MLLM-based Dataset Construction for Hazard-aware Guidance for the Visually Impaired

PEIYUAN ZHU[1,a)] MARC A. KASTNER[2,b)] HIROTAKA KATO[1,c)]
TAKATSUGU HIRAYAMA[3,1,d)] TAKAHIRO KOMAMIZU[1,e)] ICHIRO IDE[1,f)]

Abstract

Outdoor navigation in unfamiliar environments poses a considerable risk to people with visual impairments, exposing them to hazardous situations, such as collisions with overhanging sharp obstacles or falls into open pits or manholes. Conventional vision-based assistive systems struggle to balance strong generalization with lightweight deployment. Moreover, existing Multimodal Large Language Models (MLLMs) are too large and not optimized for first-person outdoor use, preventing efficient on-device execution and real-time, efficient guidance. To address this, we propose an outdoor hazard detection and avoidance guidance generation method built on a small-parameter MLLM. To enable scalable dataset creation for training the model, we design a large-parameter MLLM-driven automatic annotation pipeline that produces large-scale, high-quality labels with minimal human supervision, resulting in annotations that closely match the expert ground truth.

1. Introduction

Globally, approximately 220 million people have visual impairments, including around 43 million who are completely blind [7]. For severely visually impaired individuals, as illustrated in Fig. 1, walking along unfamiliar streets presents numerous hazards that are difficult to perceive and can potentially lead to injury. For instance, visually impaired individuals may be seriously injured if they accidentally fall into an open manhole. Similarly, overhanging objects are often undetectable by a white cane, making it easy for visually impaired individuals to bump into them. When facing hazards, guide dogs remain the safest form of assistance. However, their training is both time-consuming and expensive, which hinders their widespread adoption. With the ubiquity of smartphones, lightweight phone-centered assistance systems have gained significant traction in recent years [4].

Existing smartphone-based aids are commonly Convolutional Neural Network (CNN)-based obstacle detectors, such as Eye Navi [4], or Vision–Language Model (VLM)-based assistants, such as Magnifier on iPhone [1]. The former recognizes familiar objects but are limited to pretrained classes. In contrast, the latter can excel at free-form scene description but suffer from high reasoning latency for real-time outdoor navigation and rely on user-initiated interaction.

Fig. 1: Visually impaired individuals encounter numerous unknown and complex environments when traveling outdoors. We aim at making a model to facilitate hazard recognition and avoidance guidance generation.

In recent years, Multimodal Large Language Models (MLLMs) have demonstrated potential for assisting visually impaired individuals [6]. Unlike conventional VLMs, which first extract visual features with a separate encoder before passing them to a standalone text-generation module, MLLMs inject visual adapters or prefix-tuning layers directly into the Transformer backbone. This unified pipeline reduces feature transfer overhead, enables end-to-end reuse of multimodal context, and delivers more efficient inference for complex tasks and richer image-text dialogues under comparable hardware conditions.

Our work presents two core challenges: (1) There is a need for low latency and lightweight deployment, as visually impaired individuals require the ability to run it on their lightweight computing devices (e.g., smartphones). (2) Since the task is complex, small-parameter MLLMs without fine-tuning tend to produce unreliable results in hazard identification and avoidance guidance generation.

To address these challenges, we require a first-person dataset encompassing key hazard categories and extreme scenarios for fine-tuning. However, there is a scarcity of datasets, as existing datasets often lack hazard categories or do not incorporate first-

[1] Nagoya University
[2] Hiroshima City University
[3] University of Human Environments
[a)] zhup@cs.is.i.nagoya-u.ac.jp
[b)] mkastner@hiroshima-cu.ac.jp
[c)] katoh@cs.is.i.nagoya-u.ac.jp
[d)] t-hirayama@uhe.ac.jp
[e)] taka-coma@acm.org
[f)] ide@i.nagoya-u.ac.jp

