
Supplementary Material

IR3D-Bench: Evaluating Vision-Language Model Scene Understanding as Agentic Inverse Rendering

Anonymous Author(s)

Affiliation

Address

email

1 A Experiment Setup Details

2 A.1 Implement Details

3 To ensure consistency across models with reasoning capabilities, such as GPT-4o [1] and Grok-3 [2]
4 etc, which often generate intermediate "thought" steps or chain-of-thought reasoning, we discard all
5 non-structured outputs during inference. We extract only the final JSON-formatted response that
6 conforms to our predefined schema. In designing the evaluation metrics, we intentionally ignore
7 lighting-related attributes (shading and brightness) since these aspects are not explicitly modeled by
8 the agents and often introduce high variability. Moreover, accurately recovering lighting conditions
9 from a single image is inherently ambiguous, even for human annotators, making it an unrealistic
10 expectation for current VLMs. Thus, we focus evaluation on geometry and semantics, rather than
11 photometric fidelity. Additionally, we observe that most models output zero object rotation by default;
12 hence, we omit rotation from evaluation to simplify the task and focus on more meaningful aspects of
13 spatial understanding. In this paper, we evaluate all models on a representative subset of the CLEVR
14 dataset's validation split, containing 1,500 image-scene pairs with GT annotations.

15 A.2 Task prompt

16 **Design of Inverse Rendering Prompt** The complete prompt used for inverse rendering is presented
17 in Table 1. Given an input image, the vision-language model is instructed to extract scene-level
18 geometry, material properties, and lighting parameters, and to output the results in a strict JSON
19 format. This structured output is then parsed and used to reconstruct the 3D scene within Blender [3].
20 The prompt is carefully designed to ensure consistency, reproducibility, and compatibility with
21 downstream scene synthesis workflows.

22 **Design of LLM Score Prompt** The complete prompt used to elicit LLM-based evaluations of
23 reconstructed 3D scenes is detailed in Table 2. This prompt instructs the LLM to compare a predicted
24 scene description against a ground-truth JSON, and to assign scores across multiple dimensions of
25 fidelity and accuracy. The evaluation process is fully self-contained, requiring the model to analyze
26 only the provided JSON content without recourse to external assumptions. Scores are returned in a
27 structured JSON format, with each score accompanied by a concise justification referencing specific
28 attributes or spatial discrepancies.

29 **Design of Refinement Prompt** The complete prompt used for refining the scene description based
30 on prior outputs is presented in Table 3. This prompt instructs the LLM to revise the predicted JSON
31 scene by leveraging both the ground-truth image and the rendered image from the current JSON, under

