition progress, not whole-scene certification, and relies on the evaluator's accuracy.

3.5 Experience and counterexamples

Checking and repair experience is stored as traces $m = (u, \mathcal{T}, S_0, S, \text{checks}, \text{edits}, \text{outcomes}, \text{cost})$, split by task into disjoint learning H, admission G, and evaluation E sets. To prevent self-certification, external labels are attached only to $L = L_{\text{num}} \cup L_{\text{sem}} \subseteq \mathcal{O}$:

$$y_i = \begin{cases} \text{Judge}(m_i^{\text{indep}}; \Theta_i), & i \in L_{\text{num}} \text{ (computed)}, \\ \text{Review}(q_i, B_i), & i \in L_{\text{sem}} \text{ (human)}, \\ \text{unlabeled}, & i \notin L. \end{cases} \tag{12}$$

Judge checks whether an independent measurement m_i^{indep}, unused by the evaluator, lies in Θ_i; Review is a human judgment of a semantic condition made blind to the method. The repair label is a scene-level triple

$$y^{\text{rep}} = (s, p, n) \tag{13}$$

of holistic success s (all independent y_i pass), non-target preservation p, and new physical violations n. Learning uses only these labels, never the evaluator's own decisions; an output that disagrees with its label is a counterexample.

3.6 Sandbox revision, admission, and skill libraries

From learning traces and labels, an LLM proposes a skill composed only of allowed primitives. Both skill types share one form:

$$\sigma = (\rho, g, p, c), \qquad \rho \in \{\text{evaluate}, \text{repair}\}. \tag{14}$$

g is a guard, p a program, and c its typed inputs and outputs, assumptions, and permissions. An evaluation program only measures quantities such as distance, angle, count, or colour; a comparator decides pass or fail against the specified Θ_i. A candidate runs in a sandbox on trace copies:

$$(r_j, \ell_j, \xi_j) = \mathrm{SB}_\rho(\sigma, x_j; b), \qquad W_{\text{evaluate}} = \varnothing, \quad W_{\text{repair}} \subseteq \mathcal{A}, \tag{15}$$

where r_j, ℓ_j, ξ_j are the output, log, and status. Evaluation skills write no scene field; repair skills write only fields in $\mathcal{A}$ on a disposable copy. Labels are used only for scoring, never passed to the program, and each counterexample triggers revision:

$$\begin{aligned} \mathrm{Contrast}(H) &= \mathbf{1}[\exists j \in H : y_j = 1] \wedge \mathbf{1}[\exists j \in H : y_j = 0], \\ \mathcal{C}_n &= \{(x_j, y_j, r_j) : j \in H,\ \xi_j = \mathrm{ok},\ \mathrm{RefCheck}_\rho(r_j, y_j) = 0\}, \\ \sigma_{n+1} &= \mathrm{LLMRevise}_\rho(\sigma_n, \mathcal{C}_n, M_H). \end{aligned} \tag{16}$$

H must contain both satisfied and violated labels so that a program that always passes or always fails is caught. Only normally terminated runs that disagree with the label enter $\mathcal{C}_n$, from which LLMRevise revises the candidate.

Admission. A candidate revised on H is judged on G:

$$\mathrm{Admit}_\rho(\sigma; G) = \mathrm{Safe} \wedge \mathrm{Compatible} \wedge \mathrm{Contrast} \wedge \mathrm{NoRegression} \wedge \mathrm{Invariant} \wedge \mathrm{Benefit}_\rho\,. \tag{17}$$

The terms check execution scope, condition compatibility, separation of satisfied and violated cases, retention of solved cases, representation invariance, and benefit over a same-budget baseline. Invariant reads no label: a skill must return the same measurement when a scene changes only in names, units, position, or axes. With $\mathcal{G}_c$ the transformations capability c must ignore and $\mathrm{Measure}(\sigma, x)$ its measurement,

$$\mathrm{Invariant}(\sigma; G, \mathcal{G}_c) = \bigwedge_{j \in G} \bigwedge_{\gamma \in \mathcal{G}_c} \mathbf{1}\big[\mathrm{Measure}(\sigma, \gamma(x_j)) \simeq \mathrm{Measure}(\sigma, x_j)\big]. \tag{18}$$