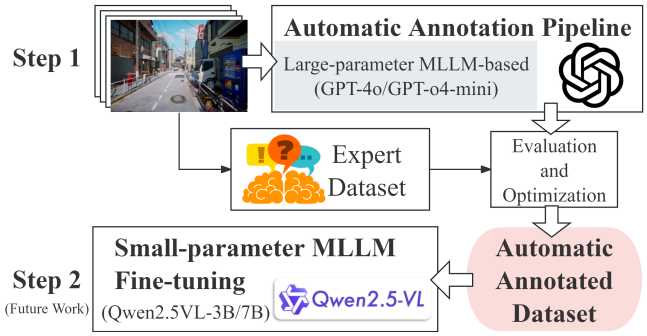


Fig. 2: Overview of the two-step process for dataset construction and model fine-tuning. We collect the dataset, apply an automatic annotation pipeline based on GPT to label the data, and use an expert dataset to evaluate and optimize the automatic annotation pipeline. In the future, we plan to fine-tune small-parameter MLLMs (e.g., Qwen2.5-VL-3B [3]) on this annotated dataset.

person perspectives. Therefore, this presentation focuses exclusively on the construction and annotation of this dataset. We develop a dataset covering extreme scenarios and devise a pipeline that combines automatic annotation with few-expert supervision, as shown in Fig. 2.

2. Related Work

Multimodal Large Language Models (MLLMs). Recent advances in vision–language pre-training have produced models that jointly encode images and text, enabling open-ended visual reasoning. Proprietary successors (e.g. GPT-4V [10] and Qwen2.5-VL [3]) further improve zero-shot generalisation and have been deployed in assistive tools such as *Be My Eyes* [16]. However, their latency and size render them impractical for on-device use, which motivates our lightweight and task-specific fine-tuning.

LAVE (LLM-Assisted VQA Evaluation). Automatic VQA metrics, such as BLEU [13] and CIDEr [15], favour *n*-gram overlap and fail to capture synonymy or factual correctness. LAVE [8] leverages an LLM as a reference-free judge. Given an image, a ground-truth answer, and a candidate answer, the LLM estimates semantic agreement and assigns a calibrated score. Experiments show higher correlation with human judgements than traditional metrics [6]. Our annotation pipeline uses LAVE to ensure the quality of automatically generated labels.

3. Dataset Construction

3.1 Data Sources

Our dataset must be presented from a first-person perspective, cover various hazardous categories, include extreme scenarios, and provide annotations for both hazard description and locations. To satisfy these requirements, we combine two complementary sources: First, we take pictures while walking through urban streets and also select samples from existing open-source hazard corpus [2] that match a first-person perspective. These two sources are collectively referred to as the real-world sources. These sources' ground truth includes only the *hazard category* and its *2-D pixel location*. In addition, we model street scenes in `Unity` [14] as the Unity-based source, and utilize a script to capture screenshots, exporting a JSON file as ground truth that includes the *hazard category, location, and distance* to it. This file serves as the *primary dataset*. Since traditional datasets often lack categories for HANGING-OBJECTS and DEEP-PITS, we concentrate on gathering these specific categories and employ `Unity` to simulate those scenarios.

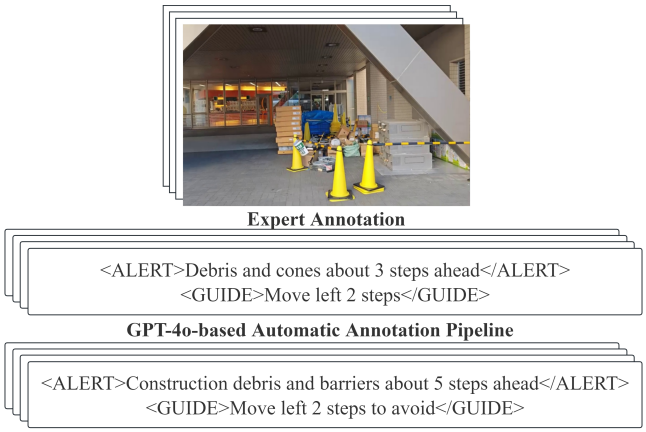


Fig. 3: Example of an annotated record from expert and automatic annotation pipelines containing `<ALERT>` and `<GUIDE>` segments.

