DEPTHAGENT: ADVANCING GEOMETRY-ADAPTIVE DEPTH ESTIMATION VIA SAMPLE-WISE EXPERT SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Monocular metric depth estimation has achieved strong progress with large-scale training and universal-camera modeling, yet robust deployment across diverse camera settings, such as perspective, fisheye, and panoramic images, remains challenging. Existing methods typically rely on a single depth estimator, overlooking that different models encode different camera assumptions and perform best under different input domains. In this paper, we show that depth experts exhibit strong sample-wise complementarity: model preference is highly correlated with camera geometry, and multi-model fusion brings the largest gains on difficult samples where individual experts are unreliable. Motivated by these observations, we propose **DepthAgent**, a vision-language agent for geometry-adaptive monocular depth estimation. DepthAgent treats existing depth models as frozen tools and learns to analyze scene and camera cues, invoke suitable experts through multi-turn tool utilization, and select or fuse their predictions for each input. To optimize such discrete decision-making toward dense geometric quality, we design a multi-reward reinforcement fine-tuning scheme that jointly encourages valid tool execution, camera/scene analysis, expert-se/lection quality, and inference efficiency. Extensive experiments across perspective, fisheye, and panoramic benchmarks show that DepthAgent consistently outperforms individual experts, fixed model fusion, and rule-based and learned routers with fewer expert calls, with strong improvements on challenging samples. The code and model will be released upon publication.

1 INTRODUCTION

Monocular depth estimation is a fundamental problem for 3D scene understanding, with broad applications in autonomous driving, robotics, and augmented/virtual reality (Wang et al., 2019; Dong et al., 2022; Du et al., 2020). Recent advances have substantially improved monocular depth prediction through large-scale supervised and self-supervised learning (Yang et al., 2024a), diffusion-based geometric priors (Ke et al., 2024), and zero-shot metric depth estimation (Yin et al., 2023). Since many applications require accurate scale for spatial reasoning, monocular metric depth estimation (MMDE), which recovers depth in absolute units, has attracted growing interest. However, real-world deployment rarely follows a single imaging assumption. Images may come from standard perspective cameras, challenging perspective settings, fisheye lenses, or equirectangular projected (ERP) panoramas. Although recent works attempt to extend monocular depth estimation to large-FoV and universal-camera settings (Guo et al., 2025b; Ganesan et al., 2026; Piccinelli et al., 2025a), robust depth estimation across heterogeneous camera geometries and diverse scenarios remains challenging.

A central challenge in robust monocular depth estimation lies not only in the lack of strong depth models, but also in the fact that existing models encode different imaging assumptions and thus perform best under different camera geometries or data distributions. For instance, perspective-oriented models are well suited to standard perspective images, while ERP-trained models are better adapted to large-FoV and native panoramic/360° inputs. As a result, no single model is consistently optimal across scenarios, motivating adaptive model selection or combination. However, model combination remains underexplored in monocular depth estimation. Although routing and expert-selection strategies have been studied in language, vision, and multimodal systems (Ong et al., 2024; Zong et al., 2024; Zhu et al., 2026a), they cannot be directly applied to depth estimation, where

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

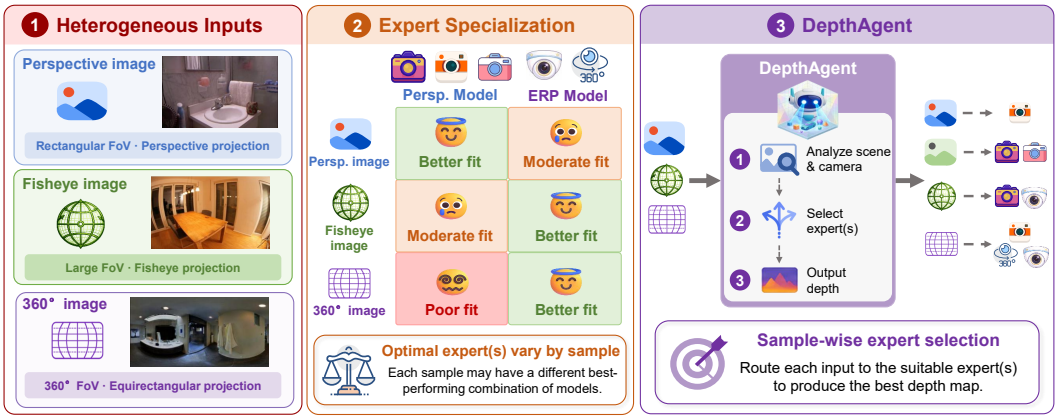


Figure 1: **Motivation of DepthAgent.** Real-world inputs span heterogeneous camera domains, including perspective, fisheye, and panoramic images, for which different depth experts exhibit different strengths. Instead of relying on a single model for all inputs, DepthAgent uses scene and camera cues to select suitable expert(s) on a per-sample basis, producing more reliable depth maps across diverse camera settings.

success depends on dense geometric accuracy rather than discrete labels or high-level preferences. A naive solution is to select a single fixed model or to fuse all expert predictions using a predefined rule, but both are limited: the former overlooks complementary expert strengths, whereas the latter may indiscriminately incorporate unreliable predictions and degrade the final depth estimate. Therefore, the key missing component is an input-adaptive mechanism that determines whether experts should be combined and which experts should participate. This leads to a research question: *given multiple depth models for the same task, how should one choose the most suitable model or model combination for each test sample?*

To examine whether such a mechanism is necessary and feasible, we first conduct a systematic analysis of single-model performance, oracle-style model selection, and multi-model fusion across diverse models and datasets. Our analysis reveals that **multi-model fusion often outperforms individual models** on most samples, and model preference is strongly correlated with camera geometry: perspective-oriented models are generally more effective on standard perspective inputs, while ERP-trained models are more reliable for native panoramic images. We further observe that the benefit of fusion is not uniform, but becomes more pronounced on difficult samples where single-model predictions are less reliable. These findings suggest that robust monocular depth estimation should not rely on a fixed estimator or a static fusion rule, but instead requires an input-adaptive mechanism that can decide which experts to use and when to combine them.

Motivated by these observations, we recast geometry-adaptive depth estimation as a sample-wise solution selection problem. Instead of seeking a single estimator or applying a fixed fusion rule, the goal is to decide for each input which depth expert(s) to trust, whether fusion is beneficial, and how much computation is justified. We propose **DepthAgent**, an agentic framework for geometry-adaptive monocular depth estimation. DepthAgent treats existing depth models as frozen tools and uses a Vision-Language Model (VLM) agent as a learned controller for sample-wise expert selection. The agent reasons over scene and camera cues, performs multi-turn tool use, and finally selects or fuses expert predictions, which are difficult to capture with a fixed hand-crafted router. Since the desired behavior involves discrete expert selection while the final objective is dense geometric quality, we optimize the agent with reinforcement fine-tuning based on GRPO (Shao et al., 2024), avoiding fixed oracle trajectories while enabling dynamic control over the performance-efficiency trade-off. We further design a **multi-reward scheme** that jointly encourages valid tool execution, camera- and scene-aware reasoning, empirically strong expert selection, and computation-aware efficiency. Experiments across perspective, fisheye, and panoramic benchmarks show consistent gains over individual experts, static fusion baselines, and rule-based and learned routers, especially on challenging samples where individual predictions are often unreliable. Our contributions are summarized as follows:

- We conduct a systematic analysis of single-model selection and multi-model fusion for monocular depth estimation across perspective and panoramic scenarios, revealing strong model complementarity and camera-dependent expert preference.

- We formulate geometry-adaptive depth estimation as a sample-wise solution selection problem and propose DepthAgent, a VLM-based agent that adaptively selects depth experts and fusion strategies for each sample.
- We design a multi-reward reinforcement fine-tuning scheme that encourages valid tool execution, camera- and scene-aware decision-making, and quality-efficiency balancing.
- We validate DepthAgent on diverse depth benchmarks, showing consistent improvements over individual experts, non-agentic fusion baselines, and rule-based and learned routers, with especially strong gains on hard samples.

2 Related Works

Monocular Depth Estimation. Early monocular depth estimation focused on perspective images, improving generalization with large-scale labeled and unlabeled data (Ranftl et al., 2020; Yin et al., 2021; Yang et al., 2024a;b) and diffusion priors (Ke et al., 2024; Rombach et al., 2022). Recent zero-shot metric methods further recover scale using camera parameters through supervision or canonical parameterization (Guizilini et al., 2023; Hu et al., 2024; Yin et al., 2023; Piccinelli et al., 2024; 2025b). Large-FoV methods extend to fisheye and 360° images to exploit wider context while handling severe distortions (Jiang et al., 2021; Li et al., 2022; Shen et al., 2022; Yun et al., 2023; Ai et al., 2023). However, limited large-FoV data and strong in-domain assumptions still hinder cross-camera generalization, despite distortion-aware and unified cross-camera modeling (Su & Grauman, 2017; Zhu et al., 2019; Xiong et al., 2024; Rey-Area et al., 2022; Feng et al., 2023; Guo et al., 2025b; Ganesan et al., 2026; Piccinelli et al., 2025a). This highlights the importance of input-adaptive expert selection for robust real-world deployment, motivating DepthAgent to choose the most suitable expert(s) for each input.

Model Fusion and Expert Selection. Expert fusion and dynamic expert selection have been widely studied in language, vision, and multimodal systems (Ong et al., 2024; Lu et al., 2024; Chen et al., 2023; Wu et al., 2025; Zong et al., 2024; Ye & Xu, 2023; Li et al., 2025; Zuo et al., 2025; Zhu et al., 2026a). However, these ideas remain relatively underexplored in depth estimation, with only limited studies considering prediction fusion or complementary cues (Ren et al., 2022; Xia et al., 2020; Dai et al., 2023), while DepthFusion (Obukhov et al., 2025) further highlights the potential of variance-aware expert selection. In contrast, we systematically investigate sample-wise model selection and fusion for depth estimation, demonstrating that adaptive expert choice and fusion can consistently improve performance across diverse settings, especially on challenging samples.

3 Method Overview

We first conduct fusion analysis to understand when different depth experts or fusion solutions succeed or fail. These findings motivate the rest of our design. Based on them, we introduce DepthAgent, which learns to dynamically select suitable solutions for each input image. Finally, we derive the reward functions from the same empirical observations, encouraging the agent to favor solutions that yield better depth quality across different scenarios.

3.1 Fusion Analysis

Analysis setup. We first specify the candidate experts, fusion strategies, and evaluation protocol used in our fusion analysis. We consider perspective-oriented experts, including UniDepth, Metric3D, and Metric3Dv2, and ERP-trained experts, including UniDAC and UniK3D. Besides individual experts, we evaluate pixel-wise mean, max, and min fusion (Zhang et al., 2020; Ma et al., 2022), as well as DepthFusion (details in Appendix D). For each sample, a *solution* refers to either a single expert or a combination of multiple experts with a fusion strategy. *Oracle solution* is defined as the candidate solution that achieves the best δ_1, providing an empirical upper bound for sample-wise selection and fusion. We conduct the analysis on perspective datasets, including A2D2 (Geyer et al., 2020), DDAD (Guizilini et al., 2020), HM3D (Ramakrishnan et al., 2021), and Taskonomy (Zamir et al., 2018), together with their ERP variants, constructed by transforming images into ERP patches, since ERP data are relatively scarce (details in Appendix). To cover native ERP inputs, we include ScanNet++ (Yeshwanth et al., 2023), Matterport3D (Chang et al., 2017), and Pano3D-GV2 (Albanis

Table 1: **Model-family preference across camera-domain groups.** Best-single columns indicate the percentage of samples for which a Perspective or ERP model achieves the highest single-model performance. Oracle-presence columns indicate the percentage of samples in which at least one model from the corresponding family is selected by the oracle solution. Avg. gain denotes the average performance improvement achieved by fusion over the corresponding best single model across the evaluated perspective and ERP models.