$\mathcal{G}_c$ is identifier renaming, unit conversion, rigid translation, and world-frame rotation; needing no labels, the test catches measurement faults that labeled cases cannot expose. Only admitted programs enter the versioned library:

$$\mathcal{L}_\rho^{v+1} = \mathcal{L}_\rho^v \cup \{\sigma : \mathrm{Admit}_\rho(\sigma; G) = 1\}, \quad \mathcal{L}_{\text{evaluate}} = \mathcal{E}, \quad \mathcal{L}_{\text{repair}} = \mathcal{R}. \tag{19}$$

A counterexample found after deployment is revised on H and re-admitted on new cases. Learning uses H, selection G, and libraries are frozen before E; an admission case used for revision moves to H and is replaced. Formal definitions and re-admission rules are in Appendices E and B.1.

4 Experiments

4.1 Setup

Data and models. We use 48 single-room, single-instruction edits: 24 from E2A (Noh et al., 2026), labeled by independent human review of the before/after scenes, and 24 from EditRoom (Zheng et al., 2025), labeled by the official source–target state difference; three annotators set the gold by majority vote. Evaluators and repairers see only the instruction and scenes; targets and gold are used only for scoring. Evaluation skills are learned and admitted on one pool of 12 E2A and 12 EditRoom tasks outside the panel (per dataset 8 learning and 4 admission, 3 per edit type, all disjoint), then frozen and only executed at test time. LLM and VLM calls use `gpt-5-2025-08-07` on one NVIDIA L40S.

Metrics. Judged conditions are of three kinds: C0 is the requested change (removal, move, rotation, or size), C1 preservation of objects and properties the instruction does not ask to change, and C2 a side effect against the official target (a numeric candidate). We report the false-acceptance rate (FAR, gold failures judged as passes), the false-rejection rate (FRR, gold passes judged as failures), coverage (Cov, the fraction of conditions decided rather than held), selective risk (Risk, the error rate among decided conditions) (Geifman & El-Yaniv, 2017), and scene AUROC, the probability that a gold-passing scene scores above a gold-failing one (verdicts score pass 1, hold 0.5, fail 0). Precision (P) and recall (R) follow LEGO-Eval (Kang et al., 2026): macro averages over pass and fail on all items, a hold counting as a miss. Details are in Appendices B.2 and D.

Table 1: Evaluator comparison on the 48-edit panel (N: labeled conditions). Cov = decided/N; Risk = wrong/decided; FAR/FRR exclude holds; P/R are macro averages over all items, a hold counting as a miss.

					All (a hold is a miss)		
Method	Cov ↑	FAR ↓	FRR ↓	Risk ↓	P ↑	R ↑	AUROC ↑
Condition level, E2A (24 edits, N = 47; scene labels 15 pass / 3 fail)							
Direct VLM	100.0	**33.3**	18.2	19.1	58.6	**74.2**	66.7
Direct State LLM	87.2	66.7	11.4	17.1	47.2	38.6	48.9
LEGO-Eval (native)	100.0	66.7	13.6	17.0	54.6	59.8	46.7
SceneEval (semantic → C0)	14.9	**33.3**	11.4	85.7	25.0	1.1	36.7
ARGUS L0 (no learned skills)	46.8	**33.3**	**4.5**	13.6	47.5	21.6	67.8
ARGUS L_E (+ registered skills)	66.0	**33.3**	**4.5**	**9.7**	**64.9**	47.3	**90.0**
Condition level, EditRoom (24 edits, N = 48; scene labels 8 pass / 16 fail)							
Direct VLM	100.0	68.4	3.4	29.2	77.0	64.1	65.2
Direct State LLM	83.3	42.1	**0.0**	20.0	88.6	59.7	81.2
LEGO-Eval (native)	100.0	15.8	20.7	18.8	80.6	**81.8**	75.0
SceneEval (semantic → C0)	20.8	**0.0**	6.9	20.0	85.7	18.3	53.1
ARGUS L0 (no learned skills)	41.7	**0.0**	**0.0**	**0.0**	**100.0**	37.2	74.6
ARGUS L_E (+ registered skills)	54.2	5.3	**0.0**	3.8	97.7	46.7	**82.8**
Condition level, pooled (48 edits, N = 95; scene labels 23 pass / 19 fail)							
Direct VLM	100.0	63.6	12.3	24.2	64.6	62.0	60.0
Direct State LLM	85.3	45.5	6.8	18.5	68.0	53.1	67.5
LEGO-Eval (native)	100.0	22.7	16.4	17.9	75.5	**80.4**	73.0
SceneEval (semantic → C0)	17.9	**4.5**	9.6	47.1	60.8	14.1	50.1
ARGUS L0 (no learned skills)	44.2	**4.5**	**2.7**	7.1	78.6	31.5	66.9
ARGUS L_E (+ registered skills)	60.0	9.1	**2.7**	**7.0**	**83.7**	44.2	**81.9**
Scene level, pooled (42 labeled scenes)							
Direct VLM (from conditions)	97.6	52.6	26.1	39.0	60.1	58.0	60.0
Direct State LLM (from conditions)	85.7	42.1	17.4	33.3	63.0	54.5	67.5
LEGO-Eval (from conditions)	100.0	10.5	43.5	28.6	74.8	**73.0**	73.0
LEGO-Eval (native success)	100.0	10.5	60.9	38.1	68.3	64.3	64.3
SceneCritic (score calibration)	52.4	10.5	17.4	27.3	70.8	40.3	75.1
SceneWeaver (score calibration)	52.4	5.3	26.1	31.8	72.9	37.2	65.0
SceneEval (native)	N/A	N/A	N/A	N/A	N/A	N/A	69.2
Edit-As-Act (native acceptance)	N/A	N/A	N/A	N/A	N/A	N/A	51.1
ARGUS L0 (from conditions)	23.8	**0.0**	**4.3**	10.0	**90.0**	21.4	66.9
ARGUS L_E (from conditions)	52.4	5.3	**4.3**	**9.1**	89.5	46.2	**81.9**