3.2 Dataset Annotation

3.2.1 Expert Annotation

We first asked 20 volunteers who had the experience of helping the visually impaired as expert annotators to label 200 images in Chinese, thereby creating an *expert-annotated dataset* of high-quality benchmark. These 200 images were drawn from a combination of real-world (76%) and Unity-based (24%) synthetic sources. The reason we use synthetic sources is that real-world data provide authentic and intuitive contexts for human annotators, but lack many critical hazard scenarios. Therefore, we supplemented these missing categories with simulated `Unity` images. After filtering and cleaning, all expert annotations were translated into English using GPT [9]. We use this benchmark to evaluate and optimize the automatic annotation pipeline.

We wrap each expert annotation in custom tokens —`<ALERT>`, `<GUIDE>`, and `<SAFE>`— to facilitate downstream parsing and modular processing [17]. The `<ALERT>` segment contains hazard recognition information, specifying the hazard's direction, approximate distance (in steps), and description. The `<GUIDE>` segment details an avoidance strategy, including the recommended direction, number of steps, and concrete action to take. The `<SAFE />` token marks segments where no hazard is present. A typical annotated record looks like the following example:

```
<ALERT> direction, distance (steps),
hazard description </ALERT>
<GUIDE> avoidance direction, steps,
concrete action </GUIDE>
```

See Fig. 3 for an example. *Steps* are used instead of meters because our user surveys revealed that visually impaired pedestrians prefer step counts. An average stride length of 0.65 m is assumed.

3.2.2 Automatic Annotation Pipeline

To scale annotation with minimal human effort, we employ

Table 1: Difference between the naive exemplar format and the revised exemplar format in the prompt. The former refers to the original, unoptimized prompt structure as the baseline, while the latter refers to our improved template. We change the specific distance (e.g., "3 steps") to "x steps" and tell the model that "x" is a variable that it needs to judge by itself. We also replace specific words (such as "front-left") with options like "ahead/front-left/front-right".

Category	Description	Naive Exemplar Format (Original)	Revised Exemplar Format (Proposed)
HANGING-OBJECT	Used when a suspended object that could hit the user's head is detected.	A hanging water pipe about 3 steps front-left.	A hanging water pipe/rope/rail is about x steps ahead/front-left/front-right.
DEEP-PIT	Used when an open manhole or deep pit is detected.	An open manhole/deep pit roughly 4 steps front-left.	An open manhole/deep pit is about x steps straight ahead/front-left/front-right.
GROUND-LEVEL	Used when stairs are detected.	4 upward steps about 5 steps front-left.	y upward/downward steps about x steps straight ahead/front-left/front-right.
GROUND-LEVEL	Used when a car or bicycle is detected.	Acar/bicycle about 4 steps front-left.	A car/bicycle is about x steps straight ahead/front-left/front-right.

a multi-stage, text-driven pipeline. First, each input image is passed through a large-parameter MLLM that performs a binary safety classification, labeling the scene as either "*Safe*" or "*Hazard*". For images classified as "*Safe*", the pipeline simply outputs a `<SAFE />` token and proceeds to the following sample. For images classified as "*Hazard*", the model is provided with the ground-truth metadata (hazard category, approximate pixel location, and distance in steps) and generates the `<ALERT>` segment. Immediately afterward, the MLLM conditions on both the original image and its own `<ALERT>` output to produce the `<GUIDE>` segment.

Once both segments are generated, we automatically compute two evaluation metrics: the CLIPScore [5] to measure the semantic alignment between the generated text and the visual content, and the LAVE [8] metric to verify format integrity (ensuring that all required tokens, e.g., hazard and avoidance direction, distance, and their internal fields are present) as well as logical consistency (checking for contradictions between what is warned in `<ALERT>` segment and what is advised in `<GUIDE>` segment). Any annotation sample falling below predefined thresholds on these metrics is flagged for manual review. Finally, flagged samples are inspected and corrected by the authors or other qualified experts to ensure high-quality and reliable annotations.

