Improvisational Reasoning with Vision-Language Models for Grounded Procedural Planning

Md Masudur Rahman, Yupeng Zhuo, and Juan P. Wachs
Edwardson School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA
rahman64@purdue.edu, zhuoy@purdue.edu, jpwachs@purdue.edu

Abstract

Improvisation is a hallmark of human intelligence, particularly in high-stakes domains such as emergency medicine, where ideal tools are often unavailable and practitioners must adapt procedures using what is at hand. While recent vision-language models (VLMs) have demonstrated strong general reasoning and perception abilities, they remain inadequate for grounded procedural adaptation under constraints. In this paper, we introduce ImPlan, an improvisational reasoning framework that augments VLMs with structured planning and transformation-aware substitution. ImPlan generates action-object graphs that adapt procedural goals to context-specific affordances in the scene. Experiments on a benchmark of expert-annotated emergency procedures show that ImPlan significantly outperforms direct VLM prompting, both proprietary and open-weight models, even when built on weaker backbone models. On average, ImPlan improves groundness scores by up to 70.8% and plausibility scores by up to 28.6%, achieving simultaneous gains in visual grounding and logical coherence. ImPlan offers a potentially generalizable path for grounded decision-making in resource-limited environments.

1 Introduction

Vision-language models (VLMs) have achieved significant success by integrating visual processing with natural language understanding [Achiam et al., 2023, Touvron et al., 2023, Liu et al., 2023, 2024, Radford et al., 2019, 2021, Li et al., 2023, Zhang et al., 2024b, Li et al., 2024, Guo et al., 2024, Zhang et al., 2024a, Shakeri et al., 2024]. While VLMs have demonstrated impressive performance on perception-grounded tasks and general instruction following [Cheng et al., 2025, Dai et al., 2023], they largely operate within a traditional reasoning paradigm [Wei et al., 2022, Wang et al., 2023]. These models are adept at following detailed prompts or retrieving known procedures, but they lack mechanisms for context-sensitive procedural adaptation. These models are adept at following detailed prompts or retrieving known procedures, but they lack mechanisms for context-sensitive procedural adaptation [Amara et al., 2024, Nikandrou et al., 2024].

Specifically, when key tools are missing or altered, VLMs tend to either hallucinate inappropriate actions or default to generic templates-failing to produce grounded, functional adaptations[Chen et al., 2024, Qian et al., 2024]. Consider a first response scenario where a field medic must perform an emergency tracheostomy to restore a patient's airway. The ideal tool, a surgical scalpel, is missing. In a high-stakes, time-critical setting, the medic does not pause the procedure—instead, they sanitize a sharp knife and proceed. This act is not a reckless guess; it is a calculated decision that substitutes an available tool for a missing one, while preserving the goal and structure of the procedure. Figure 1 illustrates an improvisation scenario for emergency tourniquet application. Such decisions exemplify *improvisational reasoning*: the ability to adapt abstract procedural knowledge to grounded, real-world constraints [Favero et al., 2024, Lyu et al., 2024].

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: The Second Workshop on GenAI for Health: Potential, Trust, and Policy Compliance.

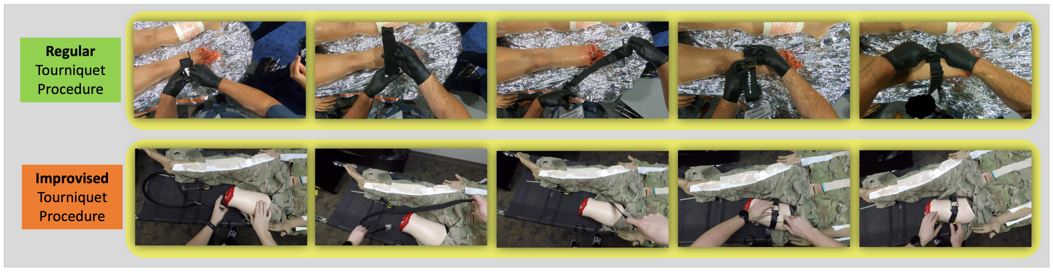

Figure 1: Comparison of Regular and Improvised Tourniquet Procedures: The **top** row demonstrates a **standard tourniquet** application using medical-grade equipment in a controlled setting. The bottom row depicts an improvised (properly referred in the medical jargon as Just-in-Time (JIT)) tourniquet procedure using a **belt and a screwdriver**, as might be required in austere environments. This study evaluates the **ability of vision-language models (VLMs)** to generate step-by-step medical procedures in **improvised settings** based on visual scene understanding.

Traditional reasoning, as modeled in current VLMs and planning systems, aims for optimality under the assumption that ideal resources are available. Improvisational reasoning, by contrast, requires systems to identify feasible substitutes, anticipate their transformed use, and assess whether the modified plan remains safe and effective. A model that knows the correct surgical procedure is not, by default, capable of adapting it in a degraded environment.

In this work, we introduce **ImPlan**, a structured framework for improvisational reasoning grounded in visual context. Given a scene and a procedural goal (e.g., tracheostomy, tourniquet application), ImPlan generates an adapted sequence of action-object steps that maintain procedural intent while replacing unavailable tools with visually present alternatives. Our system leverages a pre-trained VLM, but enhances it with explicit graph-based procedural reasoning and transformation-aware substitutions.

