CAPER: PartGround and PlanRefine for Reliable Manual-Guided Robotic Assembly Planning

Hanzhao Xi[1], Zhonghao Li[2], Suxuan Liu[2], Shanghan Zhou[2], Zesheng Chen[2], Qiufeng Wang[3], Yaran Chen*
Xi'an Jiaotong-Liverpool University

***Abstract*— Vision-Language-Models (VLMs) used in robot task planning to derive assembly procedures from existing manuals may correctly interpret individual assembly actions yet confuse exact part identities, predict unsupported Part IDs, or produce inconsistent parts across assembly steps. Thus, we propose CAPER (Capacity-Aware Part-grounding and Evidence Refinement), a manual-guided robotic assembly framework that enables VLMs to generate assembly procedures with two modules—PartGround and PlanRefine. PartGround aligns step-level manual evidence with product-specific inventory and part-reference information to identify exact Part IDs and quantities, transforming free-form hardware prediction into evidence-guided assignments represented in a shared Assembly Intermediate Representation (Assembly IR). PlanRefine audits PartGround assignments against finite inventory by removing the lowest-confidence assignments. Uncertain identity cases remain marked for review in the Assembly IR. We evaluate CAPER on the IKEA Assembly in the Wild (IAW) dataset. Experiment results show that CAPER reaches a Finite-inventory Consistency Rate of 83.33% and produces zero observed out-of-inventory Part-ID occurrences. At the plan level, PlanRefine removes all observed inventory-overuse pairs (9→0) and excess units (51→0). Overall, the results indicate that PartGround and PlanRefine improve part reliability, while also enhancing finite-inventory consistency and local prediction accuracy.**
Project page and video: xhzzzz123.github.io/CAPER/.

I. Introduction

Enabling robots to derive assembly procedures from human-readable manuals provides a practical path toward flexible manufacturing. Rather than programming each product individually, a robot can use existing illustrations, annotations, and part references to infer the required operations and components. Recent vision-language models (VLMs) make this increasingly feasible by combining high-level task knowledge with estimates of skill feasibility [2], or by generating structured plans from available actions, objects, and example programs [3].

Recent studies have explored this problem from complementary perspectives. Manual2Skill extracts hierarchical assembly graphs from instructional images and couples them with pose estimation and robot execution [5], while Manual2Skill++ further incorporates connector-level information, including connector type, specification, quantity, and spatial relations, for more detailed assembly reasoning [7]. Related planning methods such as SayCan [2], ProgPrompt [3], and VLM-TAMP [6] introduce affordance, structural, or task-and-motion constraints, while KnowNo [4] considers uncertainty-aware intervention. To better position the reliability problem studied in this work, TableI summarizes the representative methods in terms of manual input, structured planning, exact Part-ID matching, part quantities, finite-inventory consistency, reviewability, which is intended to characterize the information and constraints handled by each method. Yet semantic understanding alone cannot ensure correct part grounding or sequence-wide feasibility.

At the step level, a key difficulty is **exact part-identity grounding**. A model may correctly recognize that a step requires four screws while still failing to determine which specific screws in the product inventory should be used. Assembly manuals often contain simplified drawings, small hardware illustrations, visually similar components, and sparse or distributed annotations, making this correspondence difficult to establish. Manual2Skill++ [7], for example, explicitly represents connector quantities and specifications, but this is distinct from grounding each assembly step to the exact product-specific Part ID. Consequently, a model may generate a Part ID that is absent from the inventory or select a valid but incorrect candidate. This distinction is critical for robotic assembly, where the downstream system must retrieve and manipulate a specific physical component rather than merely recognize its semantic category.

At the plan level, locally plausible assignments may still become inconsistent when combined across the complete assembly sequence. Multiple steps may collectively consume more instances of a Part ID than are available, while ambiguous evidence may lead to conflicting assignments across steps. Existing planning methods address related forms of reliability, including skill affordances [2], structured task constraints [3], task-and-motion feasibility [6], and uncertainty-aware intervention [4]. However, **among the representative methods summarized in Table I, exact Part-ID grounding, part quantities, and finite-inventory consistency are not jointly modeled as a unified material-level constraint over the complete assembly plan**. Reliable assembly planning therefore requires not only semantically plausible local steps but also physically grounded assignments that remain consistent with the available inventory throughout the procedure.

These two reliability gaps require different but complementary forms of reasoning. Exact part identification requires aligning step-level manual evidence with product-specific inventory and part-reference information, whereas plan validation requires examining how the grounded parts are accumulated and reused across the full sequence. A structured plan alone cannot guarantee either property: a well-formed prediction may still select the wrong physical part, while an assignment restricted to legal Part IDs may still

*Corresponding author: Yaran Chen (email: yaran.chen@xjtlu.edu.cn).

TABLE I

FUNCTIONAL COVERAGE OF ASSEMBLY PLANNING METHODS.