3.3 Final Dataset

The complete dataset comprises approximately 4,000 images from real-world sources (27.3%) and Unity-based source (72.7%), divided into three *Hazard* categories and one *Safe* category. These categories are derived from interviews with visually impaired individuals and match the hazard categories as exemplified in Sec. 1:

- **HANGING-OBJECT**: Objects suspended above upper body height, which pose a threat to the upper body, such as high chassis trucks, or hanging clothes racks (3.1% in real-world sources and 25.1% in Unity-based source).
- **DEEP-PIT**: Sudden height drops, which can cause severe injury when falling into them, such as open pits or manholes (6.1% in real-world sources and 28.2% in Unity-based source).
- **GROUND-LEVEL**: All remaining obstacles resting on the ground (easily detectable by a white cane), common height changes such as staircases and situations such as slippery ground (46.8% in real-world sources and 27.3% in Unity-based source).

3.4 Prompt Design for Template Overfitting and Mitigation

We attempted to regularize the model's output using examples. However, a preliminary experiment revealed that MLLM tended to *overfit* to specific examples on the prompt. As illustrated in Table 1, frequent tokens such as "front-left" or a fixed "3 steps" were reproduced in unrelated scenes, ignoring actual visual evidence. To solve this problem, mitigation strategies, as *revised exemplar format*, are implemented as follows:

(1) **Parameterised Examples**: Replace hard-coded numerals with placeholders to prevent memorisation.
(2) **Direction Options**: Include multiple directions in the prompt so that the model learns to decide from the image rather than copy the template.

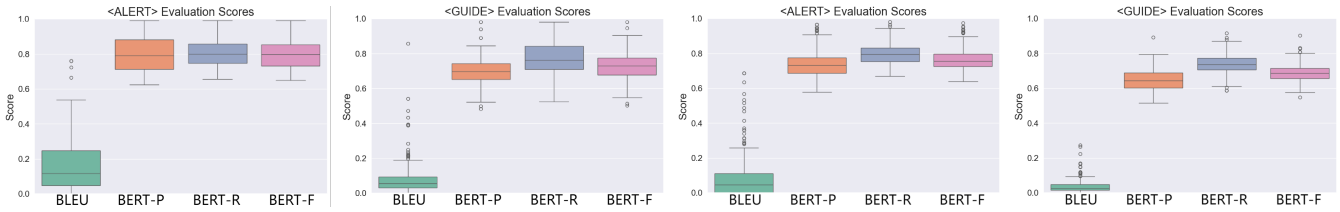
4. Analysis of Automatic Annotation

To verify and optimize the automatic annotation pipeline proposed in Sec. 3.2.2, we utilize the *expert-annotated dataset* created in Sec. 3.2.1 as the gold standard. We execute the pipeline on this dataset and assess its performance using both computational and human evaluation metrics. In our research, we primarily focus on two models: GPT-4o [11] and GPT-o4-mini [12] for automatic annotation.

4.1 Evaluation on Computational Metric

We use **BLEU** [13] and **BERTScore** [18] to quantify n-gram coverage and contextual semantic overlap between the automatically generated text by large-parameter MLLMs and expert annotations, and use **CLIPScore** to measure image-text alignment for the `<ALERT>` segment of the annotations to evaluate how well each description matches the visual content. In all experiments, we use the *revised exemplar format*.

As shown in Fig. 4, BLEU remained low (distributed between 0.00 and 0.68) because it relies on exact n-gram overlap and therefore penalises short sentences or synonymous rephrasings. In contrast, BERTScore achieved a much higher value (distributed between 0.51 and 0.98), capturing the strong semantic agreement between automatic and expert annotations. Although our pipeline did not reproduce the expert labels at the exact token level (hence the low BLEU), the high BERTScore demonstrated that it nonetheless generated semantically equivalent annotations, indicating successful automatic annotation.