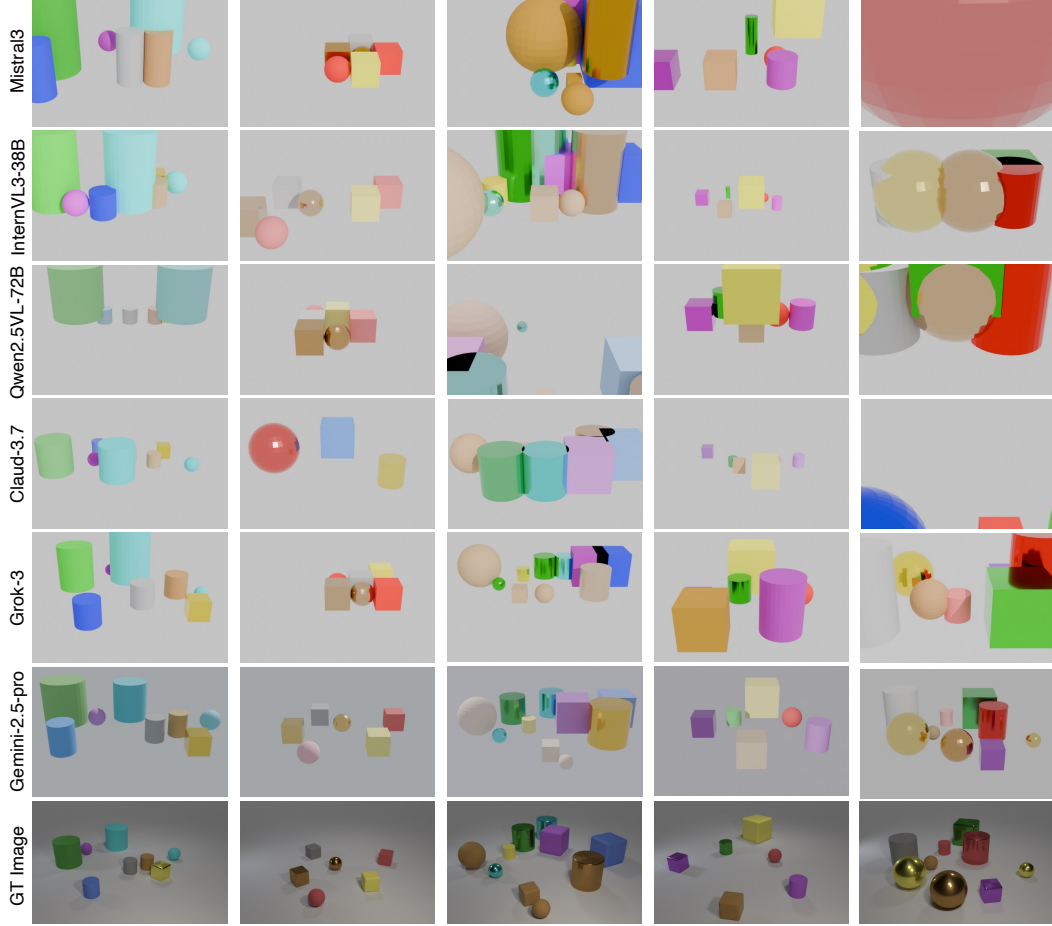


Figure 1: More Visual Results with Selected VLMs on IR3D-Bench

a fixed camera setup. The objective is to produce a refined scene JSON that is visually and spatially aligned with the ground-truth reference, in terms of object layout, attributes, and relationships. LLM is required to output a valid JSON object conforming to a strict schema (the same as the inverse rendering prompt), with no additional text, ensuring consistency and interpretability for downstream evaluation.

B More Experimental Results

B.1 More Visual Results

We present additional qualitative comparisons in Figure 1. Among all models, Gemini-2.5 Pro [4] achieves the most faithful reconstructions in terms of geometry, spatial layout, and material appearance. Grok-3 [2] shows competitive performance in recovering fine material details, though with minor spatial inconsistencies. In contrast, models such as Mistral-3 [5], InternVL-Chat-3B [6], Qwen2.5VL-72B [7], and Claude-3-7 [8] often exhibit noticeable errors in object positioning and material prediction, leading to spatial misalignments, incorrect relative depths, and inconsistencies in color or reflectance. These observations further highlight the strengths of Gemini-2.5 Pro in holistic scene understanding.

B.2 Impact of Prompt Design

To quantitatively assess the contribution of different prompt components, we perform ablation studies on four key elements: structured output format, fixed camera configuration, task decomposition, and

