Where Am I? Exploring the Situational Awareness Capability of Vision-Language Models in Vision-and-Language Navigation

Anonymous ACL submission

Abstract

Intuitively, it is important for humans to localize themselves by understanding their surroundings when navigating to a place, especially when the trajectory is long and complex. Similarly, we believe that this kind of capability, which we call situational awareness, is also crucial for developing better navigational agents. This work aims to evaluate the situational awareness capability of current popular vision-language model (VLM) based navigational agents. Inspired by the way of humans processing observations, we consider two types of visual inputs to the models: 360-degree panoramic images and egocentric navigational videos. Then we construct a new dataset, *Situational Awareness Dataset (SAD)*, comprised of around 100K such panoramic images and videos and corresponding instructions for this task. We then evaluate multiple prominent VLMs including OpenAI o1, GPT-4o, Gemini 2.0 Flash, Qwen2.5-VL, and their finetuned versions on SAD. Our results show that the situational awareness capability of these models is far behind human performance, but can be significantly improved by further finetuning. Furthermore, our findings also suggest that fine-grained alignment between observations and instructions is very helpful to the vision-and-language navigation (VLN) task, which is somehow overlooked by the community now.

1 Introduction

Situational awareness is a broad concept referring to the capability of perception, comprehension, and projection of the elements in an environment (Endsley, 1995). This capability is crucial for effective decision-making in a variety of tasks, such as aviation and healthcare. Within the realm of vision-and-language navigation (VLN), we simplify this concept to denote an agent's capability to understand its current position based on the observations in the navigation. This understanding is typically the initial step for navigation agents in assessing their progress and making informed decisions. Although fundamental, achieving situational awareness still necessitates intricate spatial reasoning and a nuanced language grounding capability.

Recent advancements in large-scale vision-language models (VLMs) have demonstrated great potential across various vision-and-language tasks. Applying these models to the task of vision-and-language navigation in continuous environments (i.e., VLN-CE task; Krantz et al., 2020) using zero-shot learning has been a burgeoning area of research. Despite this interest, the performance of VLMs in this domain still lags far behind the methods that employ supervised learning. For instance, the state-of-the-art VLM-based method, AO-Planner (Chen et al., 2024a), achieves a 22.4% success rate on the RxR-CE dataset (Ku et al., 2020), whereas the popular supervised learning based method ETPNav (An et al., 2024) achieves 54.8%. Several factors contribute to this performance gap, with the situational awareness capability of these models being a fundamental determinant of their navigation performance. However, research on this capability within the VLN field remains limited. One major obstacle is the scarcity of fine-grained annotated data that aligns navigation instructions with observations in ground-truth trajectories.

To address this gap, we introduce a new dataset, the *Situational Awareness Dataset (SAD)*, which encompasses around 200,000 observations paired with instructions designed to evaluate situational awareness capabilities. Inspired by how humans localize themselves and navigate in a scene, we consider both types of 360-degree panoramic images and egocentric navigational videos as observation input in the dataset. These two completely different types of observations test situational awareness capability from different perspectives and pose different challenges for the models. The corresponding