Dataset group	Best single model		Oracle solution presence		Avg. gain
	Perspective	ERP	Perspective	ERP	$\Delta\delta_1$
Perspective	80.1	19.9	99.0	58.8	0.019
ERP variant	77.1	22.9	94.8	52.1	0.016
Native ERP	2.2	97.8	36.5	98.2	0.044

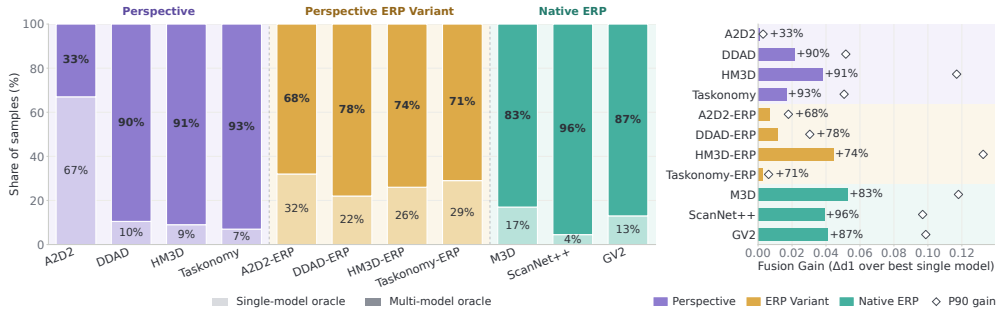


Figure 2: **Fusion consistently outperforms single models. Left:** Dataset-wise oracle proportions of single-model vs. multi-model solutions. **Right:** Fusion Gain against best single model. Bars and diamonds denote the mean and 90th percentile of per-sample δ_1 fusion gain over the best single model; annotations show the fraction of samples where fusion achieves higher δ_1.

et al., 2021). For each dataset, we randomly select 200 training samples for analysis, ensuring that no test-set bias is introduced.

Camera geometry is the primary factor in model preference. Tab. 1 shows that model preference is strongly tied to camera geometry. Perspective models dominate perspective datasets, winning 80.1% of samples, and remain preferred on ERP variants. This suggests that ERP reprojection of perspective images does not fully reproduce the distortions and model preferences of native panoramic data. In contrast, native ERP datasets show a clear reversal, where ERP-trained models win 97.8% of samples. Oracle solutions further reveal cross-family complementarity, with the dominant model family varying with the input camera type. Yet camera type alone does not determine the best solution: even on perspective data, the oracle solution includes an ERP-trained model for 58.8% of the samples. Therefore, *model selection should depend on the camera type, but still be made per sample.*

Fusion outperforms single models on most samples. In Fig. 2 (left), the proportion of oracle solutions that contain multiple models could reach 96% on ScanNet++ and 93% on Taskonomy, while the remaining samples are best handled by a single model. Across all 11 datasets, the oracle solutions are multi-model for 78.6% of samples. Fig. 2 (right) shows the average δ_1 improvement of the oracle solution compared with the best single model on each dataset. Fusion improves performance to varying degrees across all datasets. In addition to the mean gain, we report the P90 gain, defined as the 90th percentile of per-sample $\Delta\delta_1$, to characterize the upper-end improvement achieved on samples where fusion is particularly beneficial. The advantage is most pronounced on native ERP datasets: fusion achieves mean gains of $\Delta\delta_1 = 0.053$ on Matterport3D and 0.041 on Pano3D-GV2, with 83% and 87%of samples benefiting from fusion, respectively.

Fusion yields larger gains on difficult samples. Fig. 3 groups samples by the best single-model δ_1 within each dataset family, where Q1 contains the hardest samples and Q5 the easiest. In all three settings, the largest mean fusion gains appear in the hardest quintile and steadily diminish as single-model performance improves. The negative Pearson correlations further confirm that fusion is most useful precisely when the strongest individual model is insufficient. This pattern is especially pronounced in the ERP setup, where difficult samples are more frequent, making fusion a targeted correction rather than a uniform boost. This finding suggests that *fusion gains are strongly correlated with sample difficulty, rather than being uniformly distributed across samples.*

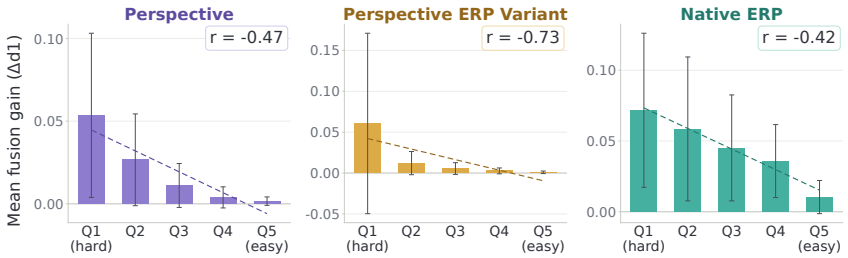


Figure 3: **Difficulty-dependent fusion gain.** Mean fusion gain ($\Delta\delta_1 \pm \sigma$) is shown across best-single δ_1 quintiles within each dataset group, where Q1/Q5 denote the hardest/easiest samples. Fusion gains are largest on hard samples (the strongest individual model performs poorly). Dashed lines indicate linear trends over quintile means, while Pearson r is computed over pooled per-sample pairs.

These analyses motivate DepthAgent: the best solution is input-dependent, governed by camera geometry and sample difficulty. Thus, fusion should be a per-sample decision over when to fuse, which experts to trust, and how to combine them. We formulate this as an agent-driven process, where a VLM selects the most suitable solution for each input.

3.2 DepthAgent

The overall framework of DepthAgent is illustrated in Fig. 4. Given an input image x, DepthAgent formulates depth estimation as an agentic process that adaptively coordinates multiple depth experts to produce the final depth solution. Since expert selection depends on high-level cues such as scene content, camera geometry, projection distortion, and intermediate prediction consistency, which are difficult to encode with a fixed hand-crafted router (Sec. 4), DepthAgent leverages a VLM agent to jointly reason about expert suitability and dynamically adjust tool usage for each sample.

Feature analysis. Before invoking any depth expert, the agent first considers the visual and geometric properties of the input image. This includes scene-level cues, such as whether the image is indoor or outdoor, and camera-related cues, such as intrinsic geometry, projection characteristics, and extracted depth-related features. These observations help the agent identify the camera type of x and which depth expert is more suitable for the current sample.

Depth expert pool and tool selection. Based on the analyzed features, the agent selects depth experts from a heterogeneous depth expert pool. The pool contains both perspective models and ERP-trained models, enabling DepthAgent to handle standard pinhole images as well as 360°, fisheye, and other geometrically distorted inputs. At each decision step, DepthAgent chooses an expert a_t according to the current image understanding and previously observed tool outputs, instead of relying on a fixed model or a predefined fusion rule. The pool can be changed without retraining (Appendix E).

Multi-turn tool execution. After a successful tool call, the selected expert returns a predicted depth map $\mathbf{D}_{a_t}$ together with auxiliary depth features f_{a_t}, such as average depth distance and depth variation. The depth map serves as a candidate solution, while the auxiliary features provide a compact summary of the prediction quality and geometric behavior, making the tool result easier for the agent to interpret. Conditioned on the current observation and previous tool results, the agent decides whether to invoke another expert, compare complementary predictions, or terminate with a final answer. This ReAct-style (Yao et al., 2023) multi-turn process allows DepthAgent to adaptively explore different depth experts and construct an appropriate depth solution for each sample.

Agent optimization. We use GRPO (Shao et al., 2024) instead of SFT because fixed oracle trajectories may become invalid when the expert pool changes, and offer limited flexibility in dynamically balancing tool call efficiency and depth performance. During training, we sample G rollout responses $\{o_1, o_2, \ldots, o_G\}$ for each input, where o_i records the agent's reasoning, tool-use trajectory, selected solution (expert combination and fusion strategy), and corresponding final depth map. Each rollout is evaluated using rewards that consider both the final depth prediction and the intermediate decision process, including format correctness, tool success, selection behavior, and efficiency. The resulting advantages $\{A^1, A^2, \ldots, A^G\}$ are then used to update the agent, enabling DepthAgent to learn how to select suitable depth tools through interaction. Furthermore, we apply multi-reward normalization (Zhu et al., 2026b; Liu et al., 2026) in the advantage calculation. Specifically, given K reward

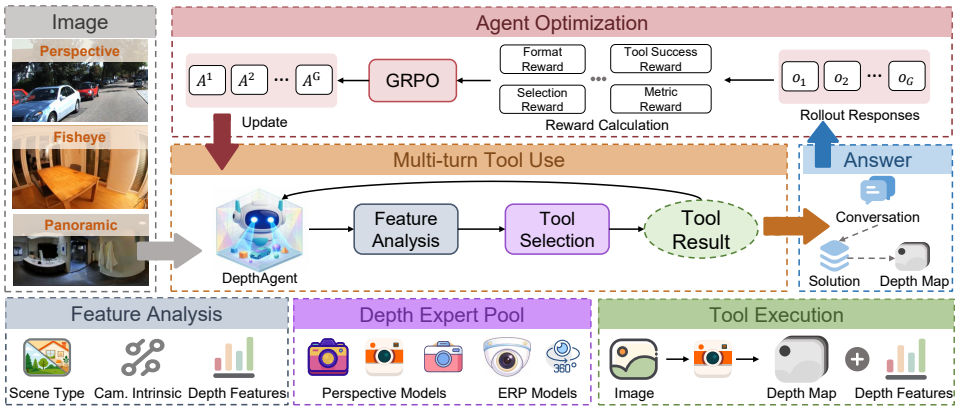


Figure 4: **Overview of DepthAgent.** Given an input image, DepthAgent analyzes the scene type and camera intrinsics, and interactively selects depth experts from a depth-expert tool pool. Each tool call returns a depth prediction along with auxiliary depth features. Based on the tool results, DepthAgent determines whether to continue exploration or produce the final solution, which includes the final depth map generated by the solution. During training, rollout conversations are optimized with GRPO using the proposed multi-reward objective.

functions $r^i_{(1)}, r^i_{(2)}, \ldots, r^i_{(K)}$ for the i-th response o_i, we compute the normalized advantage for each reward and aggregate as:

$$A^i_{(k)} = \frac{r^i_{(k)} - \text{mean}(r^1_{(k)}, \ldots, r^G_{(k)})}{\text{std}(r^1_{(k)}, \ldots, r^G_{(k)})}, A^i = \sum_{k=1}^{K} A^i_{(k)}. \tag{1}$$

3.2.1 Reward Functions

In addition to traditional rewards like format reward and tool success reward (Shao et al., 2024; Liu et al., 2025c; Zhu et al., 2026a), we further design three reward functions: scene awareness reward R_{scene}, selection prior reward R_{sel}, and efficiency-aware metric reward R_{em}. Additional details of other reward functions are provided in the Appendix.

Scene awareness reward. Correct model selection requires the agent to understand the scene context and infer the camera geometry from the input image. We introduce a scene awareness reward to encourage the agent to first analyze whether the scene is indoor or outdoor and to estimate the camera type before selecting tools. For each sample, the Depth Agent first invokes the `estimate_camera_type` tool to classify the camera as standard perspective, atypical perspective, fisheye, or ERP from camera information. This reward is implemented as a lightweight multi-label matching criterion over the first reasoning block, based on correctly identified scene category and camera type. The reward equals the average accuracy over these two labels.

Selection prior reward. The selection prior reward R_{sel} encodes our prior knowledge about expert suitability, as observed in Sec 3.1. It gives a mild reward when the selected experts match the input domain: perspective-oriented experts for perspective inputs, and ERP-trained experts for panoramic or large-FoV inputs. This provides soft guidance for faster policy adaptation in training. R_{sel} is set to 1.0 if the selected experts include at least one expert matching the input domain, and 0.0 otherwise.