We evaluate ImPlan on a new benchmark of five emergency medical procedures, annotated by domain experts performing real or simulated improvisations. Experiments show that ImPlan significantly outperforms direct VLM prompting across multiple open-weight and proprietary models of varying sizes and capabilities. On average, ImPlan improves groundness by up to **70.8%** and plausibility by up to **28.6%**, showing consistent bidirectional gains across all tested models. These results demonstrate that modeling grounded procedural improvisation is critical for deploying AI systems in high-stakes, resource-limited settings.

2 Problem Settings

We address the task of procedure graph generation in the context of emergency medical interventions, where the execution of a procedure must be adapted to real-world constraints such as limited tool availability and the urgency of immediate care. In such high-stakes scenarios, such as applying a tourniquet or clearing an airway, medical practitioners are often required to look for alternatives, relying on non-standard or makeshift tools to perform critical steps (referred to as Just-in-Time (JIT) procedures).

The goal of this task is to generate a coherent, contextually grounded sequence of medical actions that achieves the intended procedural objective using only the resources available in the environment. In practice, these procedure steps assist a general medic, who may be unfamiliar with the specific situation and likely operating under stress, in carrying out the emergency intervention. Task alteration in such scenarios is challenging, even for experts. While the underlying concept is broadly applicable to many other domains, in this paper, we focus specifically on emergency medical procedures.

Formally, let S denote a scene, represented as a short video clip or a sequence of image frames capturing a localized emergency situation. Let $C \in \mathcal{P}$ be the context label specifying the high-level medical procedure to be performed, where $\mathcal{P}$ is the set of supported procedural categories. The desired output is a procedure graph G_{out}, defined as an ordered sequence of action-object pairs:

$$G_{\text{out}} = \{(a_1, o_1), (a_2, o_2), \ldots, (a_n, o_n)\},$$

where each a_i is a discrete action (e.g., *tie*, *wrap*, *apply pressure*) and each o_i is a physical object or tool (e.g., gauze, towel, scarf) used to perform the action.

The procedure graph G_{out} must satisfy two primary constraints. First, the sequence must be internally coherent and goal-directed, preserving the logical structure of a standard execution of procedure C. Second, the graph must be grounded in the visual content of scene S, such that each object o_i is either visibly present in the scene or plausibly substitutable based on semantic or functional similarity. The ability to generate such grounded and improvisational procedure graphs is critical for intelligent assistance systems in emergency medicine, where time constraints and environmental variability prohibit reliance on ideal conditions or standard equipment.

This formulation extends classical procedural modeling by incorporating both contextual grounding and improvisational reasoning, enabling robust adaptation of procedural knowledge in highly dynamic and resource-constrained settings.

3 Methodology

Improvisation may require modifying a scene-available object before it becomes a functional substitute. For instance, converting a pen into an airway tube requires removing the ink cartridge. Let $\mathcal{T}$ denote the set of allowed transformations (e.g., *cut*, *unwrap*, *flatten*), where each $t \in \mathcal{T}$ maps a raw object to a transformed one. We define an *improvisation score* function $I(o^*, o') \in [0, 1]$ to quantify the functional similarity between the required object o^* and a transformed substitute o'. A substitution is considered feasible only if the score exceeds a minimum threshold $\tau \in [0, 1]$. This ensures that highly implausible actions, those likely to disrupt the procedure or worsen the situation, are avoided. Figure 2 general overview of our framework. The details process is described bellow.

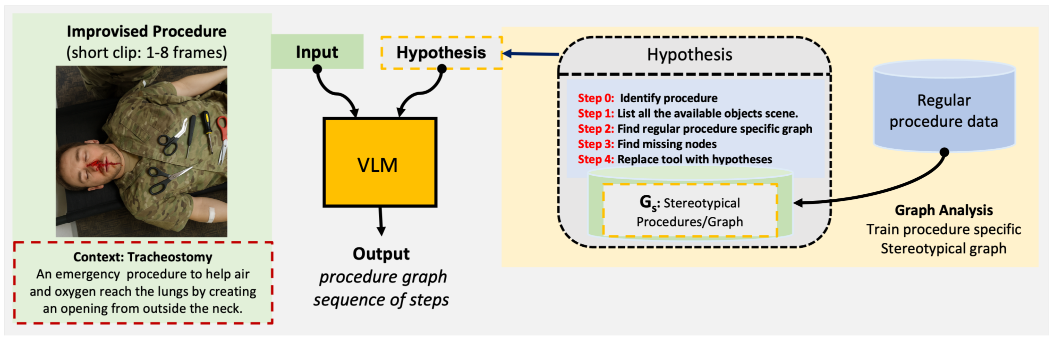
Improvised Procedure
(short clip: 1-8 frames)
Context: Tracheostomy
An emergency procedure to help air and oxygen reach the lungs by creating an opening from outside the neck.
Input
Hypothesis
VLM
Output
procedure graph
sequence of steps
Hypothesis
Step 0: Identify procedure
Step 1: List all the available objects scene.
Step 2: Find regular procedure specific graph
Step 3: Find missing nodes
Step 4: Replace tool with hypotheses
G_s: Stereotypical Procedures/Graph
Regular procedure data
Graph Analysis
Train procedure specific Stereotypical graph

Figure 2: Overview of ImPlan Framework.