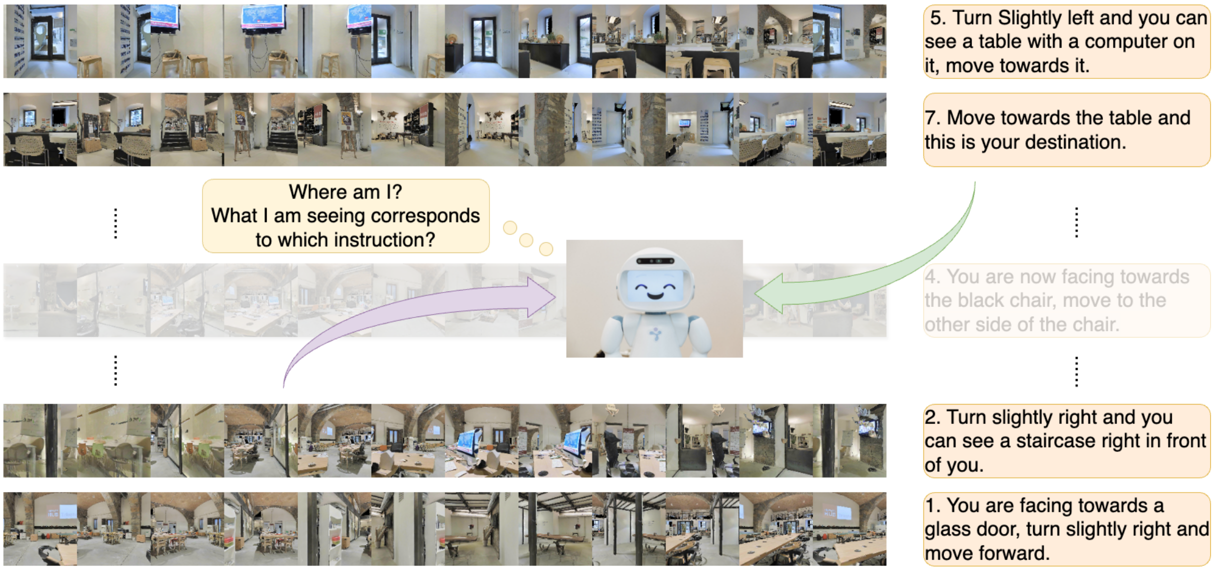

Figure 1: An example for our situational awareness task. The navigation agent takes as input a 360-degree panoramic image and the whole instruction. The agent is required to understand the surrounding observations and language instructions, then predict which sentence in the instruction the current observation corresponds to.

instructions in the dataset are available in three typologically diverse languages–English, Hindi, and Telugu–to facilitate the examination of capabilities within multilingual contexts.

We conduct evaluations of several prominent commercial and open-source VLMs in both zero-shot and finetuned settings to assess their situational awareness capability on SAD. The models tested include OpenAI o1, GPT-4o, Gemini 2.0 Flash, and Qwen2.5-VL-7B/72B-Instruct. These models are good representatives of the current state-of-the-art in both commercial and open-source VLM fields. Our findings reveal that even the most advanced model, OpenAI o1 and Gemini 2.0 Flash, perform very poorly in the zero-shot setting. But they can be significantly improved by more than 3 times through further finetuning, though still lagging a large gap behind human performance. Moreover, we further investigate whether the situational awareness capability can be helpful to the VLN task. The experimental results show an agent with better situational awareness capability also performs better in the VLN task.

2 Dataset and Evaluation Method

In order to streamline the evaluation process, we concentrate on the alignment between instructions and observations at the sentence level. This focus means we only assess the correspondence between the end of each instruction sentence and its associated observation.

2.1 Dataset

We construct the *Situational Awareness Dataset (SAD)* with the help of Habitat simulator and the existing RxR-CE dataset. The details of the construction process and the dataset can be found in Appendix §A.1. SAD contains instructions in three languages and the agent's observations corresponding to the end of each instruction. To simplify the task further, we limit our focus to instructions containing a maximum of 10 sentences. For each position, there are two types of observations: (1) a panoramic RGB image composed of 12 RGB sub-images captured from 12 different directions at equally spaced horizontal heading angles: $(0°, 30°, ..., 330°)$; (2) a video recording the agent's egocentric observations 10 steps before arriving at this position. We ensure that there are at least 5 steps of difference between each video.

2.2 Evaluation Method

With the constructed dataset, we evaluate the situational awareness capability of agents through a straightforward question-answering format. Given an instruction and the corresponding panoramic image or egocentric video observations, we pose the following question to the agent: "Which sentence in the instruction does this image/video correspond to the end of?" The agent must predict the sentence index that align with the observation (see Figure 1).