4.2 Comparison with external evaluators

Table 1 compares LEGO-Eval (Kang et al., 2026), SceneEval (Tam et al., 2026), SceneCritic (Sengupta et al., 2026), the evaluators of SceneWeaver (Yang et al., 2025) and Edit-As-Act (Noh et al., 2026), Direct VLM, and Direct State LLM with ARGUS L0 (no learned skills) and L_E (registered skills plus the relation frame, Section 3.2). "From conditions" rows fail a scene if any condition fails, pass it if all pass, and hold it otherwise. Score-only SceneCritic and SceneWeaver are thresholded with cutoffs calibrated on the other dataset and hold their least confident scenes to decide as many as L_E (score calibration, Appendix D); SceneEval and Edit-As-Act ratios are scored by AUROC only.

ARGUS makes the most reliable decisions. Deciding only what it can settle and holding the rest with a stated reason, L_E errs on 4 of its 57 decided conditions (LEGO-Eval 17, Direct State LLM 15, Direct VLM 23). Pooled at the condition level, it has the lowest Risk (7.0), the highest P (83.7) and AUROC (81.9), and the lowest FRR (2.7, tied with L0); at the scene level it also has the best Risk (9.1) and AUROC (81.9). Each of the 38 holds carries a reason code naming the missing evidence or checker (Appendix C.3). Registered skills decide 11 L0 holds, 10 correctly, and the relation frame resolves 4 relational selectors, raising coverage from 44.2% to 60.0%. Full-set recall favors LEGO-Eval (80.4 vs. 44.2) because it decides every condition while the 38 holds count as misses. On the same 57 conditions L_E decides, LEGO-Eval reaches P/R 71.9/85.9 against 83.7/83.7, so the advantage of L_E is precision, not recall. The amount guard (Appendix C.3) was added after error analysis and blocks three false accepts, so it is post hoc. With 4/57 errors, the risk interval of L_E [1.9, 17.0] overlaps that of LEGO-Eval (Appendix C.1).

4.3 Repair results

Table 2 fixes the initial outputs, editor, permissions, and budget and varies only the feedback; only the last row uses repair skills. IF, SC, and PP are the instruction-fidelity, non-target semantic-consistency, and physical-plausibility scores of the Edit-As-Act rubric (Noh et al., 2026). The

Table 2: Repair with different feedback under the same editor, permissions, and budget; linked variants share the common editor with ARGUS. IF/SC/PP are the mean of three judge runs (†: one run). No repair is the reference and is not bolded. On IF/SC/PP, a row is bolded as a tie if its gap to the best row is within the run-to-run judge SD (IF 1.1, SC 1.1, PP 0.7) and its edit-paired 95% interval includes zero (Appendix C.1).