Efficiency-aware metric reward. Although more tool calls may improve the final prediction, the marginal gain is not always worth the additional computation. We therefore introduce an efficiency-aware metric reward R_{em} that jointly considers the metric improvement and the number of model calls. Given a solution y_i from o_i, let m_i denote its task metric score and n_i denote the number of models in the solution. y_{ref} and m_{ref} denote the per-sample oracle among the 83 candidate solutions and its metric score, precomputed offline (Appendix C.1). We define the normalized metric gap and efficiency gap:

$$\Delta m_i = \frac{m_i - m_{\text{ref}}}{|m_{\text{ref}}| + \epsilon}, \qquad \Delta n_i = \frac{n_i - n_{\text{ref}}}{n_{\max}}. \tag{2}$$

The efficiency-aware reward is then computed as

$$R_{\text{em}}(y_i, y_{\text{ref}}) = \Delta m_i - \lambda \Delta n_i \exp\left(-\frac{|\Delta m_i|}{\tau}\right), \tag{3}$$

where λ controls the strength of the efficiency penalty, τ controls how strongly the penalty depends on the metric gap, and ϵ is used for numerical stability. This reward discourages extra model calls when the metric improvement is small, while still allowing higher computation when it leads to a meaningful performance gain.

4 EXPERIMENTS

Datasets. We train on indoor perspective and ERP variant from HM3D, outdoor perspective data from DDAD and A2D2, and panoramic data from Matterport3D. We evaluate on six diverse benchmarks across three camera domains: perspective datasets, including KITTI (Geiger et al., 2012), NYU-v2 (Nathan Silberman & Fergus, 2012), and IBims-1 (Koch et al., 2020); fisheye datasets ScanNet++; and panoramic / 360° datasets, including Matterport3D and Pano3D-GV2. We follow prior works (Ganesan et al., 2026; Guo et al., 2025b; Piccinelli et al., 2025a) to report the percentage of inliers (δ_i) under thresholds of 1.25^i, absolute relative error (A.Rel), and root mean squared error (RMSE). ERP models are evaluated against ERP-transformed ground truth.

Baselines. We compare DepthAgent against the **depth experts in the tool pool** and **fusion strategies** described in Sec. 3.1. We further compare with routing baselines that select from the same 83 candidate solutions (details in Appendix D): (i) **Rule-based router** estimates the camera type with the same `estimate_camera_type` tool used by DepthAgent and applies a fixed solution per camera type, chosen as the candidate with the best average δ_1 on the training images of that type. Since this per-domain configuration is found by exhaustive search on the training data, it is a strong instance of domain-level routing. (ii) **MLP router** predicts the solution from frozen CLIP image features. (iii) **Metadata router** predicts the solution from the camera cues available to DepthAgent, *i.e.*, camera intrinsics, image size, and the output of `estimate_camera_type`, without image features. (iv) **Hybrid router** takes the concatenation of the inputs of (ii) and (iii). All learned routers are trained on the same 1,000 images as DepthAgent.

Implementation. To reduce computational overhead, DepthAgent is built on Qwen2.5-VL-3B (Bai et al., 2025) and fine-tuned with GRPO (Shao et al., 2024). We train the agent on 1,000 images for 300 steps with a maximum of 5 interaction turns. The references in R_{em} are precomputed on these images in about 47 GPU-min (Appendix C.1). λ is set to 1 and 0.2, and τ is set to 0.1 and 3.4 for perspective and ERP inputs, respectively. Training is performed on 4 H100 GPUs with a batch size of 4. During inference, we use greedy decoding to ensure reproducibility. The prompt, tool schema, GRPO settings, reward details, and conversation examples are provided in the Appendix.

4.1 EXPERIMENTAL RESULTS

Comparison with baselines. Tab. 2 shows that DepthAgent consistently outperforms individual experts and non-agentic fusion baselines across camera domains, achieving the best performance on five datasets as well as on average. Although UniK3D is a strong universal-camera expert, DepthAgent significantly improves A.Rel and RMSE. Compared with fusion strategies, DepthAgent shows stronger robustness, underscoring the importance of sample-wise solution selection. DepthAgent also outperforms all routing baselines while calling fewer experts. Even the rule-based router, which applies the best per-domain solution found by exhaustive search, falls behind UniK3D in δ_1 and A.Rel, and learned routers given the same camera cues do not close the gap. Hence, the gain of DepthAgent stems from its sample-wise policy rather than domain-level routing or extra input information (Appendix E). Notably, the improvements are most evident in RMSE, which penalizes large depth errors more heavily and is thus crucial for evaluating geometric fidelity and robustness in depth estimation. Although DepthAgent still falls short of the upper bound, it consistently outperforms the baselines by reducing the negative impact of suboptimal fusion. Overall, these results suggest that *fusion is effective only when appropriate and complementary experts are selected.*

Comparison on hard samples. Hard samples are critical because they often correspond to failure cases in real-world deployment, where improving robustness is more important than average-case performance. We further evaluate DepthAgent on hard samples, defined as the worst 10% samples ranked by the δ_1 score of the best single model on each dataset. As shown in Tab. 3, DepthAgent consistently outperforms all baseline experts on perspective, fisheye, and 360° data. The improve-

Table 2: **Quantitative comparison.** Perspective reports the average performance over KITTI, NYU-v2, and IBims-1. Fisheye reports performance on ScanNet++, while panoramic reports the average performance over Matterport3D and Pano3D-GV2. $\bar{n}$: average number of expert calls per sample. ‡: best solution per sample.

Method	Perspective			Fisheye			Panoramic			$\bar{n}$
	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	
UniDepth	0.700	0.203	1.359	0.135	0.573	1.029	0.415	0.314	0.898	1.0
Metric3D	0.850	0.136	1.279	0.727	0.162	0.450	0.403	0.290	0.721	1.0
Metric3Dv2	0.864	0.124	1.193	0.745	0.170	0.474	0.422	0.300	0.847	1.0
UniDAC	0.845	0.140	1.876	0.918	0.097	0.279	0.755	0.169	0.472	1.0
UniK3D	0.937	0.091	1.261	0.927	0.103	0.287	**0.820**	0.152	0.383	1.0
Mean	0.890	0.110	1.045	0.840	0.139	0.313	0.620	0.199	0.529	5.0
Max	0.590	0.285	2.276	0.139	0.573	1.023	0.735	0.235	0.500	5.0
Min	0.912	0.095	1.258	0.663	0.190	0.573	0.266	0.354	1.091	5.0
DepthFusion	0.878	0.117	1.195	0.621	0.252	0.623	0.378	0.327	1.000	5.0
Rule-based router	0.852	0.139	1.255	0.862	0.128	0.301	0.735	0.190	0.472	2.3
MLP router	0.904	0.107	1.328	0.187	1.014	2.547	0.744	0.182	0.473	2.9
Metadata router	0.814	0.164	1.404	0.851	0.133	0.308	0.750	0.180	0.480	2.2
Hybrid router	0.889	0.113	1.288	0.239	0.519	0.938	0.739	0.209	0.491	3.0
DepthAgent	**0.948**	**0.070**	**0.918**	**0.946**	**0.084**	**0.254**	0.819	**0.145**	**0.365**	1.9
Upper-bound‡	0.985	0.057	0.844	0.967	0.080	0.239	0.860	0.132	0.360	–

Table 3: **Comparison on hard samples.** We report performance on hard samples, defined as the top 10% worst-performing samples of the best single model. Perspective reports the average performance over KITTI, NYU-v2, and IBims-1. Fisheye reports performance on ScanNet++, while panoramic reports the average performance over Matterport3D and Pano3D-GV2.

Method	Perspective			Fisheye			Panoramic		
	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.650	0.271	2.045	0.197	0.610	0.939	0.337	0.496	0.853
Metric3D	0.681	0.220	2.311	0.635	0.203	0.629	0.346	0.372	0.847
Metric3Dv2	0.737	0.212	2.072	0.659	0.204	0.593	0.292	0.427	0.919
UniDAC	0.754	0.177	2.286	0.789	0.163	0.434	0.547	0.273	0.525
UniK3D	0.794	0.155	1.950	0.799	0.156	0.434	0.560	0.261	0.466
DepthAgent	**0.833**	**0.115**	**1.658**	**0.817**	**0.144**	**0.410**	**0.587**	**0.253**	**0.450**

ments are more pronounced than on the full test sets, especially in δ_1, indicating that DepthAgent is particularly effective at selecting reliable experts for challenging cases.

Efficiency of DepthAgent. To improve efficiency, we introduce a Fast mode that skips CoT reasoning and directly selects tools before producing the final answer, while retaining the same selected solution as CoT mode. On an H100, DepthAgent runs end to end in under one second per sample in Fast mode (0.98 s; Appendix Tab. 15), on the same order as all five experts (0.51 s), and 1.97 s in CoT mode. Unlike exhaustive fusion (Tab. 2), it calls only 1.9 experts and avoids unreliable predictions.

Robustness to expert-pool changes. We modify the expert pool at inference without retraining. When an expert is removed, DepthAgent makes no call to it and still outperforms re-fitted baselines. When an expert is replaced by an unseen one, DepthAgent adopts the new tool from its description alone and makes no call to the replaced one (Appendix E).

4.2 Ablation Studies

Effects of multi-reward. We ablate different reward components on all datasets, as shown in Tab. 4a. The results suggest that relying on a partial reward is insufficient for robust solution selection. Combining R_{scene}, R_{sel}, and R_{em} achieves the best overall performance, indicating that scene understanding, selection prior, and efficiency-aware metric optimization are complementary.

Effects of λ, τ, and $n_{\max}$. Tab. 4b studies the effects of the reward hyperparameters in R_{em}. Overall, $n_{\max}$ mainly determines the tool-call budget and directly affects the average tool calls. Under a given budget, λ modulates the penalty for tool usage, while τ controls the tolerance to metric differences among candidate solutions. This suggests that a proper λ could balance performance and tool cost. Together, they shape the reward sensitivity and influence whether the agent favors compact or more diverse solution compositions. The results indicate that robust performance comes from a well-calibrated reward design, rather than simply increasing the number of tool calls.

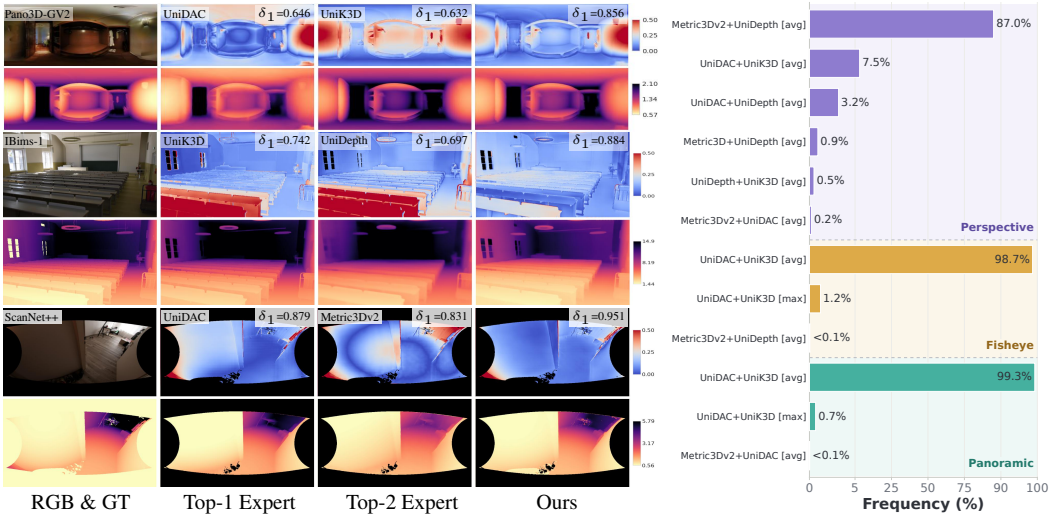
Table 4: **Ablation study of the reward design.** Combining all rewards improves overall performance, while calibrated hyperparameters drive results beyond tool usage alone. $\bar{n}$: average number of tool calls.

(a) Leave-one-out reward ablation.