We utilize two metrics to assess the agent's performance on this task: (1) Instruction-Level Accu-

	ACC_INSTR	ACC_SENT
Panoramic image observations	65.00	87.14
Egocentric video observations	-	91.00

Table 1: Human performance (%) on the constructed SAD dataset with two types of observations.

racy (ACC_INSTR): this metric measures the accuracy over the whole instruction level. Only if the predictions for all observations in an instruction are correct, the instruction-level predictions are considered correct. We don't report this metric for the egocentric video observations, as the video dataset may not contain the whole sentences for an instruction in order to avoid large overlap between videos. (2) Sentence-Level Accuracy (ACC_SENT): this metric evaluates accuracy based on individual sentences in the instruction. Each correct prediction associated with an observation contributes to the overall accuracy.

2.3 Human Performance

To provide a human performance baseline on the SAD dataset, we randomly sample 200 instances from the dataset for both types of observations and have ten individuals perform the same situational awareness task respectively (see details in Appendix §C). The results show an average instruction-level accuracy of 65% and a sentence-level accuracy of 87% with the panoramic image observations. The performance with the egocentric video observations is a little higher, suggesting that humans better situate themselves based on videos of observation history.

3 Experiments

3.1 Evaluation Settings

Dataset We utilize our constructed SAD dataset for model evaluation. We test the models across all three language splits: English, Hindi, and Telugu. Each panorama sub-image and egocentric video is evaluated at a resolution of 224×224. Our preliminary experiments with GPT-4o indicate that higher resolutions do not significantly enhance performance while substantially increasing test time. Further details are provided in Appendix B.1.

Test Models We evaluate the following models on the SAD dataset in both zero-shot setting and finetuned setting: GPT-4o (gpt-4o-2024-08-06; OpenAI, 2024a), OpenAI o1 (o1-2024-12-17; OpenAI, 2024b), Gemini 2.0 Flash (DeepMind, 2025), and Qwen2.5-VL-7B/72B-Instruct (Qwen-Team, 2025). We run each model three times and report the average performance in each evaluation setting. For the zero-shot evaluation of panoramic image observation setting, all models employ the technique of structured outputs. Specifically, we force the model's output to include the reasoning steps for each image along with the final answer, formatted in JSON. Further details about the prompts we use are provided in Appendix B.2. For the finetuned setting, we use GPT-4o and Qwen2.5-VL-7B-Instruct models as the base models and finetune them on the SAD train set[1] for each type of observation.

3.2 Evaluation Results

Table 2 presents the evaluation results of the tested models with panoramic image observations on the SAD dataset. The approximate accuracy estimates for random guesses are 0.02% and 14.29%, respectively.[2] In terms of exact match instruction-level accuracy (ACC_INSTR), all models perform very poorly. Among them, OpenAI o1 emerges as the leader, outperforming others by approximately 50%. GPT-4o and Gemini 2.0 Flash exhibit similar performance levels, while the open-sourced Qwen2.5-VL-7B/72B-Instruct models perform the poorest. This suggests that the OpenAI o1 model demonstrates a superior comprehensive reasoning capability in understanding complete trajectories compared to the other models. For sentence-level accuracy (ACC_SENT), OpenAI o1 once again achieves the highest performance, though Gemini 2.0 Flash closely follows. The Qwen2.5-VL-7B/72B-Instruct models still lag significantly behind other models. Furthermore, the evaluation across different language splits reveals no substantial performance differences, suggesting consistent model capabilities across various languages. In addition, with only 10% of the training data, the finetuned GPT-4o model achieves a quite large performance boost, surpassing the zero-shot performance of all other models.

Table 3 presents the results of sentence-level accuracy with the egocentric video as visual input. Gemini 2.0 Flash achieves the best performance, even surpassing the finetuned Qwen2.5-VL-7B-

[1]We only use 10% training data for GPT-4o due to the 8GB upload limitation of OpenAI APIs.

[2]These values are calculated as $1/7! \times 100\% \approx 0.02\%$ and $1/7 \times 100\% \approx 14.29\%$, where 7 is the average number of images per example.