Feedback / repair method	IF ↑	SC ↑	PP ↑	HSR ↑	HSR_{48} ↑	C-SR ↑	NET ↑	OVR ↓	Calls ↓
No repair (initial output)	55.1	82.4	89.9	56.8	52.1	75.0	0	0.0	0
Direct VLM + common editor	**58.8**	**83.4**	**89.6**	59.5	52.1	76.2	0	20.0	0.92
LEGO-Eval feedback (linked variant)	52.3	79.1	89.1	47.7	43.8	70.5	−4	28.0	1.02
SceneCritic feedback (linked variant)	54.0	79.3	89.1	50.0	43.8	71.4	−4	56.0	1.21
SceneCritic feedback (native)†	51.2	68.1	72.5	35.3	25.0	–	–	–	–
ARGUS feedback, no repair skills	59.2	**84.2**	**90.1**	60.5	54.2	76.7	+1	8.0	0.35
ARGUS feedback + repair skills	**59.6**	**84.1**	**89.5**	**65.1**	**58.3**	**79.1**	**+3**	**4.0**	**0.17**



Figure 2: Qualitative comparison of repairs driven by different feedback (top: E2A, bottom: EditRoom; the instruction is shown below each row; all columns share the viewpoint and crop). The E2A gaming table already passes, and only ARGUS keeps it passing; the EditRoom corner table initially fails, and only ARGUS repairs it.

holistic success rate (HSR) is the fraction of labeled edits whose independent C0 and C1 both pass (HSR_{48} divides by all 48, counting unresolved edits as failures), and the condition success rate (C-SR) is the fraction of C0 and C1 conditions passed, the partial SR of LEGO-Eval (Kang et al., 2026). NET is the 19 pre-repair failures recovered minus the 25 pre-repair passes lost; OVR (over-editing) is the fraction of those 25 whose scene repair changed; Calls is repair-stage LLM calls per edit (Wang et al., 2025a), excluding precomputed feedback.

ARGUS feedback with repair skills repairs best: it is highest on IF, HSR, HSR_{48}, C-SR, and NET and lowest on OVR and Calls, fixing the most failed edits (4 of 19, all EditRoom), changing the fewest correct scenes (1 of 25), and using the fewest calls (0.17 per edit vs. 0.35 without skills and 0.92–1.21 with external feedback). On SC and PP it ties with ARGUS feedback without skills (paired differences SC -0.1 $[-1.4, +1.7]$, PP -0.6 $[-1.8, +0.2]$); Direct VLM also ties on IF/SC/PP but trails on HSR, C-SR, OVR, and Calls. The two ARGUS rows deliver the same scene on 42 of 48 edits, so the SC/PP gap comes from six edits, mainly two on E2A: on the gaming chair the judge sees no rotation and lowers SC with IF, and on the rug enlargement (bedroom:5) ARGUS holds because it cannot account for target changes outside the reviewed fields, yet the repaired scene is delivered. Reverting held repairs would block this edit but also undo the four recoveries, since all six skill-changed edits end held. Both ARGUS rows lose the E2A gaming-chair rotation that specification compilation wrongly fails (Appendix C.3). Linked LEGO-Eval recovers one and loses five, SceneCritic three and seven, and Direct VLM four and four, so only ARGUS feedback has a positive NET, largest with repair skills (+3); the five edits without an ARGUS contract stay unchanged (Appendix C.1).

Table 3: Cumulative addition of modules. M: checker binding; V: evaluation skills; C: three-way aggregation; F: feedback; R: repair skills. EVC is self-reported evidence sufficiency, not an independent audit. Repair rows are the runs of Table 2, bolded under its tie rule; R0 is the reference and is not bolded.

Evaluation stage	M	V	C	Risk ↓	FAR ↓	FRR ↓	Cov. ↑	EVC ↑
V0: base checks	–	–	–	24.4	81.8	**1.4**	82.1	49.5
V1: + binding	✓	–	–	21.8	68.2	2.7	82.1	54.7
V2: + evaluation skills	✓	✓	–	18.3	59.1	2.7	**86.3**	**73.7**
V3: + three-way aggregation	✓	✓	✓	**7.0**	**9.1**	2.7	60.0	**73.7**
Repair stage	**F**	**R**		**IF ↑**	**SC ↑**	**PP ↑**	**HSR ↑**	**NET ↑ / OVR ↓**
R0: editor with instruction only	–	–		55.1	82.4	89.9	56.8	0 / 0.0
R1: + evaluation feedback	✓	–		59.2	**84.2**	**90.1**	60.5	+1 / 8.0
R2: + repair skills	✓	✓		**59.6**	**84.1**	**89.5**	**65.1**	**+3 / 4.0**

Table 4: Skill 2×2 on six EditRoom houses. Pose success is numeric, not certification; calls are test-time repair calls.