3.1 Procedural Graph Adaptation Setup

Let $C \in \mathcal{P}$ be a procedural context label, and let $G^C_{\text{ideal}} = \{(a_1, o^*_1), \ldots, (a_n, o^*_n)\}$ denote the canonical procedure graph for task C, where each a_i is an action and o^*_i is the ideal tool used to perform it. Let S represent a visual scene, and let $T_S \subseteq \mathcal{O}$ be the set of available objects in the scene.

The objective is to generate a modified graph $G_{\text{imp}} = \{(a_i, \tilde{o}_i, \tilde{t}_i)\}$, where each $\tilde{o}_i \in T_S$ and $\tilde{t}_i \in \mathcal{T}$ is a transformation (or the identity function), such that the resulting sequence maintains the procedural logic of G^C_{ideal}, while adapting to the object constraints imposed by S.

3.2 Stereotypical Procedure Graph Construction

To ground improvisational reasoning in clinically validated knowledge, ImPlan constructs a **stereotypical procedure graph** G^* for each procedural goal. Each graph captures the canonical sequence of actions and objects observed during regular (non-improvised) executions of the procedure. Each instance of a procedure is represented as a directed graph $G = (V, E)$, where V denotes the set of nodes, corresponding to actions (verbs) and objects (nouns), and $E \subseteq V \times V$ encodes the directed edges representing transitions between steps. Each edge $e = (v_i, v_j) \in E$ is annotated with an ordinal label indicating the chronological step order. An action at step i is described as a verb-noun pair (u_i, v_i), representing the operation and the associated object.

Each procedure instance is encoded into two matrices: an adjacency matrix $P \in \{0,1\}^{N_W \times N_W}$ representing the structural connectivity, and an ordinal matrix $Q \in \mathbb{R}^{N_W \times N_W}$ capturing the temporal ordering of steps, where N_W denotes the total number of unique verb and noun nodes across all instances. To ensure uniqueness, repeated actions within a procedure are assigned distinct counter indices.

The construction of a procedural graph proceeds by parsing action sequences sequentially: nodes are added to the graph as new actions or objects appear, and edges are created linking object-verb-object transitions, preserving temporal order. The adjacency and ordinal matrices are updated accordingly throughout the parsing process. Further details of the graph generation procedure are provided in Appendix.

Given a set of K demonstration instances $\{G_1, G_2, \ldots, G_K\}$ corresponding to the same procedure, the **stereotypical procedure graph** G^* is constructed by aggregating adjacency and ordinal matrices across instances. For each node pair (i, j), the averaged adjacency $\bar{P}(i,j)$ and ordinal $\bar{Q}(i,j)$ are computed as: $\bar{P}(i,j) = \frac{1}{K}\sum_{k=1}^{K} P_k(i,j), \quad \bar{Q}(i,j) = \frac{1}{K}\sum_{k=1}^{K} Q_k(i,j)$. To filter out rare or inconsistent transitions, a threshold $\alpha \in [0,1]$ is applied: transitions with $\bar{P}(i,j) < \alpha$ are pruned, while others are retained. The remaining edges define the final structural skeleton of G^*, while the averaged ordinal values preserve procedural sequencing information. The detailed averaging procedure is provided in Appendix.

Each procedural goal is associated with exactly one pre-constructed graph G^*, creating a direct one-to-one mapping used at inference time. By grounding adaptations in these stereotypical structures, ImPlan preserves procedural coherence while allowing flexible improvisation. Detailed steps are in the Algorithm 1.

3.3 Hypothesis Generation via Reasoning Steps

Given the scene toolset T_S, transformation set $\mathcal{T}$, the ideal graph G^C_{ideal}, and the score function I, the algorithm proceeds as follows:

Algorithm 1 ImPlan Algorithm

```
Require: Ideal concept graph $G^C_{\text{ideal}}$, object set $T_S$, transformations $\mathcal{T}$, similarity threshold $\tau$
Ensure: Improvised graph $G_{\text{imp}}$
 1: Initialize improvised graph $G_{\text{imp}} \leftarrow [\,]$
 2: for each $(a_i, o_i^*) \in G^C_{\text{ideal}}$ do
 3:     if $o_i^* \in T_S$ then
 4:         Append $(a_i, o_i^*, \text{identity})$ to $G_{\text{imp}}$
 5:     else
 6:         for each $(o, t) \in T_S \times \mathcal{T}$ do
 7:             $o' \leftarrow \text{Transform}(o, t)$
 8:             $s \leftarrow I(o_i^*, o')$
 9:         end for
10:         Select best-scoring pair $(\hat{o}, \hat{t})$ such that $I(o_i^*, \text{Transform}(\hat{o}, \hat{t})) \geq \tau$
11:         if such a pair $(\hat{o}, \hat{t})$ exists then
12:             Append $(a_i, \hat{o}, \hat{t})$ to $G_{\text{imp}}$
13:         end if
14:     end if
15: end for
16: return $G_{\text{imp}}$
```

If no feasible graph generated the algorithm return an empty set. To adapt the ideal graph to a scene S with observed tools T_S, the system follows a structured reasoning path based on the algorithm which yields a grounded hypothesis graph $G_h = \{(a_i, \tilde{o}_i, \tilde{t}_i)\}$:

Hypothesis and Reasoning Process

- **Procedure identification:** The label C defines the target procedural intent.
- **Tool detection:** Visual processing identifies objects $T_S \subseteq \mathcal{O}$ available in scene S.
- **Graph grounding:** Each step in G^C_{ideal} is compared against T_S to determine tool availability.
- **Tool substitution:** For missing tools o_i^*, the system queries a set of transformations $\mathcal{T}$ over each $o \in T_S$ and computes a similarity score
  $\hat{I}(o_i^*, \text{Transform}(o, t))$, where $\hat{I}$ is approximated using VLM embedding similarity.
- **Substitution selection:** The best substitute $(\hat{o}_i, \hat{t}_i)$ is selected for each missing tool, provided the estimated similarity exceeds threshold τ; otherwise, the step is omitted.