Method	Assembly Manual Input	Structured Assembly Plan	Exact Part-ID Matching	Part Quantities	Finite-inventory Consistency	Review / Abstention Support	Reviewable
Sera et al. [1]	✓	✓	✗	✗	✗	✗	✗
SayCan [2]	✗	✗	✗	✗	✗	✗	✗
ProgPrompt [3]	✗	✓	✗	✗	✗	✗	✗
KnowNo [4]	✗	✗	✗	✗	✗	✓	✗
Manual2Skill [5]	✓	✓	✗	✗	✗	✗	✗
VLM-TAMP [6]	✗	✓	✗	✗	✗	✗	✗
Manual2Skill++ [7]	✓	✓	✗	✓	✗	✗	✗
CAPER (Ours)	✓	✓	✓	✓	✓	✓	✓

Note: Relative to the other rows, CAPER adds **Exact Part-ID Matching**, **Finite-inventory Consistency**, and **Reviewable**, which are unmarked for every compared method. It also includes **Part Quantities** (shared only with Manual2Skill++) and **Review / Abstention Support** (shared only with KnowNo), making it the only method marked for all seven functions. Within CAPER, **Exact Part-ID Matching** and **Part Quantities** are produced by PartGround, **Finite-inventory Consistency** is produced by PlanRefine, and the two review-related columns are jointly realized through the shared Assembly IR. This added coverage is important for strengthening robot-arm assembly: the plan then specifies exact physical parts and remains consistent with finite inventory, rather than stopping at step semantics.

choose the wrong legal candidate. Likewise, an assignment that appears reasonable locally may violate finite-inventory constraints when aggregated across multiple steps. This motivates a framework that connects evidence-based physical grounding with plan-level consistency checking through a shared structured representation.

To this end, we propose **CAPER (Capacity-Aware Part-grounding and Evidence Refinement)**, a manual-guided assembly planning framework composed of two complementary modules: **PartGround** and **PlanRefine**. CAPER represents each assembly step using a shared **Assembly Intermediate Representation (Assembly IR)** that records the operation, exact Part IDs, quantities, and supporting manual evidence. PartGround addresses part-identity hallucinations by linking step-level visual evidence with product-specific inventory references, transforming free-form hardware prediction into evidence-guided physical grounding. PlanRefine addresses plan inconsistency by auditing the grounded assignments against finite inventory: when predicted consumption exceeds available quantity, it removes the lowest-confidence assignments until finite-inventory consistency is restored. In this way, PartGround focuses on the question of which physical part belongs to each step, while PlanRefine focuses on whether the resulting assignments satisfy finite-inventory consistency. Together, the two modules connect semantic manual understanding with exact physical part grounding and plan-level inventory validation, rather than treating a single VLM prediction as the final assembly decision.

We evaluate CAPER on the *IKEA Assembly in the Wild (IAW)* dataset [8] through model adaptation, Part-ID grounding, and failure-aware end-to-end evaluation. The experiments show that additional model adaptation alone does not consistently resolve exact physical grounding, motivating explicit reliability mechanisms outside the VLM. CAPER produces zero observed out-of-inventory Part-ID occurrences and reaches an 83.33% Finite-inventory Consistency Rate. At the plan level, PlanRefine removes all observed cross-step inventory-overuse pairs (9→0) and excess units (51→0). Part-ID F1 changes only slightly from 55.24% before PlanRefine to 55.77% after refinement, showing that the primary role of PlanRefine is finite-inventory consistency rather than local identity accuracy. Taken together, these results show that legal part selection, correct physical grounding, and finite-inventory consistency are related but distinct reliability requirements. They also indicate that the most difficult remaining problem is not simply restricting outputs to legal IDs, but selecting the correct legal physical identity when the manual evidence is ambiguous.

Our contributions are threefold:

- **A reliability-oriented framework for manual-guided assembly planning.** We identify two key gaps between semantic assembly understanding and reliable physical execution: exact product-specific Part-ID grounding and cross-step finite-inventory consistency. To address them, we propose CAPER, a unified framework that connects structured manual understanding with physical part grounding and plan-level inventory validation through a shared Assembly IR.
- **Evidence-guided exact Part-ID grounding.** We introduce PartGround to resolve part-identity ambiguity by aligning step-level manual evidence with product-specific inventory and part-reference information, and retrieving additional evidence when necessary. This converts free-form hardware prediction into evidence-supported assignments of exact Part IDs and quantities.
- **Finite-inventory-aware plan refinement.** We introduce PlanRefine to audit grounded Part-ID assignments across the complete assembly sequence and repair inventory conflicts by removing the lowest-confidence assignments when predicted consumption exceeds available quantities, while preserving unresolved cases for review.

II. RELATED WORK

A. Assembly Planning from Human Instructions

Early work showed that graphical manuals can be converted into structured robot assembly plans. Sera et al. build an Assembly Task Sequence Graph from a graphical instruction manual, infer missing information across the image sequence, and check total part counts before robot execution [1]. Manual2Skill uses a VLM to extract hierarchical assembly graphs

and couples them with pose estimation, motion planning, and real IKEA furniture assembly [5]. Manual2Skill++ extends this line with connector-aware graphs that represent connector type, spatial relation and so on [7]. LLM-as-BT-Planner studies another structured representation, generating behavior trees for robotic assembly tasks [9]. These works establish the value of structured representations for instruction-guided assembly. CAPER focuses on other questions: whether each step is linked to the exact product-specific hardware identity and whether those assignments remain consistent with the inventory of final plan.

B. Grounded Robot Planning

Language- and vision-based robot planning often separates semantic reasoning from feasibility checks. SayCan combines language-model preferences with robot skill affordances [2]. ProgPrompt constrains generated plans with program-like actions, objects, and preconditions [3]. VLM-TAMP uses VLM-generated subgoals to guide task-and-motion planning and replans when a subgoal cannot be refined into a feasible motion plan [6]. These methods address action or motion feasibility in the robot environment. CAPER operates at the earlier layers: it checks whether the selected physical hardware is supported by the manual and whether the complete set of Part-ID assignments respects finite inventory.