Models	English		Hindi		Telugu	
	ACC_INSTR	ACC_SENT	ACC_INSTR	ACC_SENT	ACC_INSTR	ACC_SENT
GPT-4o	6.36	26.74	4.29	25.55	8.15	27.76
OpenAI o1	11.61	32.92	17.18	37.62	15.99	37.47
Gemini 2.0 Flash	6.99	32.13	9.51	35.79	7.71	32.17
Qwen2.5-VL-7B-Instruct	2.84	18.25	4.29	20.94	3.97	21.53
Qwen2.5-VL-72B-Instruct	3.68	20.49	5.52	24.61	5.34	22.58
GPT-4o-Finetuned	**19.17**	**48.57**	**21.17**	**46.97**	**18.29**	**44.21**
Qwen2.5-VL-7B-Instruct-Finetuned	15.26	30.28	18.32	40.10	12.45	35.68

Table 2: Evaluation results with panoramic images as visual input on the SAD dataset. ACC_INSTR and ACC_SENT denote the instruction-level accuracy and sentence-level accuracy, respectively. All the results are averaged over three runs and reported in percentage. All the model without "Finetuned" suffix are evaluated in the zero-shot setting.

Models	English	Hindi	Telugu
Gemini 2.0 Flash	**43.00**	**47.58**	**39.31**
Qwen2.5-VL-7B-Instruct	12.18	14.58	18.06
Qwen2.5-VL-72B-Instruct	26.80	25.63	20.64
Qwen2.5-VL-7B-Instruct-Finetuned	42.72	45.38	30.88

Table 3: Evaluation results with egocentric videos as visual input on the SAD dataset. We only report ACC_SENT here.

Instruct model. This suggests that the Gemini 2.0 Flash model has a decent video understanding capability. Compared with using panoramic images as visual input, the performance with egocentric videos is generally better across all models, indicating that the models are more capable of situational awareness when provided with video observations.

3.3 Can Situational Awareness Capability Help the VLN Task?

Equipped with better situational awareness capability, can the model perform better in the VLN task? To answer this question, we conduct an experiment comparing the zero-shot performance of non-finetuned Qwen2.5-VL-7B-Instruct and the one finetuned through the situational awareness task with egocentric video observations for the VLN-CE task. We choose R2R-CE dataset for this experiment instead of RxR-CE to avoid the potential effects of training on RxR-CE dataset. Besides the current step's observation, we use at most 10 recent historical images as input. As shown in Table 4, using the finetuned model as the agent is better than using the non-finetuned version. Though the performance is still quite low compared to current SOTA baselines, the significant improvements can still demonstrate the usefulness of training with situational awareness task to the VLN task.

Agents	SR	SPL	Path Length
Random	6.50	6.49	0.21
Qwen2.5-VL-7B-Instruct	8.21	7.72	2.37
Qwen2.5-VL-7B-Instruct-Finetuned	11.26	9.27	2.51

Table 4: Impact of finetuning with the situational awareness tasks on the R2R-CE dataset.

4 Related Work

Situational Awareness The concept of situational awareness is extensively studied in the field of cognitive science, psychology, human factors, aviation, healthcare, and more (Munir et al., 2022; Endsley, 2021; Stanton et al., 2001). Recently, Berglund et al. (2023) studies the emergence of situational awareness in large language models (LLMs). We further specify this concept in the context of VLN task in this work.

VLN with LLMs and VLMs The VLN task is a representative research topic in the field of embodied AI, and how to make use of LLMs and VLMs to solve this task has attracted much attention (Zhou et al., 2024; Chen et al., 2024b; Long et al., 2024; Zhang et al., 2024; Lin et al., 2024; Chen et al., 2023; Cai et al., 2024; Chen et al., 2024a; Qiao et al., 2024). However, little work studies the fundamental situational awareness capability of these models. This work aims to study such capability.

5 Conclusion

This work presents the situational awareness task and a corresponding dataset SAD with two types of visual observations. Our findings based on evaluations of multiple prominent VLMs suggest that the situational awareness capability of these models is still limited, and improving such capability can benefit the performance in VLN tasks.