R_{scene}	R_{sel}	R_{em}	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
✓	✓	–	0.832	0.146	0.827
✓	–	✓	0.871	0.134	0.650
–	✓	✓	0.891	0.102	0.639
✓	✓	✓	0.905	0.097	0.623

(b) Effect of reward hyperparameters in R_{em}.

λ	τ	$n_{\max}$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\bar{n}$
0.4	1.0	1.0	0.860	0.119	0.576	1.0
0.1	2.0	2.0	0.764	0.166	0.678	3.2
0.4	3.4	2.0	0.828	0.143	0.620	2.5
0.2	3.4	2.0	0.905	0.097	0.623	1.9



(a) **Depth Map Comparison.** Odd rows show the error maps, while even rows show the predicted depth maps. For each sample, we compare the top-2 expert predictions with the final solution of DepthAgent. DepthAgent generates more accurate depth maps with lower errors.

(b) **Solution frequency.** Results are grouped by projection type, with each group reporting the top selected solution configurations across test samples.

Figure 5: **Analysis of DepthAgent behavior.**

4.3 Qualitative Results

Depthmap comparison. Fig. 5a compares DepthAgent with the top-2 individual experts on representative samples. DepthAgent produces more faithful depth structures and consistently lower error maps by selecting or combining complementary expert solutions. Additional visualizations are provided in the Appendix. Zoom in for better effects.

Solution distribution on different scenarios. Fig. 5b shows that the selection of DepthAgent is camera-dependent but not reducible to domain routing. On perspective data, the policy does not collapse to the strongest average expert UniK3D: samples are often routed to Metric3Dv2/UniDepth-based combinations. The concentrated distributions on fisheye and 360° data reflect the large gap among experts in these domains (Tab. 2): perspective experts degrade severely under strong distortion and spherical geometry, so the reward consistently favors ERP-trained experts. This concentration is thus reward-optimal rather than a loss of adaptivity.

Conversation. We provide representative visualization conversation examples in the Appendix F.

5 Conclusion

We presented **DepthAgent**, an agentic framework for geometry-adaptive monocular depth estimation that performs sample-wise expert selection and fusion. Motivated by our analysis showing the benefits of fusion and camera-dependent expert preference, DepthAgent uses a VLM agent to reason about scene and camera cues, invoke frozen depth experts, and adaptively perform the final solution. With multi-reward reinforcement fine-tuning, DepthAgent consistently improves over fusion and routing baselines across perspective, fisheye, and panoramic benchmarks, especially on challenging samples. These results highlight the importance of model selection and fusion, moving beyond fixed depth estimators toward input-adaptive expert selection for systematic depth estimation.

Reproducibility Statement

Sec. 3 describes the agent design, tool interface, and reward functions, and Sec. 4 specifies the base VLM, GRPO training configuration, hyperparameters (λ, τ, $n_{\max}$), and greedy decoding used for inference. The appendix provides the training and testing dataset statistics, full reward definitions and reference-solution construction (App. C.1), the prediction alignment pipeline, implementations of DepthFusion and all router baselines (App. D), the complete prompts and tool definitions, and the latency measurement protocol. All depth experts and datasets used in this work are publicly available. The code, trained model, and precomputed reference solutions will be released upon publication.

AI Use Statement

We used LLMs to improve the clarity and readability of the manuscript and to assist in identifying relevant literature. All AI-assisted content and retrieved references were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

References

Hao Ai, Zidong Cao, Yan-Pei Cao, Ying Shan, and Lin Wang. Hrdfuse: Monocular 360° depth estimation by collaboratively learning holistic-with-regional depth distributions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, 2023.

Georgios Albanis, Nikolaos Zioulis, Petros Drakoulis, Vasileios Gkitsas, Vladimiros Sterzentsenko, Federico Alvarez, Dimitrios Zarpalas, and Petros Daras. Pano3d: A holistic benchmark and a solid baseline for 360deg depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3727–3737, 2021.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

Boyu Chen, Zhengrong Yue, Siran Chen, Zikang Wang, Yang Liu, Peng Li, and Yali Wang. Lvagent: Long video understanding by multi-round dynamical collaboration of mllm agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025a.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Xiaxu Chen, Wei Li, Chunxu Liu, Chi Xie, Xiaoyan Hu, Chengqian Ma, Feng Zhu, and Rui Zhao. On the suitability of reinforcement fine-tuning to visual tasks. In *CVPR*, 2025b.

Yaqiao Dai, Renjiao Yi, Chenyang Zhu, Hongjun He, and Kai Xu. Multi-resolution monocular depth map fusion by self-supervised gradient-based composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Xingshuai Dong, Matthew A Garratt, Sreenatha G Anavatti, and Hussein A Abbass. Towards real-time monocular depth estimation for robotics: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):16940–16961, 2022.

Ruofei Du, Eric Turner, Maksym Dzitsiuk, Luca Prasso, Ivo Duarte, Jason Dourgarian, Joao Afonso, Jose Pascoal, Josh Gladstone, Nuno Cruces, et al. Depthlab: Real-time 3d interaction with depth maps for mobile augmented reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pp. 829–843, 2020.

Hao Feng, Wendi Wang, Jiajun Deng, Wengang Zhou, Li Li, and Houqiang Li. Simfir: A simple framework for fisheye image rectification with self-supervised representation learning. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, 2023.

Girish Chandar Ganesan, Yuliang Guo, Liu Ren, and Xiaoming Liu. Unidac: Universal metric depth estimation for any camera. In *CVPR*, 2026.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, et al. A2d2: Audi autonomous driving dataset. *arXiv preprint arXiv:2004.06320*, 2020.

Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Vitor Guizilini, Igor Vasiljevic, Dian Chen, Rareș Ambruș, and Adrien Gaidon. Towards zero-shot scale-aware monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9233–9243, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Yuliang Guo, Sparsh Garg, S Mahdi H Miangoleh, Xinyu Huang, and Liu Ren. Depth any camera: Zero-shot metric depth estimation from any camera. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 26996–27006, 2025b.

Dong Han, Beni Mulyana, Vladimir Stankovic, and Samuel Cheng. A survey on deep reinforcement learning algorithms for robotic manipulation. *Sensors*, 2023.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 2022.

Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10579–10596, 2024.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Hualie Jiang, Zhe Sheng, Siyu Zhu, Zilong Dong, and Rui Huang. Unifuse: Unidirectional fusion for 360 panorama depth estimation. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):1519–1526, 2021.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 1996.

Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Tobias Koch, Lukas Liebel, Marco Körner, and Friedrich Fraundorfer. Comparison of monocular depth estimation methods using geometrically relevant metrics on the IBims-1 dataset. *Computer Vision and Image Understanding (CVIU)*, 191:102877, 2020. doi: 10.1016/j.cviu.2019.102877.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, 2023.

Sijie Li, Chen Chen, and Jungong Han. Simmlm: A simple framework for multi-modal learning with missing modality. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 24068–24077, 2025.

Yuyan Li, Yuliang Guo, Zhixin Yan, Xinyu Huang, Ye Duan, and Liu Ren. Omnifusion: 360 monocular depth estimation via geometry-aware fusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, 2022.

Runtao Liu, Ziyi Liu, Jiaqi Tang, Yue Ma, Renjie Pi, Jipeng Zhang, and Qifeng Chen. Longvideoagent: Multi-agent reasoning with long videos. *arXiv preprint arXiv:2512.20618*, 2025a.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242*, 2026.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. Visual-rft: Visual reinforcement fine-tuning. *ICCV*, 2025c.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2024.

Jiayi Ma, Linfeng Tang, Fan Fan, Jun Huang, Xiaoguang Mei, and Yong Ma. Swinfusion: Cross-domain long-range learning for general image fusion via swin transformer. *IEEE/CAA Journal of Automatica Sinica*, 2022.

Yutaka Matsuo, Yann LeCun, Maneesh Sahani, Doina Precup, David Silver, Masashi Sugiyama, Eiji Uchibe, and Jun Morimoto. Deep learning, reinforcement learning, and world models. *Neural Networks*, 2022.

Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *The European Conference on Computer Vision (ECCV)*, 2012.

Anton Obukhov, Matteo Poggi, Fabio Tosi, Ripudaman Singh Arora, Jaime Spencer, Chris Russel, Simon Hadfield, Richard Bowden, Shuaihang Wang, Zhenxin Ma, et al. The fourth monocular depth estimation challenge. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*, 2024.

Yingzhe Peng, Gongrui Zhang, Miaosen Zhang, Zhiyuan You, Jie Liu, Qipeng Zhu, Kai Yang, Xingzhong Xu, Xin Geng, and Xu Yang. Lmm-r1: Empowering 3b lmms with strong reasoning abilities through two-stage rule-based rl. *arXiv preprint arXiv:2503.07536*, 2025.

Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10106–10116, 2024.

Luigi Piccinelli, Christos Sakaridis, Mattia Segu, Yung-Hsu Yang, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unik3d: Universal camera monocular 3d estimation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1028–1039, 2025a.

Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unidepthv2: Universal monocular metric depth estimation made simpler. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021.

René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020.

Weisong Ren, Lijun Wang, Yongri Piao, Miao Zhang, Huchuan Lu, and Ting Liu. Adaptive co-teaching for unsupervised monocular depth estimation. In *European Conference on Computer Vision*, pp. 89–105. Springer, 2022.

Manuel Rey-Area, Mingze Yuan, and Christian Richardt. 360monodepth: High-resolution 360 monocular depth estimation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3752–3762. IEEE, 2022.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, et al. Vlm-r1: A stable and generalizable r1-style large vision-language model. *arXiv preprint arXiv:2504.07615*, 2025.

Zhijie Shen, Chunyu Lin, Kang Liao, Lang Nie, Zishuo Zheng, and Yao Zhao. Panoformer: panorama transformer for indoor 360° depth estimation. In *European Conference on Computer Vision*, pp. 195–211. Springer, 2022.

Yu-Chuan Su and Kristen Grauman. Learning spherical convolution for fast features from 360° imagery. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 2017.

Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liang-Yan Gui, Yu-Xiong Wang, Yiming Yang, et al. Aligning large multimodal models with factually augmented rlhf. *arXiv preprint arXiv:2309.14525*, 2023.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

Huajie Tan, Yuheng Ji, Xiaoshuai Hao, Minglan Lin, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. Reason-rft: Reinforcement fine-tuning for visual reasoning. *arXiv preprint arXiv:2503.20752*, 2025.

Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8445–8453, 2019.

Yuanchen Wu, Junlong Du, Ke Yan, Shouhong Ding, and Xiaoqiang Li. Tove: Efficient vision-language learning via knowledge transfer from vision experts. *arXiv preprint arXiv:2504.00691*, 2025.

Zhihao Xia, Patrick Sullivan, and Ayan Chakrabarti. Generating and exploiting probabilistic monocular depth estimates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 65–74, 2020.

Yuwen Xiong, Zhiqi Li, Yuntao Chen, Feng Wang, Xizhou Zhu, Jiapeng Luo, Wenhai Wang, Tong Lu, Hongsheng Li, Yu Qiao, Lewei Lu, Jie Zhou, and Jifeng Dai. Efficient deformable convnets: Rethinking dynamic and sparse operator for vision applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10371–10381, 2024a.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024b.

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, et al. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. *arXiv preprint arXiv:2503.10615*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

Hanrong Ye and Dan Xu. Taskexpert: Dynamically assembling multi-task representations with memorial mixture-of-experts. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 21828–21837, 2023.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–22, 2023.

Wei Yin, Jianming Zhang, Oliver Wang, Simon Niklaus, Long Mai, Simon Chen, and Chunhua Shen. Learning to recover 3d scene shape from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 204–213, 2021.

Wei Yin, Chi Zhang, Hao Chen, Zhipeng Cai, Gang Yu, Kaixuan Wang, Xiaozhi Chen, and Chunhua Shen. Metric3d: Towards zero-shot metric 3d prediction from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9043–9053, 2023.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Ilwi Yun, Chanyong Shin, Hyunku Lee, Hyuk-Jae Lee, and Chae-Eun Rhee. Egformer: Equirectangular geometry-biased transformer for 360 depth estimation. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, 2023.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3712–3722, 2018.

Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization. *arXiv preprint arXiv:2503.12937*, 2025.

Yu Zhang, Yu Liu, Peng Sun, Han Yan, Xiaolin Zhao, and Li Zhang. Ifcnn: A general image fusion framework based on convolutional neural network. *Information Fusion*, 2020.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Zetong Zhou, Dongping Chen, Zixian Ma, Zhihan Hu, Mingyang Fu, Sinan Wang, Yao Wan, Zhou Zhao, and Ranjay Krishna. Reinforced visual perception with tools. *arXiv preprint arXiv:2509.01656*, 2025.

Jie Zhu, Xiao Guo, Yiyang Su, Anil Jain, and Xiaoming Liu. Fusionagent: A multimodal agent with dynamic model selection for human recognition. In *CVPR*, 2026a.

Jie Zhu, Yiyang Su, and Xiaoming Liu. Can textual reasoning improve the performance of mllms on fine-grained visual classification? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 9401–9411, June 2026b.

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets V2: more deformable, better results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Zhuofan Zong, Bingqi Ma, Dazhong Shen, Guanglu Song, Hao Shao, Dongzhi Jiang, Hongsheng Li, and Yu Liu. Mova: Adapting mixture of vision experts to multimodal context. *Advances in Neural Information Processing Systems*, 2024.

Yushen Zuo, Qi Zheng, Mingyang Wu, Xinrui Jiang, Renjie Li, Jian Wang, Yide Zhang, Gengchen Mai, Lihong V. Wang, James Zou, Xiaoyu Wang, Ming-Hsuan Yang, and Zhengzhong Tu. 4kagent: Agentic any image to 4k super-resolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

A Additional Related Work

Agentic Tool Use and Reinforcement Fine-tuning. Recent studies have advanced multimodal models into agentic systems that can plan, invoke tools, and iteratively refine predictions for complex visual tasks (Zhou et al., 2025; Liu et al., 2025a; Chen et al., 2025a). Meanwhile, Reinforcement Learning (RL) has become an effective paradigm for improving the reasoning and problem-solving abilities of LLMs and VLMs (Sutton et al., 1998; Kaelbling et al., 1996; Matsuo et al., 2022; Han et al., 2023; Guo et al., 2025a; Bai et al., 2022; Sun et al., 2023; Shao et al., 2024; Hui et al., 2024). Recent works further adapt GRPO and its variants (Yu et al., 2025; Liu et al., 2025b; Zheng et al., 2025; Liu et al., 2026) to multimodal tasks with rule-based rewards (Shen et al., 2025; Huang et al., 2025; Liu et al., 2025c; Tan et al., 2025; Zhang et al., 2025; Zhu et al., 2026b; Chen et al., 2025b; Peng et al., 2025; Yang et al., 2025). In contrast, we design selection, efficiency, and scene-awareness rewards to jointly optimize the effectiveness and efficiency of DepthAgent for depth estimation.

B Dataset statistics

Table 5: **Datasets.** List of training and testing datasets with their attributes. [Key: Syn=Synthetic, Rec=Mesh reconstruction]

(a) **Training Datasets.**

Dataset	#Images	Scene	Acquisition
HM3D-tiny (Ramakrishnan et al., 2021)	310K	Indoor	Rec
DDAD (Guizilini et al., 2020)	80K	Outdoor	LiDAR
A2D2 (Geyer et al., 2020)	350K	Outdoor	LiDAR
Matterport3D (Chang et al., 2017)	5K	Indoor	Rec

(b) **Testing Datasets.**

Dataset	Cam.Type	Scene	Acquisition
KITTI (Geiger et al., 2012)	Perspective	Outdoor	Rec
NYU-v2 (Nathan Silberman & Fergus, 2012)	Perspective	Indoor	Rec
IBims-1 (Koch et al., 2020)	Perspective	Indoor	Rec
ScanNet++ (Yeshwanth et al., 2023)	Fisheye	Indoor	Rec
Matterport3D (Chang et al., 2017)	ERP	Indoor	Rec
Pano3D-GV2 (Albanis et al., 2021)	ERP	Indoor	Rec

Statistics of training datasets. Table 5a summarizes the training datasets, covering indoor and outdoor scenes with diverse acquisition methods, including mesh reconstruction and LiDAR. Note that our DepthAgent uses only 200 images per dataset for training and is trained for merely 300 steps, far fewer than the total number of images reported in the table. Since panoramic training data is limited, we include the Matterport3D training split to train DepthAgent on panoramic inputs. This setting remains fair, since Matterport3D is not uniquely exposed to DepthAgent; Metric3Dv2 and UniK3D have also seen Matterport3D during their own training.

Statistics of testing datasets. Table 5b summarizes the testing datasets, spanning perspective, fisheye, and panoramic camera types across indoor and outdoor scenes.

C Additional Methodology

Justification of the usage of GRPO. The desired behavior of DepthAgent involves multi-turn discrete tool use, adaptive stopping, and selection among multiple near-equivalent solutions, making it less suitable to formulate as standard supervised classification. Constructing SFT targets would require committing to a single oracle solution for each sample and regenerating labels whenever the expert pool, fusion operators, or training data change. Moreover, SFT offers limited flexibility in dynamically balancing depth performance and inference efficiency, since different deployment scenarios may prefer different trade-offs and thus require multiple sets of supervised targets. Instead, reinforcement fine-tuning directly optimizes task-level depth quality and efficiency through a configurable reward function, allowing the agent to explore diverse valid solutions while adjusting the performance-efficiency trade-off without regenerating oracle labels.

C.1 Additional Details of Reward Functions.

Details of format reward R_{fmt}. The agent is trained to produce structured responses that separate reasoning, tool calls, and final answers (Shao et al., 2024; Liu et al., 2025c; Zhu et al., 2026a). We follow the multi-turn setting where each assistant's turn must satisfy a predefined structured format, such as `<think>...</think> <answer>...</answer>`. Let C denote the number

of assistant turns in a conversation, and let c_i be the response at the i-th assistant turn. We define $\mathrm{match}(c_i, P) = 1$ if c_i exactly matches the JSON tag pattern <P>...</P>, and 0 otherwise.

The multi-turn format reward is computed as:

$$R_{\mathrm{fmt}}(C) = \frac{1}{C}\sum_{i=1}^{C} \mathrm{cot}(c_i), \tag{4}$$

where

$$\mathrm{cot}(c_i) = \mathrm{match}(c_i, \texttt{think}) \cdot \left(\mathrm{match}(c_i, \texttt{answer}) \oplus \mathrm{match}(c_i, \texttt{tool_call})\right), \tag{5}$$

and $\oplus$ denotes exclusive-or. In addition to the per-turn format reward, we add an independent final-answer validity reward, which is 1 if the solution extracted from the final answer is executable, and 0 otherwise.

Details of tool success reward R_{tool}. This reward assesses whether the agent's tool calls are executable and yield valid results. Each tool call receives a binary success/failure score, and the reward is defined as the success rate across all tool calls in a trajectory. This incentivizes the agent to produce syntactically correct tool inputs.

Reference solutions and supervision cost. The reference in R_{em} is obtained by enumerating all 83 candidate solutions for each training image. Since every candidate is a fusion of the five expert predictions, we run the five experts once per training image and cache their depth maps (0.90 s per image, about 15 GPU-min for all 1,000 training images). Enumerating and evaluating the 83 candidates on the cached maps takes 1.9 s per image on average, *i.e.*, about 32 GPU-min for the whole training set. Hence, all reward signals are built in about 47 GPU-min. Dense ground truth is required only for these 1,000 images (200 per training dataset), orders of magnitude fewer than the training sets of the experts themselves, and no ground truth is needed at inference.

C.2 Group Relative Policy Optimization (GRPO)

In contrast to RL algorithms such as Proximal Policy Optimization (PPO) (Schulman et al., 2017)—which rely on a critic model to assess policy performance, GRPO eliminates the need for the critic model by directly comparing groups of candidate responses. As shown in Fig. 4, for a given input x, GRPO requires the model to sample G diverse responses $\{o_1, o_2, \ldots, o_G\}$ from the current model π_θ and obtains overall rewards $\{r_1, r_2, \ldots, r_G\}$ for o_i based on the reward function $R(x, o_i)$. In our case, it can be formatted as:

$$\begin{aligned} R(x, o_i) = w_{fmt}R_{fmt}(o_i) + w_{tool}R_{tool}(o_i) + w_{scene}R_{scene}(o_i) \\ + w_{sel}R_{sel}(o_i) + w_{em}R_{em}(y_i, y_{\mathrm{ref}}), \end{aligned} \tag{6}$$

where w_{fmt}, w_{tool}, w_{scene}, w_{sel}, and w_{em} are the reward weights for format reward R_{fmt}, tool success reward R_{tool}, scene awareness reward R_{scene}, selection prior reward R_{sel}, and efficiency-aware metric reward R_{em}, respectively. With the group normalization, GRPO encourages the model to sample preferred answers with a higher reward. The model is updated via:

$$\begin{aligned} J_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\mathrm{old}}}(O|q)} \Bigg[\frac{1}{G}\sum_{i=1}^{G} \min\left(r_i(\theta)A_i, \mathrm{clip}\big(r_i(\theta), 1-\varepsilon, 1+\varepsilon\big)A_i\right) \\ - \beta D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}}) \Bigg], \end{aligned} \tag{7}$$

where A_i denotes the advantage of the i-th response, $r_i(\theta)$is the probability ratio between the current policy and the old policy for the i-th rollout:

$$r_i(\theta) = \frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{\mathrm{old}}}(o_i \mid q)}. \tag{8}$$

ε and β are the GRPO clipping hyperparameters and the coefficient weight for controlling the Kullback–Leibler (KL) penalty, respectively. π_{ref} is the reference model.

D Additional Implementation Details

Prediction alignment pipeline. To fuse the predictions of perspective models and ERP models, we need to align their predictions based on the camera type of the input image. As shown in Fig. 7, for a perspective image, we directly feed the input to perspective models, while first transforming it into the ERP patches for ERP models. The ERP predictions are then transformed back to the original perspective view, so that all predictions are aligned in the same image space before fusion. For a fisheye / panoramic image, we directly use ERP models in the native ERP space, while first preprocessing the input into a perspective-compatible format for perspective models. The resulting perspective predictions are then mapped back to the ERP space. In this way, regardless of the input camera type, all expert predictions are converted into a shared target space for fusion, enabling evaluation with the ground truth and valid mask defined in the original image domain.

Details of DepthFusion. The original DepthFusion is designed for fusing two relative-depth predictors, *i.e.*, Marigold and Depth Anything v2. We adapt it to our multi-expert setting, where the selected expert set can contain an arbitrary number of metric depth predictors. Given K selected depth maps $\{D_k\}_{k=1}^{K}$, all predictions are first brought to the same resolution and scale by the model wrappers. We then choose a reference expert D_r to define depth regions. In our implementation, the reference expert is selected with priority UniDepth, Metric3D, Metric3Dv2, and otherwise the first available expert. We normalize the reference depth map to $[0, 255]$ and run Felzenszwalb segmentation on it, producing a set of non-overlapping regions $\mathcal{S} = \{S_j\}$. For each region S_j and each expert k, we estimate a region-level confidence from the inverse intra-region depth variance:

$$\sigma_{k,j}^2 = \mathrm{Var}\left(\{D_k(p) \mid p \in S_j\}\right), \qquad c_{k,j} = \frac{1}{\sigma_{k,j}^2 + \epsilon},$$

where $\epsilon = 10^{-6}$ avoids numerical instability. A lower variance indicates that the expert gives a more locally consistent prediction within the segmented region, and thus receives higher confidence.