C. Uncertainty and Failure Awareness

KnowNo uses conformal prediction to identify uncertain language-model plans and request human help when needed [4]. Inner Monologue [10] and SymSkill [11] also use explicit feedback or recovery mechanisms to improve long-horizon robot behavior. CAPER's review is tied to evidence quality, unresolved Part-ID assignments, upstream gate failures, and inventory conflicts. This distinction is important because a grounding-quality state and a probability of task success are different quantities.

Table I summarizes the problem coverage of representative methods. The table is a capability comparison since the methods use different datasets, tasks, and evaluation protocols.

III. CAPER Framework

A. Overall Framework

As illustrated in Fig.1, CAPER is designed to address two reliability problems in manual-guided robotic assembly planning: exact physical part identification and finite-inventory consistency. Although a vision-language model (VLM) can recover the semantic meaning of an assembly instruction, direct generation may still produce unsupported Part IDs, confuse visually similar components, or generate part assignments that become inconsistent across multiple steps. CAPER therefore uses the VLM only as a semantic planner and delegates these two problems to two dedicated modules: **PartGround** and **PlanRefine**. PartGround grounds semantic hardware descriptions to product-specific Part IDs, while PlanRefine checks whether the resulting assignments remain consistent with the finite inventory over the complete assembly sequence.

CAPER represents the intermediate and final results using a shared **Assembly Intermediate Representation (Assembly IR)**. For each assembly step, the Assembly IR records the predicted operation, action description, grounded Part IDs, quantities, supporting evidence, warnings, and grounding-quality information. This representation allows semantic understanding, physical grounding, and plan-level consistency checking to remain explicitly connected. The main design of CAPER is therefore to separate semantic reasoning from exact Part-ID determination and cross-step inventory validation, rather than relying on a single VLM prediction as the final assembly decision. Fig. 1 illustrates this organization and shows how the structured output is connected to downstream robotic assembly.

B. PartGround: Evidence-Guided Part-ID Grounding

PartGround addresses the question of which exact physical part should be used in each assembly step. A semantic prediction such as "screw" or "connector" is insufficient for robotic execution when several product-specific candidates have similar appearance or function. PartGround therefore reformulates Part-ID prediction from unrestricted generation into an inventory-constrained grounding problem. For each assembly step s, it receives the localized manual evidence E_s, the semantic prediction M_s, the product-specific candidate inventory $\mathcal{C}$, and visual references of the candidate parts. The candidate inventory is represented as

$$\mathcal{C} = \{c_i = (id_i, Q_i, r_i, v_i)\}_{i=1}^{N}, \tag{1}$$

where id_i denotes the product-specific Part ID, Q_i its available quantity, r_i its semantic or reference information, and v_i its visual reference.

The evidence required by PartGround is extracted directly from the assembly manual. Instruction, inventory, and reference regions are identified and associated with individual assembly steps, while candidate Part IDs, quantities, and reference locations are extracted to construct the candidate inventory. When visual extraction from a searchable PDF is unreliable, numeric text anchors are used as deterministic recovery cues. Qwen3-VL-4B-Instruct [12] then predicts the operation, action description, and semantic hardware requirements for each localized step. Numeric Part IDs predicted by the VLM are not accepted as final physical identities.

PartGround combines the semantic prediction with candidate references, manual evidence, and explicit hardware or quantity annotations when available. The selected physical identity is constrained by

$$\hat{c}_s \in \mathcal{C}, \tag{2}$$

so the final Part ID must correspond to a candidate extracted from the product-specific inventory. Agreement between manual callouts and candidate-reference evidence provides stronger support, whereas conflicting or incomplete evidence is retained explicitly rather than silently overwritten. The output of PartGround is

