Memory-Maze: Scenario Driven Visual Language Navigation Benchmark for Guiding Blind People

Masaki Kuribayashi*[1,2], Kohei Uehara*[2], Allan Wang[2,3], Daisuke Sato[3], Renato Alexandre Ribeiro[2], Simon Chu[3], and Shigeo Morishima[1]

***Abstract*—Visual Language Navigation (VLN) powered robots have the potential to guide blind people by understanding route instructions provided by sighted passersby. This capability allows robots to operate in environments often unknown a prior. Existing VLN models are insufficient for the scenario of navigation guidance for blind people, as they need to understand routes described from human memory, which frequently contains stutters, errors, and omissions of details, as opposed to those obtained by thinking out loud, such as in the R2R dataset. However, existing benchmarks do not contain instructions obtained from human memory in natural environments. To this end, we present our benchmark, Memory-Maze, which simulates the scenario of seeking route instructions for guiding blind people. Our benchmark contains a maze-like structured virtual environment and novel route instruction data from human memory. Our analysis demonstrates that instruction data collected from memory was longer and contained more varied wording. We further demonstrate that addressing errors and ambiguities from memory-based instructions is challenging, by evaluating state-of-the-art models alongside our baseline model with modularized perception and controls.**

***Index Terms*—Vision-Based Navigation, Performance Evaluation and Benchmarking, Human-Centered Automation**

I. Introduction

VISUAL language navigation (VLN) is a task where an agent with visual access to the surroundings navigates under a human's instructions [1]. Recently, navigation robots for blind people have been developed to help them gain independence [2], [3], [4], such as robots that allow users to choose destinations within prebuilt maps [2], [3]. One scenario in which such robots would benefit from the VLN technology is where blind people request instructions to their destinations from sighted passersby in unfamiliar buildings [5]. In this scenario, the VLN technology deployed on navigation robots may assist their blind users by understanding verbal instructions from the passersby and then autonomously guiding them to their destinations. VLN technology could also allow robots to operate autonomously without relying on building infrastructure or prebuilt maps, which is crucial for allowing robots to assist blind people in navigating various new environments [4], [6].

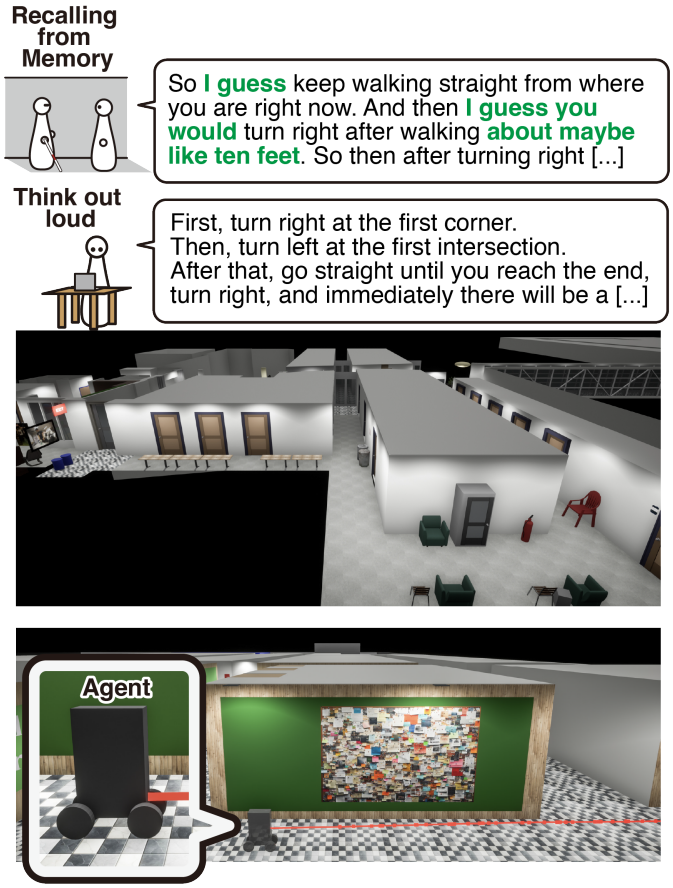


Fig. 1: **Memory-Maze Benchmark.** Top: the instructions obtained in the memory-based scenario contain unique phrases, highlighted in green, in contrast to those collected in traditional think-out-loud settings. Middle: Our benchmark environment based on the CARLA simulator [7]. Bottom: the VLN agent that navigates within the environment.

However, direct application of existing VLN models to the blind people navigation scenario is currently limited, as there is a need for a benchmark that reflects the blind users' demands realistically. Many VLN tasks have been addressed in environments such as static houses [1] or roadways [8]. Nonetheless, it is also most important for blind individuals to navigate large public spaces such as shopping malls or university hallways. Compared to existing environments, these environments are characterized by physical turning points and intersections, resembling a maze. Besides the environmental difference, in existing VLN literature, natural language in-

This work was supported by Waseda Research Institute for Science and Engineering and JSPS KAKENHI (23KJ2048 and 21H05054).

*Masaki Kuribayashi and Kohei Uehara contributed equally to this work.