(a) GPT-4o <ALERT> (b) GPT-4o <GUIDE> (c) GPT-o4-mini <ALERT> (d) GPT-o4-mini <GUIDE>

Fig. 4: Scores for model-generated annotations by GPT-4o [11] and GPT-o4-mini [12] against expert annotations. In general, BLEU Scores [13] were lower while BERTScores [18] were higher.

Table 2: CLIP Scores [5] for both model-generated annotations and expert annotations, revealing similar score distributions.

	Expert	GPT-4o [11]	GPT-o4-mini [12]
Maximum	0.6483	0.6444	0.6377
Minimum	0.5840	0.5765	0.5811
Median	0.6100	0.6123	0.6068

For the CLIPScore, Table 2 reports the maximum, minimum, and median values. The close alignment confirms that automatic descriptions matched the visual content as well as those provided by experts.

4.2 Human Evaluation

Five independent experts performed a two-stage rating on 200 image-text pairs. ALERT accuracy measures the correctness of the `<ALERT>` segment, and GUIDE quality evaluates whether the `<GUIDE>` segment is reasonable and feasible.

(1) **ALERT accuracy**: Raters judged whether *direction*, *distance* (in steps), and *hazard description* were correct. A tolerance of ± 2 steps was allowed for the distance field.

(2) **GUIDE quality**: Only if the corresponding `<ALERT>` was *entirely correct*, raters assessed whether the suggested *avoidance direction*, *steps*, and *action details* were reasonable.

Then, we calculate the accuracy and report the mean score to quantify performance.

Table 3 reports that when using the prompts with revised exemplar formats, GPT-4o [11] and GPT-o4-mini [12] achieved ALERT accuracies of 0.93 and 0.92, whereas GUIDE qualities remained lower at 0.77 and 0.75, leaving room for further improvement, respectively. As a result, it sometimes issued unsafe guidance, such as directing users off the sidewalk into the roadway to bypass a pavement hazard or choosing the wrong detour direction, despite using the revised exemplar format. Although GPT-o4-mini's overall accuracy was marginally below GPT-4o's, its guidance outputs exhibited stronger logical coherence and more consistent hazard reminders.

We also evaluated int8-quantized Qwen2.5-VL-72B [3] as an open-source alternative. As shown in Table 3, it trailed both GPT models in terms of accuracy. Consequently, we continue to employ GPT in our automatic annotation module.

Finally, a comparison between the naive and revised exemplar formats using GPT-4o showed that the revised prompt significantly improved both ALERT accuracy and GUIDE quality, thereby validating our prompt design.

Table 3: Human-evaluation accuracy of model-generated annotations. GPT-4o-Naive refers to the GPT-4o [11] model prompted with the naive exemplar format. GPT-4o-Revised, GPT-o4-mini-Revised, and QWEN-Revised denote GPT-4o, GPT-o4-mini [12], and the int8-quantized local Qwen2.5-VL-72B [3] model prompted with the revised exemplar format.

	GPT-4o-Naive	GPT-4o-Revised	GPT-o4-mini-Revised	QWEN-Revised
<ALERT>	0.8950	0.9350	0.9150	0.8350
<GUIDE>	0.6089	0.7701	0.7541	0.6287

5. Conclusion

We are aiming at utilizing small-parameter MLLMs for hazard recognition and avoidance guidance generation for the visually impaired. As the first step, we proposed a pipeline for dataset creation that combines automatic annotation with few-expert supervision, dramatically reducing labeling costs. As a result, the automatically generated annotations achieved high semantic consistency (as measured by BERTScore [18]) with only a scarce drop in median CLIPScore [5] compared to expert labels. Human evaluations showed 93% hazard ALERT accuracy and 77% avoidance GUIDE quality for avoidance guidance quality, indicating room for improvement. Furthermore, by incorporating parameterized examples and direction options, we mitigated the model's tendency to overfit to templates, yielding significant improvements in ALERT accuracy and GUIDE quality.