3.4 Practical Instantiation via Vision-Language Models

Since the improvisation score is not directly accessible, it is approximated using a pre-trained vision-language model (VLM) to estimate semantic similarity between ideal and transformed tools within a shared embedding space. Transformations are derived from scene context or prompt engineering, with a greedy substitution policy applied based on similarity scores. This allows efficient, scalable reasoning under perceptual constraints. To connect design and deployment, the algorithm and reasoning framework (Fig. 2) guide VLM behavior through structured prompts. Originally a conceptual model of procedural improvisation, these reasoning steps are encoded as system instructions, aligning inference-time behavior with high-level planning logic for greater interpretability and consistency.

3.5 Implementation Details

The system is a structured reasoning pipeline built on a pre-trained vision-language model (VLM), operating in a zero-shot, prompt-based setting without fine-tuning. Prompt engineering, input formatting, and retrieval are used to ground the procedural graph and guide reasoning. Each VLM query is framed with a system prompt encoding the scene, the high-level goal C, and the hypothesized action sequence. These prompts help the model interpret the scene, reason through steps, and suggest substitutions. The ideal procedure graph G^C_{ideal} is retrieved as a list of action-object pairs, presented as bullet points or phrases. Steps are represented both in text (e.g., “insert airway tube”) and via embeddings. Scene objects $o \in T_S$ are labeled with natural phrases (e.g., “pen”, “rolled towel”) and transformed using templates (e.g., “cut-open pen”). These variants are encoded using the VLM’s text encoder.

Inference uses API-based VLM access via chat interfaces (OpenAI, Gemini, Together AI[1]), with support for models like LLaMA and Qwen.

4 Experiments

We evaluate **ImPlan**, our improvisational reasoning framework, on a suite of procedural tasks requiring grounded planning under object constraints. These tasks simulate real-world scenarios in which ideal tools may be unavailable, necessitating adaptation using only the resources visually present in the scene. The experiments test both full-procedure adaptation and localized, step-level improvisation across a diverse set of vision-language models (VLMs) and reasoning strategies.

4.1 Dataset

All videos were recorded from a first-person perspective using head-mounted GoPro Hero7 cameras (San Mateo, California) at 1080p resolution. The cameras were angled 20–30 degrees downward from the forehead to ensure optimal framing, with the hands intentionally centered in the field of view to enhance procedural visibility. Recordings were conducted across a range of simulated clinical

[1] `https://www.together.ai/`

environments to reflect operational diversity. These recordings capture **five** life-saving interventions: Cricothyroidotomy (**CR**), Needle Thoracostomy (**ND**), Tourniquet (**TQ**), Tube Thoracostomy (**CT**), and Interosseous Insertion (**IO**).

A key component of the dataset includes **67 videos** capturing “just-in-time” (JIT) procedures—**improvised** life-saving interventions performed using non-standard or readily available materials. These scenarios emphasize adaptability in resource-limited settings, including the use of belts (e.g., Figure 1) or clothing secured with screwdrivers for tourniquets.

These **improvised videos serve as the primary evaluation** set for this paper, highlighting real-world improvisation and dynamic decision-making under pressure. In contrast, the dataset also contains **220 videos** of standard procedures performed with conventional medical tools in more controlled conditions. These represent routine, stereotypical workflows and are used to establish reference performance and baseline behavior in procedural execution. The standard procedure data were used to generate the stereotypical procedure graph for our ImPlan framework.

Each video in the dataset was annotated by trained medical professionals, who labeled start and end timestamps for each action and described the activity using verb–noun pairs (e.g., “insert needle,” “apply tourniquet”). These annotations serve as the ground truth for model training and evaluation. To ensure accuracy and reduce inconsistencies, all annotations underwent peer review by additional medical experts, supporting high-quality ground truth.

4.2 Baselines and VLMs

We compare ImPlan against several configurations that isolate different reasoning mechanisms. The direct prompting baseline uses the VLM to generate responses without structured guidance, simulating a zero-shot setting. A second baseline augments the prompt with a chain-of-thought (CoT) scaffold, encouraging sequential reasoning. The full ImPlan system introduces structured graph reasoning, tool transformation modeling, similarity-based substitution, and scoring components.

Experiments are conducted using both open-weight and proprietary VLMs. The open-weight models include variants from LLaMA, and Qwen. Proprietary models include GPT-4.1 (Mini, Nano), GPT-4o, and Gemini (2, 1.5). This range allows us to assess how model capacity and alignment quality influence improvisational performance.