[1]Masaki Kuribayashi and Shigeo Morishima are with Waseda University, Japan (rugbykuribayashi@toki.waseda.jp and shigeo@waseda.jp).

[2]Masaki Kuribayashi, Kohei Uehara, Allan Wang, and Renato Alexandre Ribeiro are with Miraikan - The National Museum of Emerging Science and Innovation, Japan ({ masaki.kuribayashi, kouhei.uehara, allan.wang, renato.ribeiro}@jst.go.jp).

[3]Allan Wang, Daisuke Sato, and Simon Chu are with Carnegie Mellon University, United States ({allanwan, daisuke, cchu2}@andrew.cmu.edu).

structions are provided by thinking out loud. In other words, annotators visually navigate a virtual environment and type out instructions for constructing routes concurrently. In our scenario, sighted passersby must describe the route from their memory, which often contains errors such as inaccurate estimates of distances, hallucinations of landmark objects, and omissions of key turning points. To the best of our knowledge, our benchmark is the first to address the scenario of a blind user seeking memory-based instructions from sighted passersby in maze-like public spaces.

We present *Memory-Maze* (Fig. 1), a benchmark that reflects the blind user navigation scenario. Memory-Maze contains virtual environments of real-world public spaces. It is based on CARLA [7], which enables us to simulate various sensor data (*e.g.*, LiDAR) from robots. It also contains instructions data gathered from two studies from sighted individuals. In the first study, instructions were gathered through online questionnaires by observing walk-through videos from a first-person perspective. This is similar to the annotation method used in existing research. In the second study, instructions were collected in-person by asking sighted passersby to describe the same routes from their memories. This reflects the novel scenarios envisioned in our benchmark. We observed different characteristics among the two studies in terms of length, number of errors, variety, among others.

To analyze the difficulty of our benchmark, we developed a VLN baseline model better designed to navigate in large public spaces, by leveraging modular APIs to handle navigation control and perceptions. Our model also fulfills two requirements for the practical deployment of VLN models for blind people: zero-shot transfer to unseen environments without navigation graphs and single inference. Navigation robots need to be used in unseen environments for blind people, directly applying existing supervised models poses a challenge due to their limited performance in unseen settings [9]. Additionally, existing models perform repeated iterative inferences during navigation, resulting in frequent stops and prolonged navigation time. Leveraging large language models' (LLM) potential for zero-shot generalization in unseen environments, our single-inference LLM-powered model converts the instruction into Python code based on the defined robot control API (Sec. III-B) for route navigation. This code generation approach modularizes low-level commands such as path-planning for collision avoidance and intersection detection, and serves as a baseline that focuses more on the language interpretation and reasoning capabilities of VLN. Through the study with our model and the current state-of-the-art methods [10], [11], we demonstrated the difficulty of our benchmark and a tendency that real-world memory-based instructions are more difficult for VLN models to handle.

We summarize our contributions below.

1) We constructed Memory-Maze, a benchmark containing virtual environments of a large public spaces, and gathered two sets of instructions, one collected by thinking out loud and one obtained from human memory.
2) Through an experiment with current state-of-the-art models and our baseline VLN model, we revealed the gap between the instructions collected based on memory and those collected by thinking out loud.

Our benchmark and codes are available at https://github.com/chestnutforestlabo/MemoryMaze

II. Related Work

A. Assistive Navigation Systems for Blind People

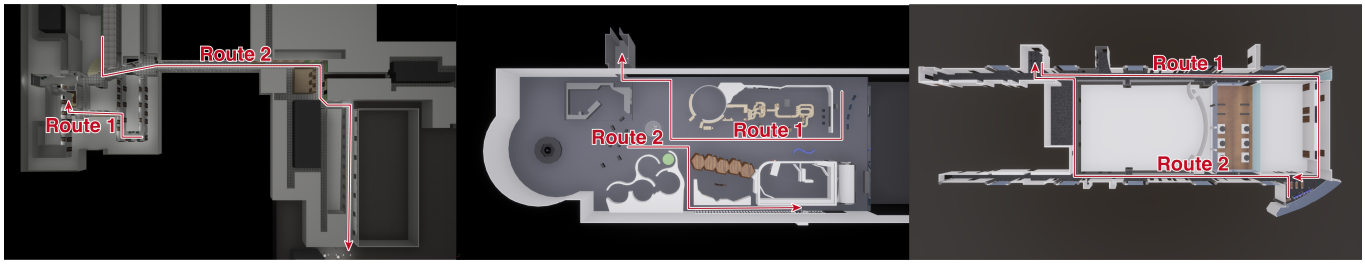
Recently, navigation robots have been explored to aid blind people in avoiding obstacles while navigating. A common practice is to prepare prebuilt maps and infrastructure for localization and manual destination selection [2], [3]. This practice poses a limitation for these systems, as prebuilt maps and infrastructure are costly to obtain and maintain. Consequently, a map-less approach was also proposed [4], [12]. For example, the PathFinder [4] system allows users to input navigation directions through the buttons on the robot's handle based on instructions from sighted passersby [5]. However, because the system needs users to understand and memorize the instructions, high cognitive loads are placed on the users. To address this, we present a novel, practical benchmark for VLN models that aims to interpret memory-based instructions from sighted passersby and navigate users autonomously to their destinations.