$$A_s^{\mathrm{PG}} = (\hat{c}_s, \hat{q}_s, \gamma_s, \mathcal{E}_s), \tag{3}$$

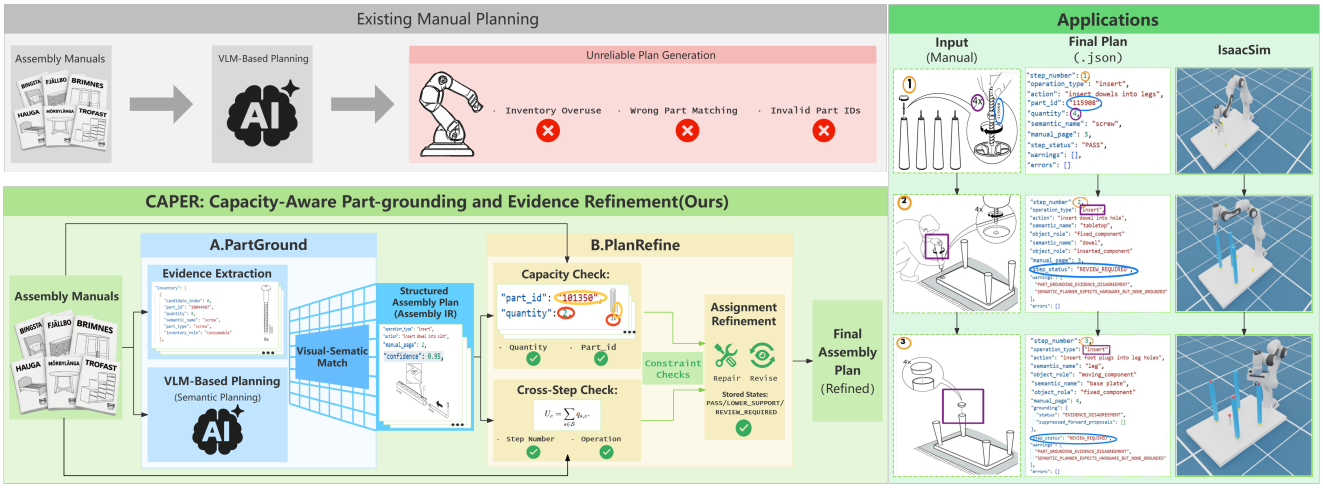

Fig. 1. **Overview of CAPER: Capacity-Aware Part-grounding and Evidence Refinement for manual-guided assembly planning.** **Top left:** Direct VLM-based planning may produce several problems. **Bottom left: (A) PartGround** combines manual evidence, inventory references, and semantic predictions to ground Part IDs and quantities in Assembly IR. **(B) PlanRefine** audits the grounded assignments, removes low-confidence assignments, and retains unresolved cases for review. **Right:** Example manual steps, structured JSON outputs, and corresponding assembly scenes in IsaacSim.

where $\hat{c}_s$ and $\hat{q}_s$ denote the grounded Part ID and quantity, γ_s denotes grounding support, and $\mathcal{E}_s$ records the supporting evidence.

The contribution of PartGround is therefore more than restricting outputs to legal identifiers. A legal Part ID can still correspond to the wrong physical component. By linking semantic hardware requirements with product-specific visual and inventory evidence, PartGround produces explicit and inspectable physical assignments that can be directly used by both PlanRefine and downstream robotic part retrieval and manipulation.

C. PlanRefine: Finite-Inventory Consistency Refinement

PartGround produces step-level physical assignments, but individually plausible assignments do not guarantee that the complete assembly plan is consistent with the available inventory. PlanRefine therefore operates on the complete set of grounded assignments and checks whether repeated use of each Part ID exceeds its available quantity. It does not perform additional Part-ID inference and does not search for alternative candidates.

For each Part ID c, let Q_c denote its available quantity and $q_{s,c}$ the quantity assigned to step s. The total predicted consumption is

$$U_c = \sum_{s \in S} q_{s,c}. \tag{4}$$

If $U_c \le Q_c$, all assignments associated with c are retained. If $U_c > Q_c$, PlanRefine orders the corresponding assignments by increasing grounding support, with later steps removed first when support values are equal. Let $\pi_c(1), \ldots, \pi_c(n_c)$ denote this ordering. PlanRefine removes the smallest number of assignments required to restore the inventory constraint:

$$k_c^* = \min \left\{ k : U_c - \sum_{i=1}^{k} q_{\pi_c(i),c} \le Q_c \right\}. \tag{5}$$

This refinement follows a minimal-intervention principle: only assignments associated with an overused Part ID are modified, and the process stops once the available capacity is restored. No new Part ID is invented, no replacement candidate is searched, and no combinatorial reassignment is performed. The resulting plan is re-audited against the same finite inventory before being emitted.

The refined assignments are stored in the final Assembly IR together with the semantic operation, action description, Part IDs, quantities, supporting evidence, and warnings. CAPER additionally retains three grounding-quality states: `PASS`, `LOWER SUPPORT`, and `REVIEW REQUIRED`, allowing uncertain assignments to remain explicit rather than being converted into confident predictions. PartGround and PlanRefine therefore provide complementary functions: PartGround determines which physical component belongs to an individual step, while PlanRefine verifies that these assignments remain jointly consistent across the complete assembly sequence.

D. Final Assembly Plan

The complete CAPER framework shown in Fig. 1 connects semantic instruction understanding with physically grounded and finite-inventory-consistent assembly planning. The semantic planner provides the operation and hardware requirement, PartGround resolves the corresponding product-specific Part IDs and quantities, and PlanRefine checks the resulting assignments over the complete assembly sequence. The final Assembly IR therefore combines semantic actions, exact physical parts, quantities, supporting evidence, and consistency information in a single structured representation.

As illustrated on the right side of Fig.1, this structured output provides an explicit interface between manual understanding and downstream robotic execution. Exact Part IDs identify the physical components to be localized and retrieved,

quantities specify the required number of parts, and operation information defines the intended assembly action. CAPER thus converts open-ended VLM reasoning into an evidence-grounded, product-specific, and finite-inventory-consistent assembly plan that can support subsequent perception, grasping, manipulation, and robotic furniture assembly.

IV. Experiments

A. Overview

We evaluate CAPER on the IKEA Assembly in the Wild (IAW) dataset [8] to examine whether evidence-guided part grounding and finite-inventory validation improve manual-guided assembly planning. The experiments compare CAPER with direct VLM generation and a same-backbone Manual2Skill baseline [5], followed by a cumulative component ablation. We distinguish local prediction quality from plan-level inventory consistency and retain pipeline non-completions in the evaluation denominator.