6 Limitations

Our work has several limitations. First, the format of the evaluation is a simple question-answering task, which may not fully capture the situational awareness capability of vision-and-language models and may not be directly applied to evaluate the agents trained with supervised learning. Second, we show that the situational awareness capability is helpful to the VLN task, but we only study the zero-shot setting. Future work could explore enhancing the trained vision-language-action agents such as NaVid with the situational awareness capability in VLN-CE tasks.

Use of AI Assistance We used AI assistant tools (ChatGPT and GitHub Copilot) to aid in rewriting code and text. All AI-generated content was thoroughly reviewed and verified by the authors. AI was not used to generate new research ideas or original findings; rather, it served as a support tool to improve clarity, efficiency, and organization. In accordance with ACL guidelines, our use of AI aligns with permitted assistance categories, and we have transparently reported all relevant usage in this paper. While AI contributed to enhancing the quality of the work, no direct research outputs are the result of AI assistance.

References

Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. 2024. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Lukas Berglund, Asa Cooper Stickland, Mikita Balesni, Max Kaufmann, Meg Tong, Tomasz Korbak, Daniel Kokotajlo, and Owain Evans. 2023. Taken out of context: On measuring situational awareness in llms. *arXiv preprint arXiv:2309.00667*.

Wenzhe Cai, Siyuan Huang, Guangran Cheng, Yuxing Long, Peng Gao, Changyin Sun, and Hao Dong. 2024. Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5228–5234. IEEE.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*.

Jiaqi Chen, Bingqian Lin, Xinmin Liu, Lin Ma, Xiaodan Liang, and Kwan-Yee K Wong. 2024a. Affordances-oriented planning using foundation models for continuous vision-language navigation. *arXiv preprint arXiv:2407.05890*.

Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee Wong. 2024b. Mapgpt: Map-guided prompting with adaptive path planning for vision-and-language navigation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9796–9810.

Peihao Chen, Xinyu Sun, Hongyan Zhi, Runhao Zeng, Thomas H Li, Gaowen Liu, Mingkui Tan, and Chuang Gan. 2023. A$\hat{2}$ nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. *arXiv preprint arXiv:2308.07997*.

DeepMind. 2025. Gemini 2.0 flash.

Mica R. Endsley. 1995. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64.

Mica R Endsley. 2021. Situation awareness. *Handbook of human factors and ergonomics*, pages 434–455.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. *Beyond the Nav-Graph: Vision-and-Language Navigation in Continuous Environments*, page 104–120. Springer International Publishing.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-Across-Room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Conference on Empirical Methods for Natural Language Processing (EMNLP)*.

Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. 2024. Navcot: Boosting llm-based vision-and-language navigation via learning disentangled reasoning. *arXiv preprint arXiv:2403.07376*.

Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. 2024. Discuss before moving: Visual language navigation via multi-expert discussions. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17380–17387. IEEE.

Arslan Munir, Alexander Aved, and Erik Blasch. 2022. Situational awareness: techniques, challenges, and prospects. *AI*, 3(1):55–77.

OpenAI. 2024a. Hello gpt-4o.

OpenAI. 2024b. Learning to reason with llms.

Yanyuan Qiao, Wenqi Lyu, Hui Wang, Zixu Wang, Zerui Li, Yuan Zhang, Mingkui Tan, and Qi Wu. 2024. Open-nav: Exploring zero-shot vision-and-language navigation in continuous environment with open-source llms. *arXiv preprint arXiv:2409.18794*.

QwenTeam. 2025. Qwen2.5-vl.

Neville A Stanton, Peter RG Chambers, and John Piggott. 2001. Situational awareness and safety. *Safety science*, 39(3):189–204.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. 2024. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*.

Gengze Zhou, Yicong Hong, and Qi Wu. 2024. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7641–7649.