This report was limited to the proposal of our large-parameter MLLM-driven automatic annotation pipeline and the presentation of its experimental results. As future work, we will utilize this dataset to fine-tune small-parameter MLLM specifically for outdoor hazard recognition and avoidance guidance generation for visually impaired individuals, to minimize latency and meet performance requirements. We will compare the real-world and Unity-based sources and refine the evaluation criteria. While the small-parameter MLLM runs efficiently on desktop GPUs, mobile deployment (smartphones' NPUs) requires further model compression and runtime optimization. We leave on-device inference and latency benchmarking for the future.

References

[1] Apple Inc.: Detection mode in magnifier, https://support.apple.com/en-my/guide/iphone/iph32deb9296/18.0/ios/18.0 (2025). (Accessed: 2025/06/15).

[2] Baba, T.: VIDVIP: Dataset for object detection during sidewalk travel, *J. Robotics Mechatronics*, Vol. 33, No. 5, pp. 1135–1143 (2021).

[3] Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H. and Lin, J.: Qwen2.5-VL technical report, *Computing Research Repository arXiv Preprint*, Vol. arXiv:2502.13923 (2025).

[4] Computer Science Institute Co., Ltd.: Eye Navi: Walking support app for the visually impaired, https://www.eyenavi.jp/en/ (2025). (Accessed: 2025/06/12).

[5] Hessel, J., Holtzman, A., Forbes, M., Bras, R. L. and Choi, Y.: CLIPScore: A reference-free evaluation metric for image captioning., *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7514–7528 (2021).

[6] Huh, M., Xu, F., Peng, Y., Chen, C., Murugu, H., Gurari, D., Choi, E. and Pavel, A.: Long-form answers to visual questions from blind and low vision people, *Computing Research Repository arXiv Preprint*, Vol. arXiv:2408.06303 (2024).

[7] International Agency for the Prevention of Blindness: International Agency for the Prevention of Blindness's vision atlas, https://www.iapb.org/ (2025). (Accessed: 2025/06/14).

[8] Mañas, O., Krojer, B. and Agrawal, A.: Improving automatic VQA evaluation using large language models, *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, pp. 4171–4179 (2024).

[9] OpenAI: ChatGPT (Feb 13 version) [Large language model], https://chat.openai.com/ (2023). (Accessed: 2025/06/14).

[10] OpenAI: GPT-4V(ision) System Card, https://cdn.openai.com/papers/GPTV_System_Card.pdf (2023). (Accessed: 2025/06/14).

[11] OpenAI: GPT-4o [Large language model], https://openai.com/product/gpt-4o (2024). (Accessed: 2025/06/14).

[12] OpenAI: GPT-o4-mini [Large language model], https://platform.openai.com/docs/models/gpt-o4-mini (2025). (Accessed: 2025/06/14).

[13] Papineni, K., Roukos, S., Ward, T. and Zhu, W.: BLEU: A method for automatic evaluation of machine translation, Technical report, IBM T. J. Watson Research Center (2002).

[14] Unity Technologies: Unity (game engine), Version 2023.2.0, https://unity.com/ (2025). (Accessed: 2025/06/14).

[15] Vedantam, R., Zitnick, C. L. and Parikh, D.: CIDEr: Consensus-based image description evaluation, *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4566–4575 (2015).

[16] Wiberg, H. J. and Erfurt, C.: Be My Eyes: Accessibility technology for blind low vision people, https://www.bemyeyes.com/ (2015). (Accessed: 2025/06/14).

[17] Wu, S., Fei, H., Qu, L., Ji, W. and Chua, T.: NExT-GPT: Any-to-any multimodal LLM, *Proceeding of the 41st International Conference on Machine Learning*, pp. 53366–53397 (2024).

[18] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q. and Artzi, Y.: BERTScore: Evaluating text generation with BERT, *Proceedings of the 8th International Conference on Learning Representations*, .43 pages (2020).