B. Experimental Setup

1) Data: We evaluate CAPER on the IAW test set [8], which contains 15 furniture instances and 297 ground-truth assembly steps. None of these furniture instances was used in our model training or adaptation, so all 15 instances are previously unseen by the models in our experimental setting. The pipeline produces outputs for 10 instances, covering 168 ground-truth steps. Three steps are omitted during localization, leaving 165 aligned steps. The remaining 5 instances and all missing steps are retained in the full 15-instance / 297-step evaluation denominator and counted as failures. This strict, failure-aware protocol evaluates performance across the entire test set rather than only on instances for which the pipeline completes.

2) Baselines: We compare CAPER against three controlled direct-generation baselines based on the same Qwen3-VL-4B-Instruct [12]: the original Base model, Supervised Fine-Tuning (SFT), and Direct Preference Optimization (DPO) [13].. These baselines use the same localized visual evidence, prompt and output schema, deterministic decoding protocol, and evaluation procedure, with only the model checkpoint changed; they directly generate the final structured report without PartGround or PlanRefine. We also include Manual2Skill [5] as a same-backbone controlled baseline while the remaining Manual2Skill planning procedure is kept unchanged and no CAPER-specific inventory grounding, PartGround, PlanRefine, or evaluator feedback is provided during inference. CAPER uses the same 4B backbone as a semantic planner and adds PartGround for exact Part-ID grounding and PlanRefine for finite-inventory consistency. Predictions from all methods are frozen before evaluation and scored using the same structured-report evaluation protocol. The numerical comparison with Manual2Skill therefore concerns this controlled planning baseline.

We conduct a cumulative component ablation using the complete CAPER framework, CAPER without PlanRefine, and the semantic planner without both PartGround and PlanRefine. The first comparison isolates the effect of finite-inventory repair, whereas the second additionally removes explicit Part-ID grounding. The semantic-planner-only configuration predicts operations and hardware mentions and is distinct from the Base direct-generation baseline. Because PlanRefine consumes grounded Part IDs, retaining PlanRefine while removing PartGround is not a well-defined independent ablation.

3) Metrics: We report Structured Report Precision, Recall, and F1, where the report is evaluated over operation facts and hardware facts defined by step-level Part ID and quantity. We additionally report Step Report Accuracy, Part-ID F1, and Hardware Exact Accuracy to measure local prediction quality. These metrics are evaluated under the full 15-instance / 297-step failure-aware protocol.

To evaluate whether the Part-ID assignments produced by PartGround and refined by PlanRefine remain consistent with the available parts across the complete assembly plan, we report Finite-inventory Consistency Rate and Complete-output Finite-inventory Consistency Rate on the common 12-instance subset with frozen and verifiable inventory references. Three instances are excluded from these two rates because they lack verifiable frozen inventory references. We also report Out-of-inventory Part-ID Occurrences to measure predictions unsupported by the supplied inventory. This setting allows us to evaluate local prediction accuracy on the full IAW test set [8] benchmark while separately examining whether the final assembly plans satisfy finite-inventory consistency.

C. Results

1) Assembly Planning Performance: Table II summarizes the primary comparison. CAPER obtains 62.99% Structured Report Precision, 30.77% Recall, and 41.34% F1, achieving the highest precision and F1 among the evaluated methods. Compared with Manual2Skill [5], CAPER increases precision from 27.08% to 62.99% and F1 from 30.09% to 41.34%, corresponding to improvements of 35.91 and 11.25 percentage points, respectively. Its Finite-inventory Consistency Rate increases from 41.67% to 83.33%, and its Complete-output Finite-inventory Consistency Rate increases from 33.33% to 58.33%, gains of 41.66 and 25.00 percentage points. Meanwhile, Out-of-inventory Part-ID Occurrences decrease from 118 to zero. These results show that CAPER improves both structured-report quality and inventory consistency over the controlled Manual2Skill baseline. Manual2Skill retains higher recall, at 33.85% versus 30.77%, so CAPER's advantage is concentrated in precision, F1, and inventory consistency.

Among the three direct Qwen3-VL-4B-Instruct baselines, DPO gives the highest Structured Report F1 at 40.91%, while SFT gives the highest precision at 62.20%. CAPER exceeds these values by 0.43 and 0.79 percentage points, respectively. Its larger advantage appears in finite-inventory consistency: CAPER reaches $10/12$ (83.33%), compared with $6/12$ (50.00%), $8/12$ (66.67%), and $6/12$ (50.00%) for Base, SFT, and DPO. The corresponding Complete-output Finite-inventory Consistency Rates are $7/12$ (58.33%) for CAPER,

TABLE II

ASSEMBLY PLANNING PERFORMANCE COMPARISON.

Method	Precision (%) ↑	Recall (%) ↑	F1 Score (%) ↑	Finite-inventory Consistency Rate (%) ↑	Complete-output Finite-inventory Consistency Rate (%) ↑	Out-of-inventory Part-ID Occurrences ↓
Qwen3-VL-4B-Instruct (Base)	60.07	30.96	40.86	50.00	25.00	1
Qwen3-VL-4B-Instruct + Supervised Fine-Tuning	62.20	30.38	40.83	66.67	41.67	2
Qwen3-VL-4B-Instruct + Direct Preference Optimization	60.30	30.96	40.91	50.00	33.33	2
Manual2Skill [5]	27.08	**33.85**	30.09	41.67	33.33	118
CAPER (Ours)	**62.99**	30.77	**41.34**	**83.33**	**58.33**	**0**