A Dataset

A.1 Dataset Construction

We develop the Situational Awareness Dataset (SAD) using the Habitat simulator by leveraging the existing RxR-CE dataset. The RxR-CE dataset is a large-scale multilingual vision-and-language navigation resource featuring 126,000 navigation instructions and demonstrations within Matterport3D (Chang et al., 2017) and Habitat environments. To construct SAD, we utilize both the standard annotation task data and extended pose trace data from the RxR-CE dataset. The annotation task data includes essential components for vision-and-language navigation, such as navigation instructions and reference paths. It also provides a "timed_instruction" field, indicating the start and end times of words or phrases in alignment with the recording. The extended pose trace data offers snapshots detailing the virtual camera parameters and field-of-view from the annotators' perspectives.

We load this dataset into the Habitat simulator and calculate the camera poses and corresponding timestamps based on the supplied camera extrinsic matrix data. By extracting the timestamp of the concluding word in each instruction sentence from the "timed_instruction" data, we align these timestamps with the camera pose data, thereby obtaining the corresponding observations within the Habitat simulator.

For each position's observation, we render a panoramic RGB image composed of 12 RGB sub-images captured from 12 different directions at equally spaced horizontal heading angles: $(0°, 30°, ..., 330°)$. These sub-images are generated in three resolutions: 224×224, 480×480, and 1024×1024. To simplify the task further, we limit our focus to instructions containing a maximum of 10 sentences. More detailed information about the dataset is provided in Table 5.

A.2 Dataset Details

The number of examples in the training, validation, and test splits of the SAD dataset is shown in Table 5. The dataset is divided into three language splits: English, Hindi, and Telugu.

Languages	Panoramic image observation			Egocentric video observation		
	Train	Val	Test	Train	Val	Test
English	10,609	1,210	1,904	58,448	8,761	10,747
Hindi	1,642	202	381	8,509	1,023	1,818
Telugu	10,016	1,141	2,175	40,391	6,961	9,375

Table 5: Statistics of the SAD dataset. The dataset is divided into three language splits. There are two types of observations: panoramic images and egocentric videos.

B Experiments

B.1 Effects of Different Image Resolutions

We study the effects of different image resolutions on the performance of GPT-4o on our proposed SAD dataset. We evaluate the model on three different image resolutions: 224×224, 480×480, and 1024×1024. The results are shown in Table 6. We find that the higher resolutions do not bring significant improvement in the performance while significantly increasing the test time. Therefore, we use the image resolution of 224×224 for evaluation in the main experiments.

Image Resolution	ACC_Instr	ACC_Sent	Inf. Time
224×224	6.36	26.74	30min
480×480	7.36	26.78	52min
1024×1024	6.93	26.85	20.5h

Table 6: Effects of image resolutions on the performance of GPT-4o on our proposed SAD dataset.

B.2 Prompts

We present the prompts we use for the VLMs in the following code snippet (see Listing 1 for the panoramic image observations and Listing 2 for the egocentric video observations). It contains the system prompt and the user prompt. We also use the technique of structured outputs to force the model to output the reasoning steps and answers in a json format. We use the same prompts for all the models we evaluate in this work.

C Human Evaluation on the SAD Dataset

We conduct a human evaluation on the SAD dataset in order to assess its quality and find potential problems during the automatic data generation process.

We randomly sample 200 English examples from the test split of the dataset and send them to 10 people who are fluent speakers of English and have at least bachelor degrees. Each participant finished 120 examples in five days. We only send them the questions without giving them the answers. We make sure that every question is sent to three different people. After receiving their results, we use a script to check their correctness and calculate the final results.