Setting	Pose success	Recovered	LLM calls
V off / R off	12/18	6/12	6
V on / R off	18/18	12/12	12
V on / R on	18/18	12/12	0
V off / R on	12/18	6/12	0

Table 5: Label-only admission versus admission with Invariant. Parentheses give label errors (label admission) or frame-rotation violations (Invariant) on single-axis conditions. Unseen errors are on 30 held-out conditions and are all false rejections.

	Label admission (errors)			
Candidate	all	single-axis	Invariant	Unseen errors
SO(3) relative angle	pass (0/18)	pass (0/15)	pass (0/15)	0/30
Euler-difference norm	fail (3/18)	pass (0/15)	fail (12/15)	4/30
yaw only	fail (3/18)	pass (0/15)	fail (12/15)	5/30

5 Ablation

Cumulative evaluation and repair modules. Table 3 adds binding (V1), evaluation skills (V2), and three-way pass/fail/hold aggregation (V3) to base checks (V0), and feedback (R1) and repair skills (R2) to an instruction-only editor (R0). Evidence coverage (EVC) is the fraction of conditions whose evidence record shows a sufficient check. V1 and V2 lower Risk from 24.4 to 18.3 and raise EVC from 49.5 to 73.7; V3 (L_E) holds unsupported checks, cutting FAR from 59.1 to 9.1 and Risk to 7.0. R1 raises HSR from 56.8 to 60.5 and R2 to 65.1; NET rises from $+1$ to $+3$ and OVR falls from 8.0 to 4.0.

2×2 contribution of the skills and label-free admission. Table 4 toggles rotation evaluation (V) and repair (R) skills on six EditRoom houses; pose success counts the 18 states (6 correct, 12 violating) whose final pose meets the numeric requirement, and recovered counts fixed violating states. The evaluation skills, registered on E2A before the joint library and admitted on four disjoint EditRoom training houses (36/36 correct; the base checker decides none), raise pose success from 12/18 to 18/18 and recovery from 6/12 to 12/12; repair alone does neither, and both together cut calls to 0. Invariant rejects the two faulty norms that single-axis labels admit (Table 5).

6 Conclusion

ARGUS makes the most trustworthy decisions among the compared evaluators, provides evidence for each condition, repairs accurately from that evidence, and grows its evaluation capability safely from experience. On 48 instruction-only edits it is wrong on 4 of its 57 decided conditions (selective risk 7.0 vs. 17.9 or more for every other evaluator) and has the highest scene AUROC (81.9); each hold names the missing evidence or checker. Admitted skills and the relation frame widen coverage from 44.2% to 60.0% at the same selective risk (7.1→7.0), and in a controlled test the invariance test rejects, without labels, constructed faulty programs that single-axis label review admits. Holistic success rises from 56.8% to 65.1% while only one of 25 correct scenes changes, whereas external feedback on the same editor loses more edits than it recovers. Because admitted skills run without LLM calls, accurate diagnosis also makes repair cheaper (0.17 repair calls per edit vs. 0.35 without skills and 0.92–1.21 with external feedback), and because every decision and hold carries its evidence record, each gain in evaluation capability remains auditable.

AI USE STATEMENT

Generative AI was used to assist in preparing this document. The proposed method also uses language models to generate candidate programs, as described in the methodology. No experimental measurements were generated.

REPRODUCIBILITY STATEMENT

The appendices specify data splits, metric denominators, comparison scope, admission tests, and planned result tables, and Appendix F gives the full text of every language-model prompt.

ETHICS STATEMENT

Certification only checks that the conditions of the digital specification fixed by the instruction and the scene are met. Reviewers who inspect before-and-after scenes to set gold labels give consent, and we record who reviewed which scene. Code and 3D assets follow the original licenses of their datasets, models, and assets. Independently grounded review is necessary to avoid reinforcing a model's own mistakes.

REFERENCES

Martin JJ. Bucher and Iro Armeni. ReSpace: Text-driven autoregressive 3D indoor scene synthesis and editing. In *ICLR Workshop on Efficient Spatial Reasoning*, 2026. URL https://arxiv.org/abs/2506.02459.