All metrics are evaluated on IAW test set, except for the **finite-inventory consistency rates**, which use a common 12-instance subset because three instances lack verifiable frozen inventory references. Compared with **Manual2Skill [5]**, **CAPER** improves the **F1 score** by 11.25 percentage points (41.34% vs. 30.09%), the **Finite-inventory Consistency Rate** by 41.66 percentage points (83.33% vs. 41.67%), and the **Complete-output Finite-inventory Consistency Rate** by 25.00 percentage points (58.33% vs. 33.33%), while reducing **Out-of-inventory Part-ID Occurrences** from 118 to zero.

compared with $3/12$ (25.00%), $5/12$ (41.67%), and $4/12$ (33.33%). CAPER also produces zero out-of-inventory Part-ID occurrences, compared with one, two, and two for these direct baselines. CAPER provides the strongest measured precision, F1, and inventory consistency in this comparison, with modest local accuracy gains over direct generation and larger gains over Manual2Skill [5].

2) Analysis of the Performance Gains: Compared with Manual2Skill [5] and Manual2Skill++ [7], CAPER focuses on the complementary requirements of linking each step to exact product-specific Part IDs and checking their accumulated use against finite inventory. The capability comparison in Table I contextualizes this distinction; the quantitative evidence in Table II is specific to the evaluated baselines, including Manual2Skill [5].

The semantic planner already captures much of the step-level assembly meaning from the manual. However, semantic understanding alone is insufficient for deciding which exact physical part should be used when multiple legal candidates are visually similar. PartGround addresses this gap by constraining hardware predictions to the product-specific candidate inventory and linking them with step-level manual evidence. Numeric Part IDs predicted by the VLM are not accepted as final physical identities. This evidence-guided grounding provides a mechanism for avoiding unsupported identifiers, consistent with CAPER's zero observed out-of-inventory Part-ID occurrences in Table II.

PlanRefine addresses a different source of error: assignments that appear plausible individually can exceed the available inventory when combined across steps. It audits the grounded assignments and removes low-confidence assignments implicated in overuse until finite-inventory consistency is restored. The shared Assembly IR connects semantic operations, exact Part IDs, quantities, and supporting evidence, allowing grounding and refinement to operate on the same assignments. This separation of responsibilities explains why CAPER's largest measured advantage over direct generation is in inventory consistency. Relative to direct structured generation and the controlled Manual2Skill procedure [5], CAPER's contribution is the explicit combination of evidence-guided physical assignment and cross-step capacity validation.

TABLE III

ABLATION STUDY OF CAPER.

Method	F1 Score (%) ↑	Step Report Accuracy (%) ↑	Part-ID F1 Score (%) ↑	Hardware Exact Accuracy (%) ↑	Finite-inventory Consistency Rate (%) ↑
CAPER (Ours)	**41.34**	**23.23**	**55.77**	**42.42**	**83.33**
w/o PlanRefine	41.18	**23.23**	55.24	41.75	58.33
w/o PartGround and PlanRefine	40.77	22.90	54.31	41.08	58.33

Table III examines these two contributions in turn.

D. Ablation Study

To address part-identity ambiguity and cross-step inventory inconsistency, our system comprises two complementary modules, PartGround and PlanRefine. We examine their effectiveness through the cumulative ablation in Table III, using the configurations defined in Section IV-B. The two modules are analyzed separately below to distinguish their effects on local grounding and plan-level inventory consistency.

1) PartGround: TableIII shows that adding PartGround to the semantic planner increases Part-ID F1 from 54.31% to 55.24% and Hardware Exact Accuracy from 41.08% to 41.75%, gains of 0.93 and 0.67 percentage points, respectively. Structured Report F1 increases from 40.77% to 41.18%, and Step Report Accuracy increases from 22.90% to 23.23%. Finite-inventory Consistency Rate remains unchanged at 58.33%. These results indicate that explicit physical grounding provides modest improvements in local identity and hardware prediction, while grounding alone does not improve the measured cross-step inventory consistency.

This pattern is consistent with the design of PartGround in Section III-B. For each step, the module forms an inventory-constrained proposal from the localized manual evidence and candidate references, while also searching for explicit hardware callouts and quantity annotations. Agreement between these evidence sources provides stronger support; forward-only evidence is retained with lower support, and disagreement is not silently overwritten. These mechanisms connect generic hardware mentions to specific physical candidates while preserving evidence for review. This is consistent with the smaller gain in overall Structured Report

F1, which also reflects operation facts supplied by the semantic planner.

These results show PartGround's contribution to exact physical assignment and hardware-level prediction. Its distinctive role is to transform free-form hardware prediction into evidence-guided, product-specific Part IDs and quantity assignments within the Assembly IR. This extends structured manual interpretation, as represented by Manual2Skill [5], with an explicit grounding stage, although discrimination among visually similar legal candidates remains difficult.

2) PlanRefine: Adding PlanRefine to the PartGround configuration increases Finite-inventory Consistency Rate from 7/12 (58.33%) to 10/12 (83.33%), a gain of 25.00 percentage points in TableIII. In comparison, Structured Report F1 changes from 41.18% to 41.34%, and Part-ID F1 increases from 55.24% to 55.77%. Hardware Exact Accuracy rises from 41.75% to 42.42%, while Step Report Accuracy remains unchanged at 23.23%. Thus, the primary demonstrated effect of PlanRefine is improved finite-inventory consistency, accompanied by small changes in local prediction quality.