For the IRB approval, no ethics review board approval was sought for this study because the human evaluations were designed solely to collect anonymous, non-identifiable responses, did not involve challenging psychological content, and imposed no obligations on participants - criteria that, under current guidelines, do not warrant formal ethical oversight.

```python
class SingleImageStep(pydantic.BaseModel):
    explanation: str
    answer: int


class SituationalAwarenessOutput(pydantic.BaseModel):
    number_of_input_images: int
    reasoning_steps: list[SingleImageStep]
    answer: list[int]

SYSTEM_PROMPT = inspect.cleandoc(
    """You are an agent navigating through a virtual environment according to
    the given instruction. But now your task is not to navigate, but to predict
    the positions of the given observation images in the corresponding
    instruction.  You would be given a set of images and an corresponding
    instruction.  The given images are the RGB {image_type} observation of your
    current position. Each panoramic image is comprised of 12
    sub-egocentric-images, where each sub-image corresponds to a different
    direction.  You need to think of where the position is in the instruction.
    The entire instruction is comprised of multiple sub-instructions.  Each
    sub-instruction starts with '#' followed by a number, which is the index of
    the sub-instruction.  Each position is the end of each sub-instruction.  So
    your task is to predict at the end of which sub-instruction you could see
    the current given image.  Note that the number of input images are strictly
    equal to the number of sub-instructions. Moreover, There will not be two
    images corresponding to the same position.  Your final answer should be a
    list of integers, where each integer represents that image's positions in
    the instruction.   For example, "[2, 3, 1, 4]" means you would observe the
    first input image at the end of the second sub-instruction, the second
    input image corresponds to the end of the third sub-instruction, the third
    input image corresponds to the end of the first sub-instruction, and the
    fourth input image corresponds to the end of the fourth sub-instruction.
    """
).replace("\n", " ")

USER_PROMPT = inspect.cleandoc(
    """Given the following {num_input_images} images, please predict their
    observation positions in the instruction.  The instruction is:
    {instruction_with_index}"""
).replace("\n", " ")


response = client.beta.chat.completions.parse(
    model=test_model,
    messages=[
        {
            "role": "system",
            "content": [
                {
                    "type": "text",
                    "text": SYSTEM_PROMPT.format(image_type=image_type),
                }
            ],
        },
        {
            "role": "user",
            "content": [
                {
                    "type": "text",
                    "text": USER_PROMPT.format(
                        num_input_images=len(multiple_images_input),
                        instruction_with_index=instruction_with_index,
                    ),
                }
            ]
            + multiple_images_input,
        },
    ],
    response_format=SituationalAwarenessOutput,
```

```
)
```

Listing 1: Prompts of OpenAI APIs for panoramic images as input.

```
USER_PROMPT = inspect.cleandoc(
    f"""
    <video>
    You are an agent navigating through a virtual environment according to
    the given instruction. But now your task is not to navigate, but to
    predict the positions of the given observation videos in the
    corresponding instruction.  You would be given a video and an
    corresponding instruction.  The given video are your most recent RGB
    observation while moving to your current position.
    You need to think of where your current position is in the
    instruction.  The entire instruction is comprised of multiple
    sub-instructions.  Each sub-instruction starts with '#' followed by a
    number, which is the index of the sub-instruction.  Each position is
    the end of each sub-instruction.  So your task is to predict at the end
    of which sub-instruction you are moving to in the given video.
    Your final answer should be an integer, which represents the
    sub-instruction index.
    The instruction is as follows:
    <INSTRUCTION> {instruction} </INSTRUCTION>
    Which instruction index is the answer?
    """
).replace("\n", " ")


video_messages = [
    {"role": "system", "content": "You are a helpful assistant."},
    {
        "role": "user",
        "content": [
            {"type": "text", "text": USER_PROMPT},
            {
                "type": "video",
                "video": video_path,
                "total_pixels": 20480 * 28 * 28,
                "min_pixels": 16 * 28 * 2,
                "fps": 1.0,
            },
        ],
    },
]

video_messages, video_kwargs = prepare_message_for_vllm(video_messages)

n_try_times = 1
while True:
    try:
        chat_response = client.chat.completions.create(
            model=model_path,
            messages=video_messages,
            extra_body={"mm_processor_kwargs": video_kwargs},
        )
    except Exception as e:
        logger.error(f"Error during vLLM prediction: {e}. Retrying ...")
        if n_try_times > 3:
            logger.error("Max retry attempts reached. Skipping this example.")
            break
        else:
            n_try_times += 1
            time.sleep(2)
            continue
    break
```

Listing 2: Prompts of OpenAI APIs for videos as input.