Table 1: Prompt for Inverse Rendering

3D Inverse Rendering Prompt Specification	
#1	Task Description Please carefully analyze the provided image, identifying all major geometric objects, their properties, and the scene’s lighting setup. Your task is to extract object and lighting information and return the result strictly following the JSON format specified below. The Camera parameters are fixed and should be used as provided in the JSON structure. This JSON will be used by a Python script to reconstruct the scene in Blender.
#2	Output Format Requirements <ul style="list-style-type: none"> Please output only a valid JSON object, without any additional explanations, comments, or code block markers (like “<code>json ...</code>”). The JSON object must adhere to the following structure: <pre> "camera": { // --- CRITICAL: Use these FIXED values for the camera --- "name": "MainCamera", "location": [0.0, -10.0, 5.0], "lens": 50.0, "rotation_euler": [65.0, 0.0, 0.0], // Provided in degrees "sensor_width": 36.0, "sensor_height": 36.0, "clip_start": 0.1, "clip_end": 100.0 // --- Do NOT estimate camera parameters --- }, "lighting": { "sun_energy": float, // Estimated sun light intensity (e.g., between 2.0 and 5.0) "sun_rotation_euler_degrees": [float, float, float], // Rotation angles [X, Y, Z] "environment_color": [float, float, float, float], // RGBA, e.g., [0.8, 0.8, 0.8, 1.0] "environment_strength": float // Light strength, e.g., between 1.0 and 1.5 }, "objects": [{ "name": "descriptive_name_string", // e.g., "green large metal cylinder" "location": [float, float, float], // [X, Y, Z] "rotation_euler": [float, float, float], // Usually [0, 0, 0] "size_params": { // One of the following based on shape: // "size": float // If 'cube' // "radius": float, "depth": float // If 'cylinder' // "radius": float // If 'sphere' }, "material": { "name": "MaterialNameString", // e.g., "GreenMetal" "base_color": [float, float, float, float], // [R, G, B, A] "metallic": float, // Between 0.0 and 1.0 "roughness": float // Between 0.0 and 1.0 } } // ... Potentially multiple object entries ...] </pre>
#3	Object Analysis Guidelines: <ul style="list-style-type: none"> Identification: Find all clearly visible, primary geometric objects in the image. Name (name): Use the format "color size material shape", e.g., "blue large rubber cube". Location (location): Estimate the object’s center position as [X, Y, Z]. Assume the ground is at Z = 0, and Z is usually half the object’s height. Estimate X and Y based on the object’s left-right and front-back position in the image. Rotation (rotation_euler): For CLEVR-style images, objects are usually upright. Use [0.0, 0.0, 0.0] unless there is visual evidence to the contrary. Size Parameters (size_params): Based on the object shape, include: <ul style="list-style-type: none"> cube: "size": float (estimated edge length) cylinder: "radius": float, "depth": float (estimated base radius and height) sphere: "radius": float (estimated radius) Estimate all size values visually relative to other objects in the image. Material (material): <ul style="list-style-type: none"> name: Generate a concise material name, e.g., "GreenMetal". Reuse the same name for identical materials across objects. base_color: RGBA color in format [R, G, B, A], where values are between 0.0 and 1.0. metallic: Use 1.0 for metallic surfaces, 0.0 for non-metal surfaces. roughness: Estimate based on surface appearance — smooth/mirror-like surfaces have low roughness (near 0.0), matte/dull surfaces have high roughness (near 1.0).

Table 2: Evaluation Prompt for Scoring 3D Scene JSON

3D Scene JSON Description Evaluator Prompt	
#1	Role and Task You are an AI evaluator specializing in 3D scene descriptions. Your task is to compare a Predicted JSON scene description with a Ground Truth (GT) JSON and evaluate the accuracy and consistency of the predicted scene.
#2	Instructions <ul style="list-style-type: none"> • The Predicted JSON will be provided first. • The GT JSON will follow. • Focus only on the data in the JSONs — no external visual interpretation or assumptions. • Acknowledge and account for structural differences across JSONs. • Your output must be a single valid JSON object following the format below — no other text or explanations.
#3	Scoring Scale (Per Dimension) <ul style="list-style-type: none"> • 5: Excellent — Highly accurate and consistent • 4: Good — Mostly accurate, minor discrepancies • 3: Fair — Captures core ideas but with noticeable issues • 2: Poor — Significant inaccuracies • 1: Very Poor — Major incorrect aspects • 0: Completely Incorrect or Missing
#4	Evaluation Dimensions
1	GPT4.1-JSON Object Appearance Fidelity <i>Focus:</i> Can plausible object matches be found? How accurate are the predicted attributes vs GT? <ul style="list-style-type: none"> • Match predicted name (color/size/material/shape) with GT attributes. • Compare predicted material fields (e.g., metallic) with GT material category. • Compare predicted size_params to size descriptors like "small"/"large". <i>Score reflects object match quality and attribute-level accuracy.</i>
2	GPT4.1-JSON Scene Layout Accuracy <i>Focus:</i> For matched objects, how close are predicted location values to GT 3d_coords? <i>Score reflects 3D spatial alignment accuracy.</i>
3	GPT4.1-JSON Overall Visual Quality & Similarity <i>Focus:</i> Holistic assessment of how well the predicted JSON matches the GT. <ul style="list-style-type: none"> • Consider object count, attributes, locations. • Identify any major inconsistencies within the predicted data. <i>Score reflects overall scene description quality.</i>
#5	Expected Output Format (After receiving both JSONs) <pre> json { "GPT4_1_JSON_Object_Appearance_Fidelity": { "score": <integer 0-5>, "justification": "<string explanation of matching success and attribute accuracy, noting structural differences and specific examples>" }, "GPT4_1_JSON_Scene_Layout_Accuracy": { "score": <integer 0-5>, "justification": "<string qualitative assessment of the similarity between predicted locations and GT 3d_coords for matched objects>" }, "GPT4_1_JSON_Overall_Visual_Quality_and_Similarity": { "score": <integer 0-5>, "justification": "<string explanation for the overall data accuracy score>" } } </pre>