This outcome follows from the capacity-repair mechanism in Section III-C. PlanRefine aggregates each Part ID's predicted consumption across the complete assembly sequence and compares it with the available quantity. When consumption exceeds inventory, it removes assignments in increasing order of grounding confidence, breaking ties by removing later-step assignments first, and stops once capacity is restored. It then re-audits the repaired plan against the same inventory. PlanRefine does not infer replacement Part IDs or search alternative candidates, so its main effect is to resolve cross-step over-allocation rather than improve local identity discrimination.

PlanRefine substantially improves the measured inventory consistency while preserving the aggregate level of local prediction accuracy. Its contribution is an explicit, confidence-guided capacity repair applied to grounded assignments, complementing PartGround's evidence-guided identity selection. Together, the two modules extend step-level structured prediction with both physical grounding and finite-inventory validation, addressing reliability requirements that are not ensured by the controlled direct-generation and Manual2Skill baselines [5].

E. Robotic Assembly Experiments

To investigate the downstream impact of part grounding on robotic execution, we conducted furniture assembly experiments in NVIDIA IsaacSim [14] using a Franka Emika Panda manipulator [15].The simulation experiments were run on a server equipped with four NVIDIA GeForce RTX 5090 GPUs and covered five furniture instances. These experiments examine whether identifying the exact parts required by an instruction step improves the robot's ability to select the intended components and carry out the corresponding assembly operation.

We compared assembly execution with and without explicit Part-ID grounding. Without grounding, we observed incorrect part selection and failures to initiate required grasps when the intended component was ambiguous. With CAPER, the execution system received explicit Part-ID and quantity assignments through the Assembly IR. The success rate in simulation improved significantly.

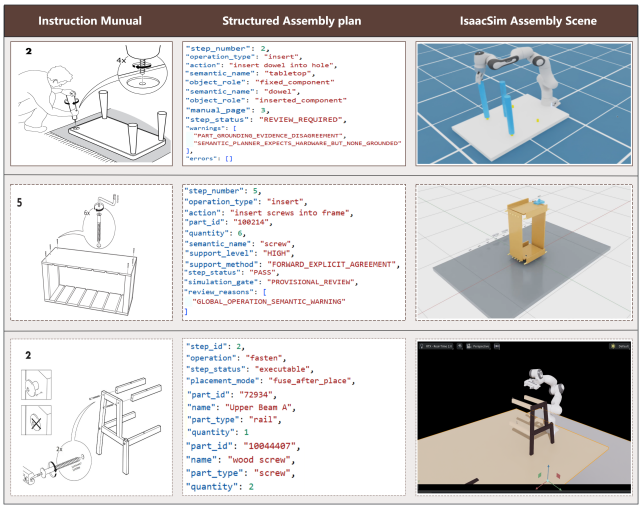

Fig. 2. **Representative furniture assembly examples.**

From top to bottom, the rows show MAMMUT (Step 2), TROFAST (Step 5), and BEKVÄM (Step 2).
Left: assembly illustrations from the instruction manuals. Middle: corresponding structured assembly plan excerpts, including assembly operations, part information, hardware quantities, and execution or review status. Right: corresponding assembly scenes in NVIDIA IsaacSim.

Across the five evaluated furniture instances, execution with CAPER exhibited improved part-selection accuracy and more reliable assembly-step completion compared with execution without explicit Part-ID grounding. The observed improvement illustrates the practical value of resolving part identities before robot execution: selecting the correct component is a prerequisite for performing the intended assembly operation. These findings support the practical value of CAPER's part grounding for improving part selection and assembly execution reliability in the evaluated simulation tasks.

Fig. 2 shows three representative steps (MAMMUT Step 2, TROFAST Step 5, and BEKVÄM Step 2). From left to right, each row aligns the instruction-manual illustration, the structured Assembly IR excerpt (operation, Part IDs, quantities, and PASS or REVIEW REQUIRED status), and the corresponding NVIDIA IsaacSim scene. These views make the mapping from manual evidence to physical assignment and simulated execution inspectable.

V. Discussion and Limitations

The current IAW test set [8] results show that CAPER improves both local prediction quality and plan-level reliability. CAPER achieves a Structured Report F1 of 41.34%, compared with 40.91% for the strongest direct Qwen3-VL-4B-Instruct baseline, while reaching an 83.33% Finite-inventory Consistency Rate and a 58.33% Complete-output Finite-inventory Consistency Rate with zero observed inventory

violations. These gains reflect the complementary roles of the two modules: PartGround improves evidence-guided physical grounding, whereas PlanRefine strengthens cross-step finite-inventory consistency through minimal refinement of conflicting assignments.

Furthermore, the full CAPER system incurs considerable computational overhead originating from layout parsing, multi-region visual-evidence matching and VLM inference, rather than an exact combinatorial inventory-assignment solver.

VI. Conclusion

We presented CAPER, a manual-guided robotic assembly framework for Part-ID grounding and finite-inventory consistency. The VLM is used only as a semantic planner. PartGround converts step-level manual evidence into inventory-constrained Part IDs and quantity assignments, while PlanRefine audits those assignments against finite inventory without re-inferring identities. Both write to a shared Assembly IR, which retains review states when the evidence is insufficient.

On the IAW test set [8], the gain is generally above peer level. CAPER reaches a Finite-inventory Consistency Rate of 83.33% and produces zero out-of-inventory Part-ID occurrences, while Structured Report F1 remains above the strongest direct Qwen3-VL-4B-Instruct baseline. PartGround turns free-form hardware mentions into inventory-legal Part-ID assignments; PlanRefine then raises inventory consistency from 58.33% to 83.33% and removes all observed overuse pairs ($9 \rightarrow 0$) and excess units ($51 \rightarrow 0$).