4.3 Evaluation Metrics

Performance is measured using multiple complementary metrics. The **plausibility score** quantifies the logical consistency and goal alignment of generated action sequences. It is computed using cosine similarity in an embedding space, where embeddings are obtained from a sentence transformer [Reimers and Gurevych, 2019] applied to the entire procedure graph represented as text. The score ranges from 0 to 1 (higher is better). The **groundness score** evaluates the visual and contextual feasibility of each substituted tool within the scene. This metric assesses the ability of VLMs to generate plans that are grounded in the provided scene context, rather than relying on memorized standard procedures. It is calculated by comparing the ground truth annotations, which include improvised tool use, with the predicted or generated plans produced by the VLMs. A higher score indicates better grounding, and values are normalized between 0 and 1.

5 Results

5.1 Groundness Score

The **groundness score** evaluates the extent to which generated tool-use procedures are contextually and visually grounded in the provided scene. This metric is critical for assessing the ability of models to reason beyond memorized routines and adapt to dynamic, scene-specific constraints. Table 1 shows that our proposed method, **ImPlan**, consistently improves groundness across all evaluated models, demonstrating its effectiveness in enhancing scene-aware planning.

Strong Improvements Across All Model Types. Our ImPlan approach demonstrates gains across all ten models, with particularly large relative improvements for open-weight models. **Qwen2.5-72B**, for example, shows the highest relative improvement of **+70.80%**, boosting its average score from 0.27 to

Table 1: Comparison of model **Groundness Score** across different settings. AVG = average of CR, CT, IO, ND, TQ; Gain (%) = percentage improvement from base to implan.

Model	Setting	CR	CT	IO	ND	TQ	AVG	Gain (%)
GPT-4.1	base	0.40	0.61	0.46	0.69	0.32	0.50	
	implan	0.52	0.62	0.78	0.49	0.68	**0.62**	**24.60**
GPT-4.1-mini	base	0.31	0.35	0.52	0.62	0.24	0.41	
	implan	0.36	0.49	0.70	0.58	0.60	**0.55**	**33.82**
GPT-4.1-nano	base	0.11	0.33	0.29	0.56	0.07	0.27	
	implan	0.27	0.37	0.38	0.40	0.33	**0.35**	**28.68**
GPT-4o	base	0.34	0.36	0.60	0.72	0.31	0.47	
	implan	0.52	0.53	0.41	0.46	0.61	**0.51**	**8.58**
Gemini 1.5	base	0.20	0.35	0.25	0.48	0.21	0.30	
	implan	0.26	0.45	0.35	0.51	0.58	**0.43**	**44.30**
Gemini 2.0	base	0.36	0.42	0.60	0.68	0.30	0.47	
	implan	0.44	0.64	0.59	0.89	0.37	**0.59**	**24.15**
LLaMA3.2-11B	base	0.12	0.21	0.34	0.52	0.09	0.26	
	implan	0.30	0.32	0.57	0.57	0.15	**0.38**	**49.22**
LLaMA4-Mav-17B	base	0.20	0.45	0.27	0.65	0.22	0.36	
	implan	0.37	0.51	0.36	0.55	0.30	**0.42**	**16.76**
LLaMA4-Scout-17B	base	0.19	0.30	0.30	0.40	0.19	0.28	
	implan	0.33	0.40	0.29	0.64	0.53	**0.44**	**58.70**
Qwen2.5-72B	base	0.16	0.35	0.30	0.39	0.17	0.27	
	implan	0.44	0.39	0.51	0.69	0.31	**0.47**	**70.80**

0.47. Similarly, **LLaMA4-Scout-17B** improves from 0.28 to 0.44 (**+58.70%**), and **LLaMA3.2-11B** gains **+49.22%**. These results underscore ImPlan's utility for lifting the performance floor of weaker base models in terms of contextual understanding.

ImPlan scales effectively across models ranging from compact versions (e.g., **GPT-4.1-nano**, **LLaMA3.2-11B**) to large-scale architectures (**Qwen2.5-72B**, **LLaMA4-Mav-17B**). Importantly, the magnitude of improvement does not correlate linearly with base performance: weaker models often show the highest relative gains, but even high-performing base models see meaningful improvements.

Gains for Proprietary Models. Proprietary models, including **GPT-4.1**, **GPT-4o**, and **Gemini**, also benefit notably. **GPT-4.1**, already a strong performer (0.50 base), sees a **+24.60%** gain, reaching an average groundness of **0.62**. **Gemini 1.5** and **Gemini 2.0** experience improvements of **+44.30%** and **+24.15%**, respectively. Notably, **GPT-4.1-mini** exhibits a substantial **+33.82%** increase, suggesting that ImPlan is especially beneficial for lower-capacity variants that may lack innate scene-specific reasoning skills.

Category-Level Trends. ImPlan's strongest effects are observed in **IO** and **TQ**—categories that require flexible adaptation to non-standard tools and task conditions. For instance, **GPT-4.1** improves from 0.46 to 0.78 in IO, and **Gemini 1.5** jumps from 0.21 to 0.58 in TQ. These improvements suggest that ImPlan enables models to break from rigid, canonical patterns and propose more plausible alternatives suited to the actual visual context.

While gains are generally consistent, a few categories show flat or slightly reduced scores post-ImPlan. For example, **GPT-4.1** drops in ND from 0.69 to 0.49, and **GPT-4o** dips in IO from 0.60 to 0.41. These cases may reflect occasional overemphasis on scene adaptation at the expense of broader generalization. Further refinement of the balancing mechanism between grounding and default procedural knowledge could help mitigate these trade-offs.