Table 3: Structured refinement prompt for 3D scene JSON correction based on GT and predicted image comparison.

Refinement Prompt Based on GT and Predicted Images	
#1	Inputs <ul style="list-style-type: none"> • GT Image: Ground truth image of the scene (accurate reference). • Pred Image: Rendered image from the current JSON scene. • Current JSON: Scene description generated from the predicted image.
#2	Refinement Goals <ul style="list-style-type: none"> • Objective: Refine the parameters of all objects in a 3D scene JSON to closely match a provided ground truth (GT) image, under a fixed camera setup. NOTICE: The refined attributes of all objects in the refined json file should not be all the same as the input json file. • Goal: Achieve a refined scene JSON whose rendered image (with the fixed camera) is visually and spatially consistent with the GT image, in terms of object count, placement, size, shape, material, and inter-object relationships.
#3	Constraints <ul style="list-style-type: none"> • All changes must be grounded in visual evidence from GT vs Pred and metric feedback. • The final output must be a valid JSON object that strictly follows the original schema. • Output only the JSON object — no extra explanation, comments, or formatting.
#4	Output Format Requirements <ul style="list-style-type: none"> • Please output only a valid JSON object, without any additional explanations, comments, or code block markers (like json ...). The JSON object must adhere to the following structure: • Camera format: <pre> "camera": { // --- CRITICAL: Use these FIXED values for the camera --- "name": "MainCamera", "location": [0.0, -10.0, 5.0], "lens": 50.0, "rotation_euler": [65.0, 0.0, 0.0], // Provided in degrees "sensor_width": 36.0, "sensor_height": 36.0, "clip_start": 0.1, "clip_end": 100.0 // --- Do NOT estimate camera parameters --- }, </pre> • Lighting format: <pre> "lighting": { "sun_energy": float, // Estimated sun light intensity (e.g., between 2.0 and 5.0) "sun_rotation_euler_degrees": [float, float, float], // Rotation angles [X, Y, Z] "environment_color": [float, float, float, float], // RGBA, e.g., [0.8, 0.8, 0.8, 1.0] "environment_strength": float // Light strength, e.g., between 1.0 and 1.5 }, </pre> • Objects format: <pre> "objects": [// --- CRITICAL: Refine the parameters of each object --- { "name": "descriptive_name_string", // e.g., "green large metal cylinder" "location": [float, float, float], // [X, Y, Z] "rotation_euler": [float, float, float], // Usually [0, 0, 0] "size_params": { // One of the following based on shape: // "size": float // If 'cube' // "radius": float, "depth": float // If 'cylinder' // "radius": float // If 'sphere' }, "material": { "name": "MaterialNameString", // e.g., "GreenMetal" "base_color": [float, float, float, float], // [R, G, B, A] "metallic": float, // Between 0.0 and 1.0 "roughness": float // Between 0.0 and 1.0 } } // ... Potentially multiple object entries ...] </pre>

Table 4: **Quantitative results of models with various prompt designs on IR3D-Bench.** We report performance across key aspects of 3D scene reconstruction from a single image.