The Assembly IR is the interface to execution. It aligns manual instructions with exact Part IDs, quantities, and a PASS or REVIEW REQUIRED state for the corresponding IsaacSim scene. Thus,CAPER turns VLM-generated assembly knowledge into inventory-consistent physical targets for downstream perception, grasping, and execution.

Acknowledgment

Generative AI (OpenAI ChatGPT) was used for initial drafting, language organization, and schematic figure ideation. All technical claims, experimental results, code, figures, and references remain the authors' responsibility.

References

[1] I. Sera, N. Yamanobe, I. G. Ramirez-Alpizar, Z. Wang, W. Wan, and K. Harada, "Assembly planning by recognizing a graphical instruction manual," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 3138–3145.

[2] B. Ichter, A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, D. Kalashnikov, S. Levine, Y. Lu, C. Parada, K. Rao, P. Sermanet, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, M. Yan, N. Brown, M. Ahn, O. Cortes, N. Sievers, C. Tan, S. Xu, D. Reyes, J. Rettinghouse, J. Quiambao, P. Pastor, L. Luu, K.-H. Lee, Y. Kuang, S. Jesmonth, N. J. Joshi, K. Jeffrey, R. J. Ruano, J. Hsu, K. Gopalakrishnan, B. David, A. Zeng, and C. K. Fu, "Do as i can, not as i say: Grounding language in robotic affordances," in *Proceedings of the 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 205. PMLR, 2023, pp. 287–318.

[3] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 11 523–11 530.

[4] A. Z. Ren, A. Dixit, A. Bodrova, S. Singh, S. Tu, N. Brown, P. Xu, L. Takayama, F. Xia, J. Varley, Z. Xu, D. Sadigh, A. Zeng, and A. Majumdar, "Robots that ask for help: Uncertainty alignment for large language model planners," in *Proceedings of the 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 229. PMLR, 2023, pp. 661–682.

[5] C. Tie, S. Sun, J. Zhu, Y. Liu, J. Guo, Y. Hu, H. Chen, J. Chen, R. Wu, and L. Shao, "Manual2skill: Learning to read manuals and acquire robotic skills for furniture assembly using vision-language models," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025.

[6] Z. Yang, C. Garrett, D. Fox, T. Lozano-Pérez, and L. P. Kaelbling, "Guiding long-horizon task and motion planning with vision language models," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 16 847–16 853.

[7] C. Tie, S. Sun, Y. Lin, Y. Wang, Z. Li, Z. Zhong, J. Zhu, Y. Pang, H. Chen, J. Chen, R. Wu, and L. Shao, "Manual2skill++: Connector-aware general robotic assembly from instruction manuals via vision-language models," in *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026, arXiv:2510.16344.

[8] J. Zhang, A. Cherian, Y. Liu, Y. Ben-Shabat, C. Rodriguez, and S. Gould, "Aligning step-by-step instructional diagrams to video demonstrations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 2483–2492.

[9] J. Ao, F. Wu, Y. Wu, A. Swikir, and S. Haddadin, "Llm-as-bt-planner: Leveraging llms for behavior tree generation in robot task planning," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 1233–1239.

[10] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, T. Jackson, N. Brown, L. Luu, S. Levine, K. Hausman, and B. Ichter, "Inner monologue: Embodied reasoning through planning with language models," in *Proceedings of the 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 205. PMLR, 2023, pp. 1769–1782.

[11] Y. Shao, Y. Zheng, S. Sun, P. Chaudhari, V. Kumar, and N. Figueroa, "Symskill: Symbol and skill co-invention for data-efficient and reactive long-horizon manipulation," in *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026, arXiv:2510.01661.

[12] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, W. Ge, Z. Guo, Q. Huang, J. Huang, F. Huang, B. Hui, S. Jiang, Z. Li, M. Li, M. Li, K. Li, Z. Lin, J. Lin, X. Liu, J. Liu, C. Liu, Y. Liu, D. Liu, S. Liu, D. Lu, R. Luo, C. Lv, R. Men, L. Meng, X. Ren, X. Ren, S. Song, Y. Sun, J. Tang, J. Tu, J. Wan, P. Wang, P. Wang, Q. Wang, Y. Wang, T. Xie, Y. Xu, H. Xu, J. Xu, Z. Yang, M. Yang, J. Yang, A. Yang, B. Yu, F. Zhang, H. Zhang, X. Zhang, B. Zheng, H. Zhong, J. Zhou, F. Zhou, J. Zhou, Y. Zhu, and K. Zhu, "Qwen3-vl technical report," *arXiv preprint arXiv:2511.21631*, 2025.

[13] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[14] NVIDIA, "Isaac Sim," https://github.com/isaac-sim/IsaacSim, version 6.0.1, accessed Sep. 16, 2026.

[15] S. Haddadin, S. Parusel, L. Johannsmeier, S. Golz, S. Gabl, F. Walch, M. Sabaghian, C. Jähne, L. Hausperger, and S. Haddadin, "The franka emika robot: A reference platform for robotics research and education," *IEEE Robotics & Automation Magazine*, vol. 29, no. 2, pp. 46–64, 2022.