Ata Çelen, Guo Han, Konrad Schindler, Luc Van Gool, Iro Armeni, Anton Obukhov, and Xi Wang. I-Design: Personalized LLM interior designer. In *Computer Vision – ECCV 2024 Workshops*, volume 15624 of *Lecture Notes in Computer Science*, pp. 217–234. Springer, 2024. doi: 10.1007/978-3-031-92387-6_17. URL https://arxiv.org/abs/2404.02838.

Yankuan Chi, Xiang Li, Zixuan Huang, and James M. Rehg. Vinedresser3D: Towards agentic text-guided 3D editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12673–12683, 2026. URL https://arxiv.org/abs/2602.19542.

Shivam Duggal, Yushi Hu, Oscar Michel, Aniruddha Kembhavi, William T. Freeman, Noah A. Smith, Ranjay Krishna, Antonio Torralba, Ali Farhadi, and Wei-Chiu Ma. Eval3D: Interpretable and fine-grained evaluation for 3D generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13326–13336, 2025. doi: 10.1109/CVPR52734.2025.01244. URL https://arxiv.org/abs/2504.18509.

Shuangkang Fang, Yufeng Wang, Yi-Hsuan Tsai, Yi Yang, Wenrui Ding, Shuchang Zhou, and Ming-Hsuan Yang. Chat-Edit-3D: Interactive 3D scene editing via text prompts. In *European Conference on Computer Vision*, pp. 199–216, 2024. doi: 10.1007/978-3-031-72946-1_12. URL https://arxiv.org/abs/2407.06842.

Weixi Feng, Wanrong Zhu, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. LayoutGPT: Compositional visual planning and generation with large language models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.15393.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4885–4894, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/4a8423d5e91fda00bb7e46540e2b0cf1-Abstract.html.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.11738.

Yuze He, Yushi Bai, Matthieu Lin, Wang Zhao, Yubin Hu, Jenny Sheng, Ran Yi, Juanzi Li, and Yong-Jin Liu. T^3Bench: Benchmarking current progress in text-to-3D generation. *arXiv preprint arXiv:2310.02977*, 2023. URL https://arxiv.org/abs/2310.02977.

Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A. Ross, Cordelia Schmid, and Alireza Fathi. SceneCraft: An LLM agent for synthesizing 3D scenes as Blender code. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *PMLR*, pp. 19252–19282, 2024. URL https://arxiv.org/abs/2403.01248.

Susmit Jha, Sumit Gulwani, Sanjit A. Seshia, and Ashish Tiwari. Oracle-guided component-based program synthesis. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*, pp. 215–224, 2010. doi: 10.1145/1806799.1806833.

Minseok Kang, Dongwook Choi, Gyeom Hwangbo, Seungwon Lim, Kai Tzu-iunn Ong, and Jinyoung Yeo. Apples on the table? evaluating text-guided 3D scene synthesis via fine-grained constraint verification. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, 2026. URL https://arxiv.org/abs/2511.03001.

Chenguo Lin and Yadong Mu. InstructScene: Instruction-driven 3D indoor scene synthesis with semantic graph prior. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2402.04717.

Lu Ling, Chen-Hsuan Lin, Tsung-Yi Lin, Yifan Ding, Yu Zeng, Yichen Sheng, Yunhao Ge, Ming-Yu Liu, Aniket Bera, and Zhaoshuo Li. Scenethesis: A language and vision agentic framework for 3D scene generation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2505.02836.

Yitao Liu, Chenglei Si, Karthik R. Narasimhan, and Shunyu Yao. Contextual experience replay for self-improvement of language agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14179–14198, 2025. doi: 10.18653/v1/2025.acl-long.694.

Yiting Lu, Wei Luo, Peiyan Tu, Haoran Li, Hanxin Zhu, Zihao Yu, Xingrui Wang, Xinyi Chen, Xinge Peng, Xin Li, and Zhibo Chen. 4DWorldBench: A comprehensive evaluation framework for 3D/4D world generation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 34322–34332, 2026. URL https://arxiv.org/abs/2511.19836.

Jun Luo, Jiaxiang Tang, Ruijie Lu, and Gang Zeng. SceneAssistant: A visual feedback agent for open-vocabulary 3D scene generation. *arXiv preprint arXiv:2603.12238*, 2026. URL https://arxiv.org/abs/2603.12238.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2303.17651.