Models	Layout & Localization		Relation	Instance Seg.	CLIP Score					LLM Score			
	Pix. Dist.↓	Count ACC↑			Bbox↑	Rel. ACC↑	IOU↑	DICE↑	Color↑	Size↑	Material↑	Shape↑	Overall↑
w.o. Given Format	-	-	-	-	-	-	-	-	-	-	-	-	-
w.o. Given Camera	0.7156	0.94	0.06	0.24	0.02	0.03	96.57	97.58	98.02	99.59	93.46	2.69	1.91
w.o. Task Analysis	0.6718	0.91	0.27	0.28	0.06	0.09	97.01	97.53	97.79	99.14	93.65	2.84	2.05
w.o. Attribute Guides	0.5891	0.94	0.26	0.27	0.07	0.1	97.15	98.15	97.59	99.59	94.03	2.73	1.85
GPT-4o	0.5528	0.94	0.29	0.3	0.07	0.11	96.7	98.36	98.66	99.88	94.22	2.9	1.94

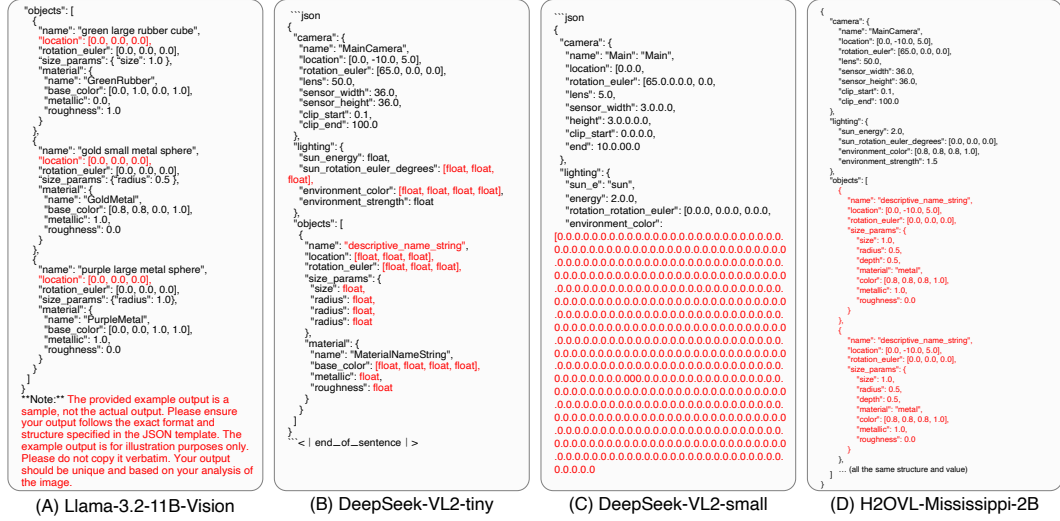


Figure 2: Failure output of selected models on IR3D-Bench

attribute specification. We use GPT-4o [1] as the VLA for conducting these experiments. As shown in Table 4, the full prompt achieves the best overall performance across almost all evaluation metrics, with a total Pixel Distance of 0.5528 and DICE score 0.11. Removing the structured format results in outputs lacking essential attributes required for rendering, thereby making it fail to render and compute downstream reconstruction metrics. This highlights the critical role of structured prompts in constraining the output space and ensuring syntactic and semantic completeness. Eliminating the fixed camera setup results in worsened spatial consistency, particularly evident in reduced shape accuracy (99.14 vs. 99.88) and a lower layout score (1.91 vs. 1.94). Without task decomposition and clarification, the model struggles with compositional reasoning, leading to a drop in object count accuracy (0.91 vs. 0.94) and Pixel Distance (0.7156 vs. 0.5528). Finally, omitting attribute guidelines significantly impacts fine-grained predictions, notably reducing material accuracy (97.59 vs. 98.66) and object-level score (2.73 vs. 2.90). These results empirically validate that each prompt component plays a critical role in guiding the VLM toward faithful and consistent scene reconstruction.