Instead of always taking the most confident expert, we use a conservative reference-biased switching rule. For each segment S_j, we keep the reference expert unless another expert is significantly more confident:

$$k_j^\star = \begin{cases} k, & \text{if } c_{k,j} > T c_{r,j} \text{ for some } k \neq r, \\ r, & \text{otherwise,} \end{cases}$$

where $T = 1.2$ in our experiments. The fused depth is then assembled segment by segment:

$$D_{\mathrm{fuse}}(p) = D_{k_j^\star}(p), \qquad p \in S_j.$$

This extension allows DepthFusion to operate over our full expert pool rather than only the original Marigold–Depth Anything v2 pair, while preserving the main intuition of the method: use local depth consistency to decide which expert to trust in each region.

Details of Depth Features. During training, each depth tool call returns a compact set of statistics computed from the predicted depth map. These features are computed over valid pixels only, where a pixel is valid if its depth is finite and positive. Let D denote the predicted depth map and $V = \{D_{ij} \mid D_{ij} > 0,\ D_{ij} \text{ is finite}\}$. If fewer than 100 valid pixels are available, feature computation is skipped. All values are rounded to two decimal places. The same feature extractor is used for both individual model predictions and fused depth maps.

These statistics provide the agent with lightweight feedback about the scale, spread, dynamic range, and spatial smoothness of each predicted depth map. In particular, distributional features such as `depth_mean`, `depth_std`, and percentiles summarize the global depth layout, while gradient- and Laplacian-based features describe local discontinuities, surface roughness, and potential noise. The agent can compare these features across different depth experts or fusion results before selecting the final solution.

Details of MLP router. We train a lightweight MLP router as a supervised baseline for depth-solution selection. The router takes only the input RGB image as input, without any language instruction or reasoning trace. Specifically, each image is encoded by a frozen CLIP ViT-L/14-336 image encoder, producing a 768-dimensional visual feature. A trainable two-layer MLP head maps this feature to

Table 6: Depth features returned by the depth tools. V denotes valid finite positive depth pixels.

Feature	Definition	Interpretation
`depth_mean`	$\mathrm{Mean}(V)$	Average scene depth.
`depth_std`	$\mathrm{Std}(V)$	Depth variation across the scene.
`depth_range`	$\max(V) - \min(V)$	Overall near-to-far extent.
`depth_median`	$\mathrm{Median}(V)$	Robust central depth, less sensitive to outliers.
`depth_skewness`	$\mathrm{Skew}(V)$	Asymmetry of the depth distribution.
`depth_p10`	$P_{10}(V)$	Near-depth percentile.
`depth_p90`	$P_{90}(V)$	Far-depth percentile.
`depth_p90_p10_ratio`	$P_{90}(V)/(P_{10}(V)+\epsilon)$	Relative depth dynamic range.
`gradient_mean`	$\mathrm{Mean}(\lVert\nabla D\rVert_2)$	Average local depth change; high values indicate many edges or discontinuities.
`gradient_std`	$\mathrm{Std}(\lVert\nabla D\rVert_2)$	Spatial variability of local depth changes.
`laplacian_std`	$\mathrm{Std}(\Delta D)$	Local curvature/noise level; high values suggest rough or unstable depth.

logits over the candidate solution space:

$$\mathbf{h} = \mathrm{CLIP}_{\mathrm{frozen}}(I), \qquad \mathbf{z} = \mathrm{MLP}(\mathbf{h}) \in \mathbb{R}^K,$$

where the MLP is

$$\mathrm{Linear}(768, 256) \to \mathrm{ReLU} \to \mathrm{Dropout}(0.1) \to \mathrm{Linear}(256, K).$$

The output is a categorical distribution over depth solutions, and the predicted solution is selected by $\arg\max_k z_k$.

The solution space contains $K = 83$ candidates constructed from five depth predictors:

$$\mathcal{M} = \{\texttt{UniDepth}, \texttt{Metric3D}, \texttt{Metric3Dv2}, \texttt{UniDAC}, \texttt{UniK3D}\}.$$

A solution is defined as a pair (S, f), where $S \subseteq \mathcal{M}$ is a non-empty subset of depth models and f is a fusion operator. For single-model solutions, only the `single` operator is valid. For multi-model solutions, we use `avg`, `max`, and `min` fusion. Thus,

$$K = \binom{5}{1} + 3\sum_{r=2}^{5}\binom{5}{r} = 5 + 30 + 30 + 15 + 3 = 83.$$

For each training image, we compute a hard oracle label by running all five depth models and enumerating all 83 candidate solutions. Each candidate solution is applied to produce a depth prediction, which is evaluated against the ground-truth depth using the valid depth mask. The oracle label is the index of the solution with the best δ_1 score:

$$y^\star = \arg\max_{k \in \{1,\ldots,K\}} \delta_1\left(\hat{D}_k, D^\star\right).$$

The router is trained with standard cross-entropy loss:

$$\mathcal{L}_{\mathrm{router}} = -\log \frac{\exp(z_{y^\star})}{\sum_{k=1}^{K}\exp(z_k)}.$$

Only the MLP head is optimized; the CLIP image encoder is kept frozen. The router is trained for 50 epochs with batch size 4 using AdamW, learning rate 10^{-3}, weight decay 10^{-4}, gradient clipping at 1.0, and seed 42. The learning rate uses a 2-epoch linear warmup from $0.1\times$ the base learning rate, followed by cosine decay to 10^{-5}.

Details of rule-based router. This baseline instantiates a camera-type classifier with a fixed per-domain expert set and fusion rule. For each training image x, we obtain its camera type t_x with the same `estimate_camera_type` tool used by DepthAgent. For each camera type t, let $\mathcal{D}_t = \{x \mid t_x = t\}$. We evaluate all 83 candidate solutions (S, f) on $\mathcal{D}_t$ and store the one with the highest average δ_1 in a lookup table $C[t] = (S_t^\star, f_t^\star)$. At test time, the router estimates t_x and executes only

the experts in $C[t_x]$ with the stored fusion rule. Since each entry is selected by exhaustive search, C is the best fixed per-domain configuration obtainable from the training set.

Details of metadata and hybrid routers. The metadata router takes (i) 11 camera-metadata features, *i.e.*, $(f_x, f_y, c_x, c_y, W, H)$ and the scale-free quantities hFoV, vFoV, c_x/W, c_y/H, and f_x/W, and (ii) the 4-d one-hot camera type from `estimate_camera_type`; these 15 features are standardized with the statistics of the training images. The hybrid router concatenates the 768-d CLIP feature of the MLP router with (i) and (ii), giving a 783-d input. Both routers use the same two-layer MLP as the MLP router with the input dimension adjusted, but are trained to regress the δ_1 of all K candidate solutions with an L1 loss, and select the candidate with the highest predicted δ_1 at inference. We adopt this objective because it outperforms cross-entropy training on oracle labels with the same inputs for both routers (mean δ_1 over the six test sets: 0.799 vs. 0.614 for the metadata router and 0.730 vs. 0.617 for the hybrid router). The optimizer, schedule, training images, and seed are the same as for the MLP router.

Details of random selection baselines. The full tables (Tab. 8 and Tab. 9) additionally report two random selection baselines: Rand-Model randomly selects one expert per sample, and Rand-Sol randomly selects one candidate solution. For both, we restrict the sampling space to the model family that matches the input camera domain, *e.g.*, perspective-oriented candidates for perspective datasets and ERP-oriented candidates for fisheye and panoramic datasets. This gives the random baselines access to a coarse camera-domain prior and therefore makes them stronger than sampling from the full candidate set. Without this restriction, random selection performs substantially worse. Therefore, the reported random baselines provide conservative comparisons rather than artificially weakened ones.

Details of training dataset. We use A2D2, HM3D, HM3D (ERP), DDAD, and Matterport3D, with sampling weights $1, 1, 1, 1, 2$, respectively, and at most 200 samples per dataset.

Prompts. We provide the system prompt for CoT mode used in the agent training and inference in Fig. 6. The tool schema is the function documentation with input and output formats and meaning.

Details of `estimate_camera_type` tool. The `estimate_camera_type` tool takes camera intrinsic parameters and image dimensions as input, and classifies the camera into one of four types: *standard perspective*, *atypical perspective*, *ERP*, or *fisheye*. It first computes the horizontal field of view (hFoV) from the focal length and image width, then applies a series of threshold-based rules on the hFoV, principal point position, and image resolution to determine the camera type.

Additional details of implementation. For GRPO, we use a group size of $G = 4$ and set the KL regularization coefficient to $\beta = 0.06$. We adopt LoRA (Hu et al., 2022) with rank $r = 64$ and scaling factor $\alpha = 128$. The learning rate is initialized at 2×10^{-5} and decayed linearly during training. All depth experts are frozen throughout training. The reward weights w_{fmt}, w_{tool}, w_{scene}, w_{sel}, and w_{em} are 1 for every reward function. For Rand-Model and Rand-Sol, we report results averaged over five random seeds.

E Additional Experiment Details

Full table of quantitative comparison. Tab. 8 and Tab. 9 show that DepthAgent consistently outperforms individual experts and non-agentic fusion baselines across camera domains, achieving the best average results on perspective, fisheye, and panoramic datasets, with clear gains on challenging benchmarks such as IBims-1.

Sample-wise vs. domain-level routing. Since expert preference is strongly tied to camera geometry (Sec. 3.1), a natural question is whether routing by camera domain alone suffices. The rule-based router tests this directly: although it uses the same camera-type estimate as DepthAgent and the best per-domain solution found on the training data, it is worse than DepthAgent on every metric in all three domains while calling more experts ($\bar{n}$ 2.3 vs. 1.9; Tab. 2). Averaged over the six test sets, DepthAgent reduces A.Rel by 37% and RMSE by 25% relative to it. The rule-based router even falls behind the single expert UniK3D in δ_1 and A.Rel, showing that a domain-level optimum does not transfer, as the best solution varies across samples within each domain. Learned routers given the

Depth Agent System Prompt

Role
You are a depth estimation routing agent. Your goal is to produce the highest-quality depth map from a single RGB image by selectively calling depth models and declaring the optimal ensemble, while minimizing unnecessary model calls.
Workflow
Each turn: output <think>...</think> then ONE action. Wait for <tool_result> before the next turn.
Budget Awareness
Every <tool_result> contains remain_turn: the number of turns remaining after this turn. When remain_turn > 1, you may call another model or output <answer>. When remain_turn == 1, you must output <answer>.
Step 1 – Anchor (mandatory, Turn 1)
Analyze the image before any tool call, and explicitly identify the scene. Also note texture density, lighting, reflective/transparent regions, and large open areas vs clutter.
Understand the camera intrinsics together with the estimated camera type from [Metadata] to choose suitable models, and explicitly state the camera type and scene type (indoor / outdoor) in your first <think> block. Use the visual appearance as a secondary check.
Camera type guidance:
- standard_perspective: normal photo with typical optics → prefer perspective models.
- atypical_perspective: perspective image but unusual focal length → use erp/fisheye models.
- erp: panorama/panorama-strip layout, strong top/bottom stretching → use erp/fisheye models.
- fisheye: very wide-angle or fisheye projection → use erp/fisheye models.
Step 2 – Observe & Decide (optional)
After each get_depth_tool result, carefully analyze the returned depth_features, then decide whether to call another model or output <answer>.
- depth_features: statistics of this model's depth map (depth_mean, depth_std, depth_range). Use these to judge depth map quality.
- remain_turn: turns remaining after this turn (including the answer turn).
Step 3 – Answer
NOTE: Calling get_depth_tool does not commit a model to the final answer.
Output <answer> declaring your chosen experts and fusion method.
Always check remain_turn before each action: when remain_turn == 1, you MUST output <answer> this turn.
Output Format
Each turn MUST output EXACTLY TWO blocks – no text outside tags:
1. <think>...</think>
2. ONE action: <tool_call>{{"name": ..., "parameters": ...}}</tool_call> OR <answer>{JSON}</answer>
Tools
You may call one or more functions to assist with the user query.
You are provided with function signatures within <tools></tools> XML tags: <tools> {TOOL_SCHEMA} </tools>
For each function call, return a json format object with function name and arguments within <tool_call></tool_call> XML tags:
<tool_call> {{"name": <function-name>, "parameters": <args-json-object>}} </tool_call>.
Only call declared tools.
Answer Schema
Output <answer> with a single JSON object – no list, no extra keys:
{ "solution_id": "001",
"experts": ["<model_1>", "<model_2>"],
"fusion": "<method>", }
- experts: subset of called models to include in the final ensemble.
- fusion: one of [single, avg, max, min]. When experts contains exactly one model, fusion MUST be "single"; any other fusion is invalid.
Tips
- **Analyze depth_features**: compare depth_mean, depth_std, depth_range across models to assess quality.
- **Answer**: output <answer> immediately before remain_turn reaches 1.