Léopold Maillard, Francis Engelmann, Tom Durand, Boxiao Pan, Yang You, Leonidas Guibas, and Maks Ovsjanikov. SceneTeract: Probing and improving agent-aware activity reasoning in 3D indoor scenes. *arXiv preprint arXiv:2603.29798*, 2026. URL https://arxiv.org/abs/2603.29798.

Seongrae Noh, SeungWon Seo, Gyeong-Moon Park, and HyeongYeop Kang. Edit-As-Act: Goal-regressive planning for open-vocabulary 3D indoor scene editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19864–19873, 2026. URL https://arxiv.org/abs/2603.17583.

Pedro Orvalho, Mikoláš Janota, and Vasco M. Manquinho. Counterexample guided program repair using zero-shot learning and MaxSAT-based fault localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 649–657, 2025. doi: 10.1609/aaai.v39i1.32046.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.25140.

Kathakoli Sengupta, Kai Ao, and Paola Cascante-Bonilla. SceneCritic: A symbolic evaluator for 3D indoor scene synthesis. *arXiv preprint arXiv:2604.13035*, 2026. URL https://arxiv.org/abs/2604.13035.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2303.11366.

Armando Solar-Lezama, Liviu Tancau, Rastislav Bodík, Sanjit A. Seshia, and Vijay A. Saraswat. Combinatorial sketching for finite programs. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 404–415, 2006. doi: 10.1145/1168857.1168907.

Hou In Ivan Tam, Hou In Derek Pun, Austin T. Wang, Angel X. Chang, and Manolis Savva. SceneEval: Evaluating semantic coherence in text-conditioned 3D indoor scene synthesis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 7355–7365, 2026. doi: 10.1109/WACV61042.2026.00710. URL https://arxiv.org/abs/2503.14756.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2305.16291.

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks. In *Conference on Language Modeling*, 2025a. URL https://arxiv.org/abs/2504.06821.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *PMLR*, pp. 63897–63911, 2025b. URL https://arxiv.org/abs/2409.07429.

Dongli Wu, Jingyu Hu, Ka-Hei Hui, Xiaobao Wei, Chengwen Luo, Jianqiang Li, and Zhengzhe Liu. PhyMix: Towards physically consistent single-image 3D indoor scene generation with implicit–explicit optimization. *arXiv preprint arXiv:2604.10125*, 2026. URL https://arxiv.org/abs/2604.10125.

Hongchi Xia, Xuan Li, Zhaoshuo Li, Qianli Ma, Jiashu Xu, Ming-Yu Liu, Yin Cui, Tsung-Yi Lin, Wei-Chiu Ma, Shenlong Wang, Shuran Song, and Fangyin Wei. SAGE: Scalable agentic 3D scene generation for embodied AI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22358–22368, 2026. URL https://arxiv.org/abs/2602.10116.

Yandan Yang, Baoxiong Jia, Shujie Zhang, and Siyuan Huang. SceneWeaver: All-in-one 3D scene synthesis with an extensible and self-reflective agent. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2509.20414.

Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, Chris Callison-Burch, Mark Yatskar, Aniruddha Kembhavi, and Christopher Clark. Holodeck: Language guided generation of 3D embodied AI environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16227–16237, 2024. doi: 10.1109/CVPR52733.2024.01536. URL https://arxiv.org/abs/2312.09067.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Yuhan Zhang, Mengchen Zhang, Tong Wu, Tengfei Wang, Gordon Wetzstein, Dahua Lin, and Ziwei Liu. 3DGen-Bench: Comprehensive benchmark suite for 3D generative models. *arXiv preprint arXiv:2503.21745*, 2025a. URL https://arxiv.org/abs/2503.21745.

Yuhan Zhang, Long Zhuo, Ziyang Chu, Tong Wu, Zhibing Li, Liang Pan, Dahua Lin, and Ziwei Liu. Hi3DEval: Advancing 3D generation evaluation with hierarchical validity. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2025b. URL https://arxiv.org/abs/2508.05609.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936.

Kaizhi Zheng, Xiaotong Chen, Xuehai He, Jing Gu, Linjie Li, Zhengyuan Yang, Kevin Lin, Jianfeng Wang, Lijuan Wang, and Xin Eric Wang. EditRoom: LLM-parameterized graph diffusion for composable 3D room layout editing. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.12836.