B.3 Analysis of Failure Cases

Among the evaluated models, LLaMA-3.2-11B-Vision [9], DeepSeek-VL2 [10] (tiny and small version), and H2OVL-Mississippi-2B [11] represent clear failure cases on IR3D-Bench. Despite the general success of many models, these models consistently fail to produce valid outputs suitable for inverse rendering. As illustrated in Figure 2, LLaMA-3.2-11B-Vision repeatedly emits a static template across all test cases, with identical object attributes and all object locations fixed at (0, 0, 0), rendering the outputs semantically meaningless and non-renderable. DeepSeek-VL2-tiny directly copies the JSON template, including placeholders such as [float, float, float], without generating any instance-specific values. Similarly, DeepSeek-VL2-small fails to populate essential fields, instead outputting large arrays of zeros without assigning any object-level attributes. H2OVL-Mississippi-2B generates multiple objects, but with identical and repetitive attributes across all instances, suggesting template replication without true scene interpretation. In all these cases, the lack

75 of structurally valid and semantically grounded output prevents rendering and quantitative evaluation,
76 highlighting the importance of model understanding and prompt grounding in 3D scene tasks.

77 C Further Analysis

78 C.1 Limitation

79 While IR3D-Bench offers a novel lens to evaluate vision-language models (VLMs) through agentic
80 inverse rendering, several limitations remain. First, the benchmark is constructed on the CLEVR
81 dataset, which contains synthetic scenes with clean geometry and controlled semantics. While
82 this design enables precise evaluation, it lacks the visual richness and noise inherent in real-world
83 data. We intentionally refrain from using real-world datasets at this stage because most models still
84 struggle to perform reliably even under the simplified CLEVR setting. Second, our current evaluation
85 focuses on single-view, static scene reconstruction, without considering temporal consistency or
86 multi-view fusion, both of which are essential for reasoning in dynamic and embodied environments.
87 Lastly, we fix the camera intrinsics and extrinsics and omit illumination modeling. This decision is
88 made not only to reduce task ambiguity, but also because current models already exhibit substantial
89 limitations in reconstructing geometry and semantics under these simplified conditions. Introducing
90 additional complexity at this stage may obscure rather than clarify the core challenges in agentic
91 visual understanding.

92 C.2 Future Extension

93 IR3D-Bench provides a structured setting that can support the construction of high-quality instruction-
94 output pairs for supervised fine-tuning (SFT) or chain-of-thought (CoT) training. By leveraging
95 successful inverse rendering examples, we can curate targeted datasets to improve VLMs’ com-
96 positional reasoning and program generation capabilities. Building on this, future extensions of
97 IR3D-Bench could extend to multi-view and dynamic scenes, and incorporate camera parameter
98 estimation and illumination modeling. Ultimately, extending the benchmark to real-world datasets
99 with diverse appearance, clutter, and geometry will enable comprehensive evaluation and training of
100 VLAs in open-world, visually complex environments.

101 References

- 102 [1] OpenAI. gpt4o, 2024.
- 103 [2] Anthropic. The Grok Model Family: xAI. <https://grok.com/>.
- 104 [3] Blender Online Community. Blender - a 3D modelling and rendering package, 2016.
- 105 [4] Anthropic. The Gemini 2 Model Family: Google Deepmind. <https://gemini.google.com/>.
- 106 [5] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh
107 Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile
108 Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut
109 Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint*
110 *arXiv:2310.06825*, 2023.
- 111 [6] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Yuchen Duan,
112 Hao Tian, Weijie Su, Jie Shao, et al. InternV13: Exploring advanced training and test-time
113 recipes for open-source multimodal models. *arXiv:2504.10479*, 2025.
- 114 [7] Qwen An Yang, Baosong Yang, and Beichen Zhang et al. Qwen2.5 technical report.
115 *arXiv:2412.15115*, 2024.
- 116 [8] Anthropic. The Claude 3 Model Family: Opus, Sonnet, Haiku. <https://www.anthropic.com/news/claude-3-family>.
- 117
- 118 [9] Meta AI. The Llama 3 Herd of Models. *arXiv:2407.21783*, 2024.

- 119 [10] Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao,
120 Yiyang Ma, Chengyue Wu, Bingxuan Wang, et al. Deepseek-vl2: Mixture-of-experts vision-
121 language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*,
122 2024.
- 123 [11] Shaikat M. Galib, Shanshan Wang, Guanshuo Xu, Pascal Pfeiffer, Ryan Chesler, Mark Landry,
124 and SriSatish Ambati. H2ovl-mississippi vision language models technical report. *ArXiv*,
125 abs/2410.13611, 2024.