Figure 6: CoT prompt for DepthAgent.

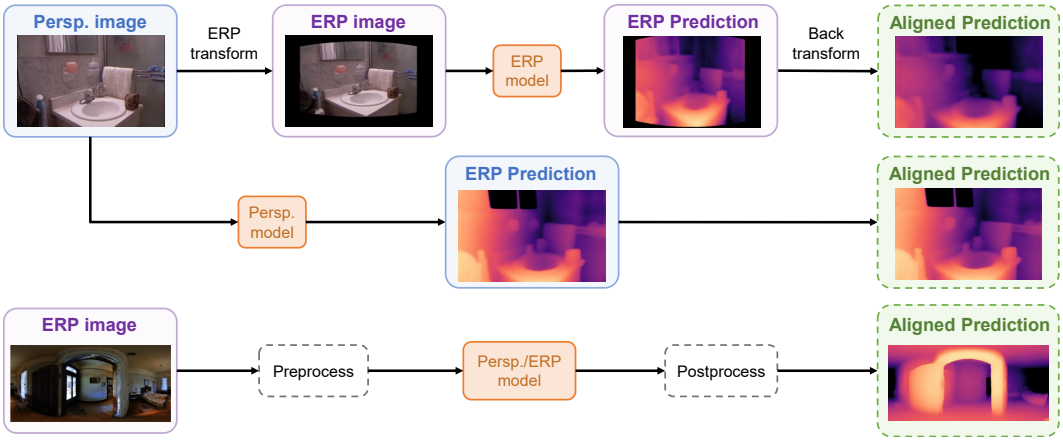


Figure 7: Prediction alignment pipeline. For heterogeneous inputs, expert predictions are generated in their compatible camera formats and then transformed back to the original image space, enabling fusion and evaluation under a shared target domain.

Table 7: **Robustness to expert-pool changes.** DepthAgent is not retrained, while the baselines are re-fitted on the modified pool. Perspective averages KITTI, NYU-v2, and IBims-1; fisheye is ScanNet++; panoramic averages Matterport3D and Pano3D-GV2.

Method	Perspective			Fisheye			Panoramic		
	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
Original pool									
DepthAgent	0.953	0.068	0.907	0.947	0.083	0.254	0.819	0.145	0.365
Remove Metric3Dv2									
Mean (4 experts)	0.898	0.110	1.078	0.819	0.151	0.316	0.675	0.184	0.482
Rule-based router (re-fitted)	0.910	0.101	1.341	0.918	0.097	0.279	0.755	0.168	0.472
DepthAgent (no retraining)	**0.943**	**0.074**	**0.962**	**0.947**	**0.083**	**0.254**	**0.819**	**0.145**	**0.365**
Replace UniDepth with UniDepthV2 (unseen during training)									
DepthAgent (no retraining)	0.949	0.071	1.010	0.947	0.083	0.254	0.819	0.145	0.365

same camera cues do not close the gap either. The metadata router effectively reduces to domain-level routing, *e.g.*, it assigns a single solution to all 1,068 ScanNet++ images. The hybrid router, like the MLP router, fails on the unseen fisheye domain (δ_1 0.239 on ScanNet++): although the camera prior labels these images as ERP, its most frequent choice is a combination of three perspective experts (Metric3D+Metric3Dv2+UniDepth), selected for 478 of the 1,068 images. Since the metadata router, which has no image features, avoids this failure, the image features appear to override the camera prior on out-of-distribution inputs. Hence, the advantage of DepthAgent stems from its sample-wise policy rather than from additional input information.

Robustness to expert-pool changes. We modify the tool schema at inference time while keeping the trained agent unchanged. **Removing an expert.** We delete Metric3Dv2 from the pool. The agent makes no call to the removed tool, whereas Mean fusion and the rule-based router have to be re-fitted on the remaining four experts. As shown in Tab. 7, DepthAgent still outperforms both baselines on all metrics without retraining. On perspective data, it replaces Metric3Dv2-based combinations with Metric3D-based ones, while the fisheye and panoramic results are unchanged since these inputs are routed to ERP-trained experts. In contrast, the re-fitted rule-based router changes considerably (perspective δ_1 from 0.852 to 0.910), showing that domain-level optima are sensitive to the pool. **Introducing an unseen expert.** We replace UniDepth with UniDepthV2 (Piccinelli et al., 2025b), whose tool name never appears during training. Based only on the tool description in the prompt, the agent adopts UniDepthV2 on perspective inputs, makes no call to the replaced UniDepth, and performs close to the original pool. All results in Tab. 7 are obtained with the same checkpoint and inference script, which may slightly differ from Tab. 2.

Full table of comparison on hard samples. Tab. 10 and Tab. 11 report the full comparison on hard samples from the perspective, fisheye, and 360° datasets. On the perspective dataset, our method

Table 8: **Quantitative comparison on fisheye and 360° datasets**. We report on ScanNet++, Matterport3D, and Pano3D-GV2. **Avg.** is the arithmetic mean over three datasets. ‡: best solution per sample.

Methods	ScanNet++			Matterport3D			Pano3D-GV2			Avg.		
	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.135	0.573	1.029	0.448	0.310	0.961	0.381	0.318	0.834	0.321	0.400	0.941
Metric3D	0.727	0.162	0.450	0.378	0.295	0.818	0.427	0.285	0.623	0.511	0.247	0.630
Metric3Dv2	0.745	0.170	0.474	0.438	0.292	0.884	0.405	0.307	0.810	0.529	0.256	0.723
UniDAC	0.918	0.097	0.279	0.742	0.176	0.550	0.768	0.161	0.394	0.809	0.145	0.408
UniK3D	0.927	0.103	0.287	**0.818**	0.152	**0.419**	0.822	0.151	0.347	0.856	0.135	0.351
Mean	0.840	0.139	0.313	0.615	0.204	0.597	0.625	0.193	0.460	0.694	0.179	0.457
Max	0.139	0.573	1.023	0.724	0.245	0.564	0.745	0.224	0.435	0.536	0.351	0.674
Min	0.663	0.190	0.573	0.284	0.340	1.150	0.247	0.367	1.032	0.398	0.299	0.918
DepthFusion	0.621	0.252	0.623	0.405	0.323	1.058	0.351	0.330	0.941	0.459	0.302	0.874
Rand-Model	0.921	0.100	0.284	0.779	0.164	0.487	0.794	0.156	0.371	0.831	0.140	0.381
Rand-Sol	0.927	0.097	0.279	0.784	0.162	0.480	0.799	0.153	0.363	0.837	0.138	0.374
Rule-based router	0.862	0.128	0.301	0.723	0.197	0.545	0.747	0.182	0.399	0.777	0.169	0.415
MLP router	0.187	1.014	2.547	0.733	0.189	0.549	0.755	0.174	0.396	0.558	0.500	1.164
Metadata router	0.851	0.133	0.308	0.736	0.191	0.574	0.765	0.168	0.386	0.784	0.164	0.423
Hybrid router	0.239	0.519	0.938	0.733	0.210	0.553	0.744	0.208	0.428	0.572	0.312	0.640
DepthAgent	**0.946**	**0.084**	**0.254**	0.811	**0.150**	0.420	**0.826**	**0.140**	**0.310**	**0.861**	**0.125**	**0.328**
Upper-bound‡	0.967	0.080	0.239	0.853	0.136	0.412	0.867	0.128	0.308	0.896	0.115	0.320

Table 9: **Quantitative comparison on perspective datasets**. We report on KITTI, NYU-v2, and IBims-1. **Avg.** is the arithmetic mean over three datasets. ‡: best solution per sample.

Methods	KITTI			NYU-v2			IBims-1			Avg.		
	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.977	0.052	2.305	**0.982**	0.069	0.224	0.142	0.487	1.549	0.700	0.203	1.359
Metric3D	0.967	0.056	2.676	0.915	0.103	0.364	0.668	0.250	0.796	0.850	0.136	1.279
Metric3Dv2	0.975	0.051	2.503	0.973	0.066	0.256	0.644	0.255	0.821	0.864	0.124	1.193
UniDAC	0.797	0.178	4.556	0.922	0.098	0.370	0.817	0.143	0.703	0.845	0.140	1.876
UniK3D	0.974	0.069	3.006	0.972	**0.060**	0.243	0.865	0.145	0.535	0.937	0.091	1.261
Mean	0.976	0.053	2.244	0.968	0.064	0.231	0.725	0.212	0.660	0.890	0.110	1.045
Max	0.780	0.188	4.736	0.895	0.132	0.402	0.094	0.537	1.691	0.590	0.285	2.276
Min	0.957	0.076	2.822	0.943	0.076	0.308	0.835	0.133	0.645	0.912	0.095	1.258
DepthFusion	0.971	0.056	2.440	0.954	0.078	0.291	0.708	0.217	0.854	0.878	0.117	1.195
Rand-Model	0.973	0.053	2.486	0.954	0.079	0.280	0.475	0.344	1.096	0.801	0.159	1.287
Rand-Sol	0.973	0.053	2.447	0.958	0.078	0.276	0.492	0.320	1.028	0.808	0.150	1.250
Rule-based router	0.977	0.053	2.598	0.935	0.109	0.345	0.643	0.256	0.823	0.852	0.139	1.255
MLP router	0.957	0.067	2.891	0.947	0.091	0.304	0.781	0.163	0.788	0.904	0.107	1.328
Metadata router	0.932	0.102	2.961	0.973	0.066	0.256	0.538	0.325	0.995	0.814	0.164	1.404
Hybrid router	0.910	0.099	3.052	0.966	0.073	0.262	0.790	0.166	0.549	0.889	0.113	1.288
DepthAgent	**0.980**	**0.045**	**2.114**	0.979	0.061	**0.218**	**0.886**	**0.103**	**0.422**	**0.948**	**0.070**	**0.918**
Upper-bound‡	0.987	0.046	2.025	0.993	0.051	0.186	0.975	0.073	0.322	0.985	0.057	0.844

Table 10: **Detailed comparison on hard samples from perspective datasets. Avg.** denotes the arithmetic mean over the datasets.

Method	KITTI			NYU-v2			IBims-1			Avg.		
	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.924	0.080	3.916	0.916	0.099	0.393	0.111	0.634	1.825	0.650	0.271	2.045
Metric3D	0.886	0.090	4.803	0.806	0.151	0.576	0.351	0.420	1.555	0.681	0.220	2.311
Metric3Dv2	0.909	0.081	4.376	0.890	0.101	0.464	0.413	0.453	1.377	0.737	0.212	2.072
UniDAC	0.760	0.197	5.278	0.811	0.146	0.582	**0.690**	0.188	0.997	0.754	0.177	2.286
UniK3D	0.915	0.095	4.439	**0.917**	**0.092**	0.483	0.551	0.277	0.928	0.794	0.155	1.950
DepthAgent	**0.927**	**0.073**	**3.832**	0.907	0.094	**0.395**	0.666	**0.177**	**0.746**	**0.833**	**0.115**	**1.658**
Upper-bound‡	0.951	0.068	3.433	0.962	0.085	0.366	0.868	0.132	0.596	0.927	0.095	1.465

improves most metrics, with larger gains under the fisheye setting, indicating stronger robustness to severe geometric distortion. On the 360° Datasets, our method achieves consistent improvements on challenging samples, further demonstrating its effectiveness and practical importance in difficult scenarios.