5.2 Plausibility Score

The **plausibility score** captures the internal logical consistency and goal-directedness of the generated procedure as a whole. This metric complements groundness by assessing whether the action sequence, aligns with coherent human-like reasoning. As shown in Table 2, ImPlan improves plausibility across all model types, though the magnitude and pattern of improvement vary with model scale, architecture, and baseline performance.

Consistent Gains Across Models. All ten evaluated models show a positive gain in average plausibility after applying ImPlan. The most significant improvements are observed in **Gemini 1.5** (**+28.57%**), **Gemini 2.0** (**+28.46%**), and **Qwen2.5-72B** (**+21.97%**). These gains are substantial and consistent across all five subcategories (CR, CT, IO, ND, TQ), indicating that ImPlan robustly

Table 2: Comparison of model **Plausibility Score** across different settings. AVG = average of CR, CT, IO, ND, TQ; Gain (%) = percentage improvement from base to implan.

Model	Setting	CR	CT	IO	ND	TQ	AVG	Gain (%)
GPT-4.1	base	0.59	0.54	0.63	0.70	0.53	0.60	
	implan	0.79	0.55	0.74	0.65	0.60	**0.67**	**11.37**
GPT-4.1-mini	base	0.56	0.52	0.70	0.68	0.56	0.60	
	implan	0.75	0.62	0.81	0.67	0.58	**0.69**	**13.58**
GPT-4.1-nano	base	0.45	0.47	0.64	0.67	0.48	0.54	
	implan	0.61	0.48	0.57	0.65	0.61	**0.58**	**7.75**
GPT-4o	base	0.56	0.47	0.66	0.71	0.59	0.60	
	implan	0.78	0.58	0.67	0.64	0.58	**0.65**	**8.70**
Gemini 1.5	base	0.32	0.38	0.50	0.58	0.39	0.43	
	implan	0.43	0.50	0.62	0.66	0.58	**0.56**	**28.57**
Gemini 2.0	base	0.50	0.48	0.58	0.65	0.46	0.53	
	implan	0.68	0.62	0.80	0.80	0.53	**0.69**	**28.46**
LLaMA3.2-11B	base	0.52	0.49	0.61	0.57	0.46	0.53	
	implan	0.58	0.51	0.78	0.63	0.53	**0.61**	**14.34**
LLaMA4-Mav-17B	base	0.56	0.55	0.57	0.63	0.49	0.56	
	implan	0.76	0.50	0.59	0.65	0.58	**0.62**	**10.00**
LLaMA4-Scout-17B	base	0.51	0.49	0.52	0.56	0.48	0.51	
	implan	0.55	0.49	0.50	0.63	0.53	**0.54**	**5.47**
Qwen2.5-72B	base	0.53	0.52	0.62	0.56	0.41	0.53	
	implan	0.73	0.55	0.67	0.71	0.56	**0.64**	**21.97**

enhances models' ability to reason coherently about tool-use scenarios. Notably, the **Gemini series**, despite moderate to low base performance, benefits most from the addition of ImPlan, with **Gemini 2.0** achieving a new peak plausibility of **0.69**, surpassing GPT variants. Looking across the five sub-tasks, the most consistent improvements are seen in **CR** and **IO**—categories. For instance, **GPT-4.1** improves its CR score from 0.59 to 0.79, and **Gemini 2.0** improves IO from 0.58 to 0.8.

Gains for Proprietary Models. The **GPT family** consistently benefits from ImPlan, though to a lesser extent than some of the open-weight models. **GPT-4.1-mini** improves from 0.60 to **0.69** (**+13.58%**), while **GPT-4.1** sees a gain of **+11.37%**. Interestingly, **GPT-4.1-nano**, the smallest variant, shows the smallest absolute improvement (+0.04) and a relatively modest percentage gain (**+7.75%**), despite a low starting point. This suggests that while ImPlan improves coherence across the board, its effectiveness may be constrained by the capacity of smaller proprietary models to encode and retain long-range procedural logic.

Open-Weight Model Behavior. Among open-weight models, **Qwen2.5-72B** and **LLaMA3.2-11B** show strong gains in plausibility (**+21.97%** and **+14.34%**, respectively), confirming that ImPlan's benefits generalize well beyond proprietary VLMs. **LLaMA4-Mav-17B** shows a gain (**+10.00%**), while **LLaMA4-Scout-17B** shows the lowest relative improvement (**+5.47%**). Interestingly, the Scout model sees stronger gains in groundness, indicating that it may already produce moderately coherent plans, and ImPlan primarily helps align those with scene context rather than improving their internal logic.

5.3 Unified Evaluation of Groundedness and Plausibility

While groundness and plausibility measure distinct aspects of procedural quality, contextual feasibility and logical coherence, respectively, robust real-world planning demands strong performance on both. Overemphasis on plausibility alone can lead to superficially coherent but visually implausible plans, while high groundness without coherent logic can result in disjointed or incomplete action sequences. Thus, a method that can simultaneously enhance both dimensions is crucial for generating usable and trustworthy plans.

Figure 3 presents a joint visualization of model performance across these two axes (average scores). Each model is represented as a vector from its base configuration (empty circle) to its ImPlan-enhanced version (filled circle). Across all models, regardless of architecture family (GPT, Gemini, LLaMA, Qwen) or parameter scale, ImPlan consistently **shifts performance upward and to the right**. This pattern clearly indicates simultaneous improvement in both contextual grounding and logical planning, without evidence of trade-offs between the two.