Comparison on largest-gain samples. To further analyze where DepthAgent brings the most benefit, we conduct a diagnostic evaluation on the top 10% samples with the largest per-sample improvement of DepthAgent over the strongest non-oracle baseline within each dataset. This subset highlights cases where fixed expert models are less reliable and adaptive tool selection is most beneficial.

Table 11: **Detailed comparison on hard samples from fisheye and 360° datasets. Avg.** denotes the arithmetic mean over the datasets.

Method	ScanNet++			Matterport3D			Pano3D-GV2			Avg.		
	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.197	0.610	0.939	0.343	0.513	0.908	0.331	0.478	0.797	0.290	0.534	0.881
Metric3D	0.635	0.203	0.629	0.351	0.372	0.931	0.340	0.372	0.762	0.442	0.316	0.774
Metric3Dv2	0.659	0.204	0.593	0.311	0.422	0.931	0.272	0.431	0.906	0.414	0.352	0.810
UniDAC	0.789	0.163	0.434	0.531	0.286	0.608	0.562	0.260	0.441	0.627	0.236	0.494
UniK3D	0.799	0.156	0.434	0.534	0.273	0.532	0.586	0.248	0.400	0.640	0.226	0.455
DepthAgent	**0.817**	**0.144**	**0.410**	**0.561**	**0.268**	**0.529**	**0.612**	**0.238**	**0.370**	**0.663**	**0.217**	**0.436**
Upper-bound‡	0.904	0.117	0.384	0.631	0.238	0.569	0.670	0.211	0.378	0.735	0.189	0.444

Table 12: **Detailed comparison on largest-gain samples from perspective datasets.** For each dataset, we select the top 10% samples where DepthAgent achieves the largest per-sample improvement over the oracle single model baseline. **Avg.** denotes the arithmetic mean over the datasets.

Method	KITTI			NYU-v2			IBims-1			Avg.		
	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.974	0.055	2.503	0.984	0.076	0.252	0.140	0.490	1.694	0.699	0.207	1.483
Metric3D	0.954	0.066	3.132	0.890	0.109	0.373	0.561	0.236	1.146	0.802	0.137	1.550
Metric3Dv2	0.967	0.006	3.132	0.976	0.064	0.245	0.645	0.343	1.258	0.863	0.156	1.501
UniDAC	0.748	0.190	4.728	0.902	0.111	0.355	0.719	0.164	1.059	0.790	0.155	2.047
UniK3D	0.966	0.078	3.553	0.958	0.061	0.264	0.820	0.159	0.765	0.915	0.099	1.527
DepthAgent	**0.983**	**0.047**	**2.264**	**0.990**	**0.057**	**0.202**	**0.965**	**0.074**	**0.566**	**0.979**	**0.059**	**1.011**

Table 13: **Detailed comparison on largest-gain samples from fisheye and 360° datasets.** For each dataset, we select the top 10% samples where DepthAgent achieves the largest per-sample improvement over the oracle single model baseline. **Avg.** denotes the arithmetic mean over the datasets.

Method	ScanNet++			Matterport3D			Pano3D-GV2			Avg.		
	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	δ_1	A.Rel	RMSE	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$
UniDepth	0.124	0.625	1.000	0.415	0.349	1.001	0.319	0.383	0.904	0.286	0.452	0.968
Metric3D	0.761	0.157	0.447	0.320	0.325	0.886	0.334	0.355	0.745	0.472	0.279	0.693
Metric3Dv2	0.762	0.166	0.456	0.298	0.356	1.079	0.287	0.377	0.981	0.449	0.300	0.839
UniDAC	0.889	0.118	0.316	0.703	0.173	0.588	0.701	0.175	0.433	0.764	0.155	0.446
UniK3D	0.910	0.117	0.322	0.768	0.169	0.436	0.746	0.179	0.376	0.808	0.155	0.378
DepthAgent	**0.953**	**0.087**	**0.275**	**0.837**	**0.144**	**0.420**	**0.835**	**0.144**	**0.306**	**0.875**	**0.125**	**0.334**

Table 14: Ablation of reward hyperparameter λ, τ, and $n_{\max}$. $\bar{n}$: average number of tool calls.

λ	τ	$n_{\max}$	Perspective			Fisheye			Panoramic			$\bar{n}$
			$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	$\delta_1 \uparrow$	A.Rel $\downarrow$	RMSE $\downarrow$	
0.1	1.0	1.0	0.859	0.116	1.071	0.933	0.091	0.243	0.773	0.175	0.418	1.0
0.4	1.0	1.0	0.907	0.092	0.978	0.918	0.096	0.279	0.755	0.168	0.471	1.0
0.1	2.0	2.0	0.797	0.128	1.174	0.733	0.204	0.430	0.762	0.167	0.431	3.2
0.4	3.4	2.0	0.822	0.139	1.158	0.890	0.114	0.285	0.773	0.175	0.418	2.5
0.2	3.4	2.0	0.948	0.070	0.918	0.946	0.084	0.254	0.818	0.145	0.365	1.9

Additional details of efficiency of DepthAgent. All latency numbers are measured end to end under the same input resolution with batch size 1 on a single H100 GPU, averaged over 200 samples. We decompose the latency into VLM decoding, expert execution, pre-/post-processing, and fusion (Tab. 15). VLM decoding is served with vLLM (Kwon et al., 2023), which does not change the routing decisions in Fast mode. Exhaustive fusion runs all five experts before applying mean fusion. By invoking only the selected experts, DepthAgent reduces the expert execution time from 0.422 s to 0.048 s and also lowers the pre-/post-processing and fusion costs; the remaining overhead mainly comes from VLM decoding, which can be further reduced with smaller VLMs or policy distillation. The full system, including the VLM and all five experts, runs on a single GPU with 14.23 GB peak memory.

Additional effects of reward hyperparameter. As shown in Tab. 14, $n_{\max}$ is closely correlated with the average number of total calls $\bar{n}$. Specifically, when $n_{\max} = 1$, $\bar{n}$ remains fixed at 1.0,

Table 15: **End-to-end latency and GPU memory.** Latency (s) is measured with the same input resolution and batch size 1 on one H100 GPU, averaged over 200 samples; VLM decoding uses vLLM.

Method	VLM decode	Expert exec.	Pre/Post	Fusion	Total	Params	Peak VRAM
All 5 experts (Mean)	–	0.422	0.076	0.012	0.510	1.68B	8.21 GB
DepthAgent (Fast)	0.910	0.048	0.019	0.001	0.978	5.43B	14.23 GB
DepthAgent (CoT)	1.844	0.096	0.023	0.002	1.965	5.43B	14.23 GB



Figure 8: **Conversation visualization of DepthAgent in CoT mode.** Given an image and its metadata, DepthAgent infers the camera type and scene characteristics, then routes it to suitable depth experts: perspective models such as UniDepth and Metric3Dv2 for standard images, or geometry-aware models such as UniDAC and UniK3D for 360°/fisheye inputs. The selected predictions are then combined with a simple fusion strategy.

while allowing $n_{\max} = 2$ results in increased call usage. We also observe that λ can affect $\bar{n}$ under comparable settings, with larger values tending to encourage more tool calls. In addition, τ influences the final performance across camera types, suggesting that the reward temperature is important for controlling the optimization behavior.

F Additional Visualizations

Conversation. Fig. 8 illustrates the CoT mode of DepthAgent, where the agent interprets the input image and dynamically selects suitable experts for analysis. The visualization shows that the agent can use both visual appearance and metadata to identify the camera type, *e.g.*, distinguishing a standard perspective image from a fisheye/360° image. Based on this judgment, DepthAgent routes the input to different groups of depth experts: perspective-oriented models are selected for regular indoor perspective images, while geometry-aware models are invoked for panoramic or fisheye inputs. This adaptive routing allows the system to avoid applying a fixed expert set to all images, and instead perform model selection according to the image geometry before fusing the expert outputs into the final answer.

Depth map comparison. We provide additional depth map comparsion in Fig. 9.

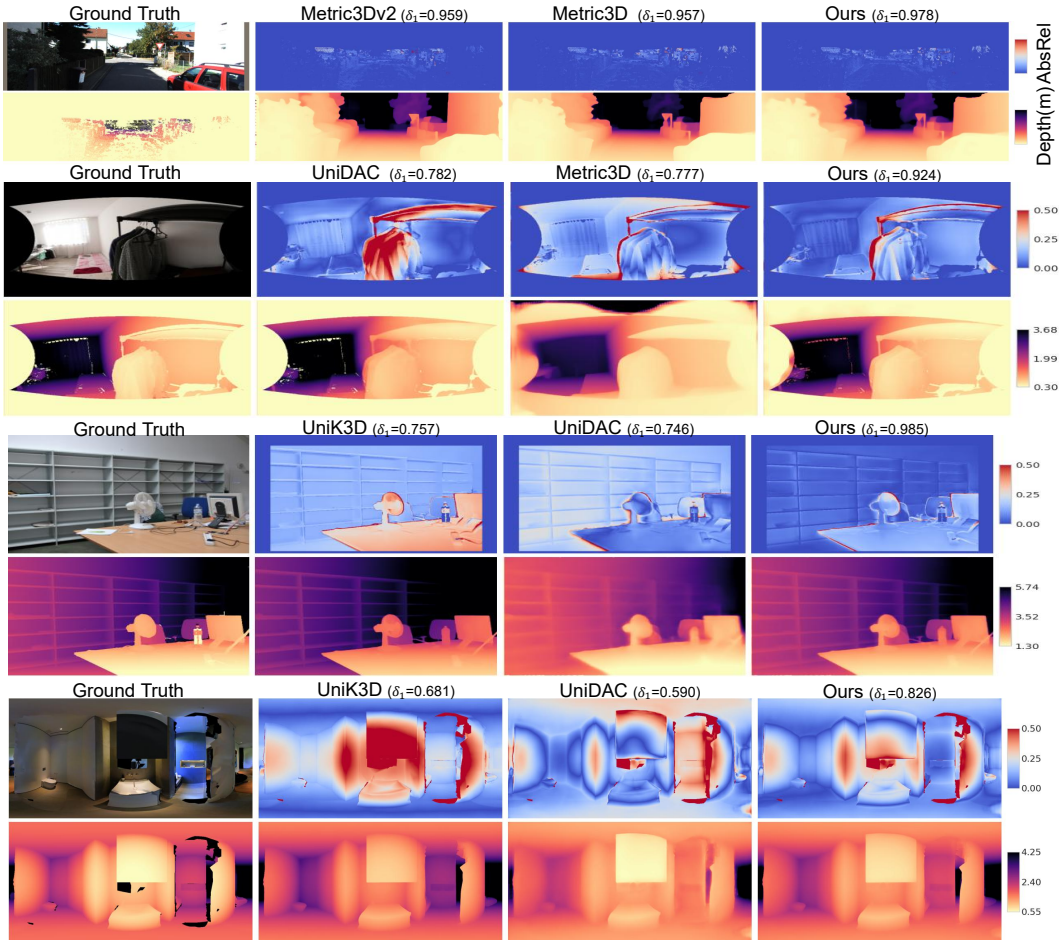


Figure 9: **Depth Map Comparison.** For each sample, we compare the top-2 expert predictions with the final solution of DepthAgent. While individual experts may exhibit high errors in different regions, DepthAgent produces more accurate depth maps and lower error maps. Zoom in for better effects.

G LIMITATIONS

Although our method consistently outperforms the baseline, a noticeable gap remains between DepthAgent and the upper bound, indicating that the current policy has not yet fully exploited the available solution space.

Another limitation lies in the domain gap between training and evaluation. Specifically, the oracle solution used during training may not always be optimal at test time, making robustness to imperfect solution supervision essential. In addition, introducing a VLM-based agent inevitably increases model size and computational cost compared with non-agent approaches. Nevertheless, our experiments demonstrate that the agent yields clear performance gains, and improving its efficiency remains an important direction for future work.