The directionality and length of the improvement vectors offer additional insight. For instance, models with weaker base performance, such as Qwen2.5-72B, LLaMA3.2-11B, and Gemini 1.5,

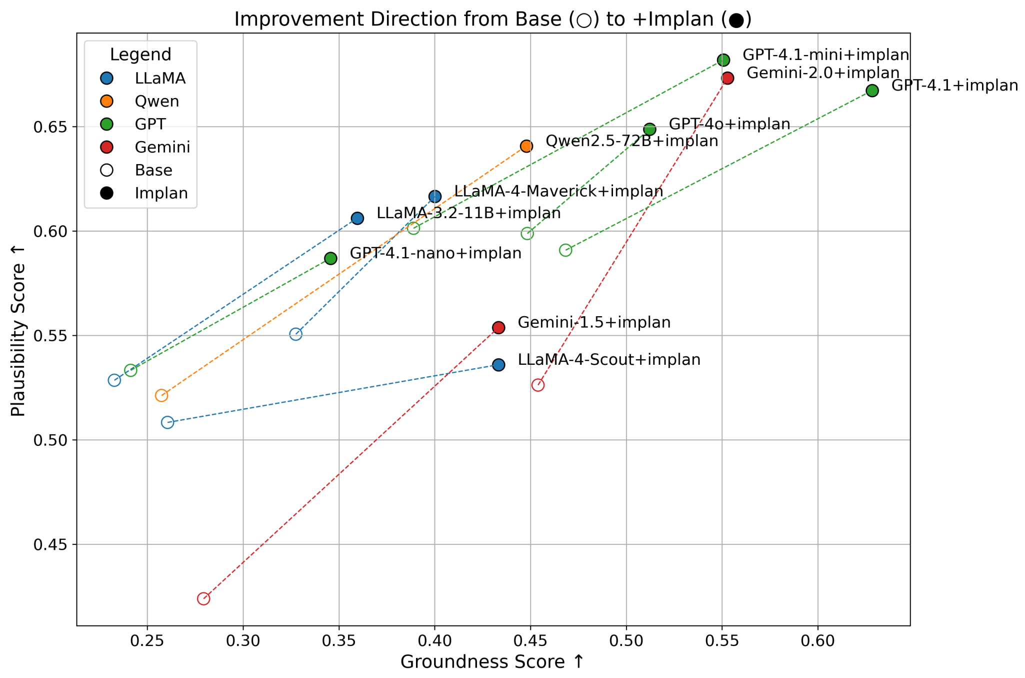

Figure 3: Joint performance of models on **Groundness Score** and **Plausibility Score**. Arrows indicate improvement direction from base (○) to ImPlan (●). ImPlan consistently enhances both context grounding and logical coherence across all model families.

show substantial leaps in both dimensions, demonstrating ImPlan's ability to elevate underperforming models across the board. Meanwhile, already strong models like GPT-4.1 and Gemini 2.0 benefit from more modest but still significant gains, suggesting that ImPlan complements rather than replaces existing procedural priors. Interestingly, post-ImPlan models tend to cluster in the upper-right quadrant of the plot, forming a tighter performance band than in the base configuration. This convergence indicates that ImPlan serves as a regularizing force across heterogeneous models, improving not just raw scores but also alignment in quality across systems. Furthermore, the lack of any downward or leftward movements confirms that ImPlan does not introduce a negative effect in either metric.

Overall, the joint analysis reinforces the findings of the previous subsections: ImPlan provides broad and balanced improvements in zero-shot procedural plan generation. Its consistent bidirectional gains across model types support the core claim of this work—that explicit scene-grounded reasoning mechanisms can elevate large language models beyond memorized, default behaviors toward more adaptive and context-sensitive planning.

6 Conclusion

We presented **ImPlan**, a framework that equips vision-language models with structured improvisational reasoning for grounded procedural adaptation. Unlike standard prompting, ImPlan enables models to generate context-sensitive action plans when ideal tools are missing or altered. Experiments on expert-annotated emergency procedures show that ImPlan yields significant gains in both *groundness* (up to **70.8%**) and *plausibility* (up to **28.6%**) across a range of models. These results demonstrate that structured adaptation can harmonize visual grounding and logical coherence, advancing the robustness of AI planning in real-world, resource-limited settings.

Acknowledgments and Disclosure of Funding

This work was supported by the US Army Medical Research and Development Command under Contract No. W81XWH21C0119 and by NSF under Award No. 521982.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Kenza Amara, Lukas Klein, Carsten Lüth, Paul Jäger, Hendrik Strobelt, and Mennatallah El-Assady. Why context matters in vqa and reasoning: Semantic interventions for vlm input modalities. *arXiv preprint arXiv:2410.01690*, 2024.

Xuweiyi Chen, Ziqiao Ma, Xuejun Zhang, Sihan Xu, Shengyi Qian, Jianing Yang, David Fouhey, and Joyce Chai. Multi-object hallucination in vision language models. *Advances in Neural Information Processing Systems*, 37:44393–44418, 2024.

An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 135062–135093, 2025.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=vvoWPYqZJA`.

Alessandro Favero, Luca Zancato, Matthew Trager, Siddharth Choudhary, Pramuditha Perera, Alessandro Achille, Ashwin Swaminathan, and Stefano Soatto. Multi-modal hallucination control by visual information grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14303–14312, 2024.

Yunpeng Guo, Xinyi Zeng, Pinxian Zeng, Yuchen Fei, Lu Wen, Jiliu Zhou, and Yan Wang. Common vision-language attention for text-guided medical image segmentation of pneumonia. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 192–201. Springer, 2024.

Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. Llava-med: Training a large language-and-vision assistant for biomedicine in one day. *arXiv preprint arXiv:2306.00890*, 2023.

Qingqiu Li, Xiaohan Yan, Jilan Xu, Runtian Yuan, Yuejie Zhang, Rui Feng, Quanli Shen, Xiaobo Zhang, and Shujun Wang. Anatomical structure-guided medical vision-language pre-training. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 80–90. Springer, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.

Xinyu Lyu, Beitao Chen, Lianli Gao, Jingkuan Song, and Heng Tao Shen. Alleviating hallucinations in large vision-language models through hallucination-induced optimization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 122811–122832. Curran Associates, Inc., 2024. URL `https://proceedings.neurips.cc/paper_files/paper/2024/file/dde040998d82553cf7f689e8ae173d5a-Paper-Conference.pdf`.

Malvina Nikandrou, Georgios Pantazopoulos, Nikolas Vitsakis, Ioannis Konstas, and Alessandro Suglia. Crope: Evaluating in-context adaptation of vision and language models to culture-specific concepts. *arXiv preprint arXiv:2410.15453*, 2024.

Shengyi Qian, Weifeng Chen, Min Bai, Xiong Zhou, Zhuowen Tu, and Li Erran Li. Affordancellm: Grounding affordance from vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7587–7597, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1:9, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763, 2021.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019.

Fereshteh Shakeri, Yunshi Huang, Julio Silva-Rodríguez, Houda Bahig, An Tang, Jose Dolz, and Ismail Ben Ayed. Few-shot adaptation of medical vision-language models. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 553–563. Springer, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Jiajin Zhang, Ge Wang, Mannudeep K Kalra, and Pingkun Yan. Disease-informed adaptation of vision-language models. *IEEE Transactions on Medical Imaging*, 2024a.

Sheng Zhang, Yanbo Xu, Naoto Usuyama, Hanwen Xu, Jaspreet Bagga, Robert Tinn, Sam Preston, Rajesh Rao, Mu Wei, Naveen Valluri, Cliff Wong, Andrea Tupini, Yu Wang, Matt Mazzola, Swadheen Shukla, Lars Liden, Jianfeng Gao, Angela Crabtree, Brian Piening, Carlo Bifulco, Matthew P. Lungren, Tristan Naumann, Sheng Wang, and Hoifung Poon. A multimodal biomedical foundation model trained from fifteen million image–text pairs. *NEJM AI*, 2(1), 2024b. doi: 10.1056/AIoa2400640.

A Procedural Graph Construction and Averaging

A.1 Graph Construction from Sequential Actions

Given a sequential list of procedural actions annotated as verb-noun pairs, the construction of the procedural graph proceeds incrementally. Each unique action or object is mapped to a distinct node, and directed edges are created to represent the operational transitions. Temporal order is captured via ordinal labels on edges.

The following pseudocode summarizes the graph construction process:

Algorithm 2 Graph Construction from Action Sequences

```
Require: Action sequence $\{a_1, a_2, \ldots, a_{N_A}\}$
Ensure: Adjacency matrix $P$, Ordinal matrix $Q$
 1: Initialize node sets $U, V \leftarrow \emptyset$, edge set $E \leftarrow \emptyset$
 2: Initialize counters for repeated actions
 3: for each action $a_i$ do
 4:     Parse $a_i$ into verb $u_i$ and noun $v_i$
 5:     if $u_i$ or $v_i$ not in $U$ or $V$ then
 6:         Add new nodes $u_i, v_i$
 7:     end if
 8:     Add directed edge from previous noun $v_{i-1}$ to current verb $u_i$
 9:     Add directed edge from current verb $u_i$ to current noun $v_i$
10:     Annotate edges with ordinal index $i$
11: end for
12: Build matrices $P, Q$ from collected edges
```

A.2 Averaged Stereotypical Graph Construction

To construct the canonical stereotypical graph G^* for each procedural goal, multiple instance graphs are aggregated. The adjacency and ordinal matrices across different demonstrations are averaged elementwise. A thresholding step is applied to prune unreliable transitions.

The averaging procedure is detailed below:

Algorithm 3 Stereotypical Graph Averaging

```
Require: Matrices $\{(P_1, Q_1), (P_2, Q_2), \ldots, (P_K, Q_K)\}$, threshold $\alpha$
Ensure: Averaged matrices $\bar{P}, \bar{Q}$
 1: for each node pair $(i, j)$ do
 2:     Compute $\bar{P}(i,j) = \frac{1}{K} \sum_{k=1}^{K} P_k(i,j)$
 3:     Compute $\bar{Q}(i,j) = \frac{1}{K} \sum_{k=1}^{K} Q_k(i,j)$
 4:     if $\bar{P}(i,j) < \alpha$ then
 5:         Set $\bar{P}(i,j) = 0$, remove edge
 6:     else
 7:         Set $\bar{P}(i,j) = 1$, retain edge
 8:     end if
 9:     Set $\bar{Q}(i,j) \leftarrow \bar{P}(i,j) \times \bar{Q}(i,j)$
10: end for
```