B. Benchmarks in VLN tasks

The VLN task has been conducted in various benchmarks, ranging from indoor [1], [13] to outdoor [8] settings. Most of the instruction annotations of these benchmarks were created by annotators who typed while concurrently observing a virtual environment or by researchers who constructed them manually. This way of obtaining instructions is not suitable for our purpose, as it does not reflect the scenario of people describing routes from their memories. Researchers have also explored benchmarks with longer routes for long-horizon navigation tasks [14]. Still, existing benchmarks do not feature large public areas where blind people navigate, such as shopping malls or university hallways. These areas contain both static and dynamic obstacles and are characterized by the existence of turning points and intersections (Fig. 2). A related benchmark is Touchdown [8], which also emphasizes navigation through intersections and dynamic environments. However, its map structure is represented by a navigation graph (*i.e.*, an undirected graph that represents navigable points with nodes), whereas the Memory-Maze assumes no prior knowledge such as navigation graphs.

C. VLN Models

Researchers have explored solutions for VLN tasks using supervised models [1], which learn from a sequence of observations and actions to take. These supervised models often do not transfer well in unseen environments [9]. With the recent advancements in LLM, researchers have also explored methods that do not require retraining [10], [15], [16]. One such approach was to use LLMs to extract landmarks from instructions and follow chronologically [15]. Another approach was to utilize LLM to flexibly determine actions at



(a) University (b) Museum 5F (c) Museum 7F

Fig. 2: **Bird's-Eye Views of Memory-Maze.** The benchmark contains three environments. The university includes features such as classrooms, offices, hallways, a kitchen, and a library. The 5th floor of the museum mainly contains exhibits. The 7th floor contains conference rooms, hallways, and a terrace area. Each environment includes two routes, totaling six routes. In the on-site study, participants were asked to describe the route from the starting point to the end point, thus, their descriptions may vary from the visualized route.

each step. NavGPT [10] is a model that uses LLM iteratively to select the node to navigate to within a navigation graph. Additionally, researchers have explored approaches that utilize the code generation capability of LLM [17], [18]. In the method proposed by Biggie *et al.* [18], given a prebuilt 3D map, images from their robot, and a Python API, the model generates codes that locate a target object [19], maps the object's location on the 3D map, and navigates to the mapped location. While these methods are effective when the given instructions include sufficient landmarks, instructions recalled from memory often contain insufficient landmarks, potentially leading to failure. Furthermore, these methods are limited by the need for a navigation graph or 3D map, which is difficult to construct for every unseen environment. To eliminate this requirement, models have been proposed to predict navigation graphs [20] or low-level actions [11] iteratively. However, the need for iterative inference prolongs inference time, which may affect navigation by not reacting to dynamic obstacles responsively. Moreover, iterative inference may be impractical in large public spaces, where the model could be required to perform over many inference steps, due to the need to process long time-horizon data. Our model utilizes LLM to produce navigation codes that follow a specified path in a single iteration, and allows flexible integration of low-level planning algorithms for obstacle avoidance. This direct generation of navigation codes, coupled with existing low-level planning algorithms, allows operation without navigation graphs.

III. Memory-Maze

Here, we describe our benchmark's virtual environment and the robot simulation program. To simulate our scenario, we selected a floor of a university building and two floors in a museum building (Fig. 2), which is characterized by the existence of multiple turning points.

A. Selecting and Building the Simulator

To simulate a scenario where a robot guides a blind person, it is necessary to simulate high-fidelity egocentric visuals that are realistic enough to run an image recognition algorithm. Thus, we built a novel virtual environment from scratch on top of the CARLA [7] simulator. While primarily developed for autonomous driving simulations, CARLA's flexibility and compatibility with the Unreal Engine allowed us to create a detailed 3D model of the experimental site. CARLA also offers the ability to configure the existence of static and dynamic obstacles and to simulate various sensors like RGB cameras, depth sensors, and LiDAR sensors. We created a 3D model of the experimental site using Fusion 360 and imported it into CARLA. This 3D model accurately reproduces the experimental site, both visually and in terms of floor layout. It also includes major objects along the route (doors, chairs, a statue, *etc*).

B. Implementation of the Control Program

Our next step was to develop a control program for the robot in the simulator to be used by our baseline VLN model. Utilizing CARLA's Python API to control the navigation robot, we implemented various control functions. We describe four major functions implemented.

We implemented functions for the agent to move forward (`move_forward(distance)`), find a turning point (`detect_turning_point()`), and turn (`turn(direction)`) using CARLA's `vehicle.apply_control` API. When using the `move_forward(distance)` function, to ensure the robot moves along the path without colliding with walls, we implemented a feature that makes the robot navigate as closely to the center of the corridor as possible. We calculate the central path based on the coordinates of the four corners of the corridor in the 3D model. The central path tracking is realized through PID control, which adjusts the robot's steering angles. When the `detect_turning_point()` function is used, it determines if the robot is in the pre-annotated areas of turning points and returns navigable directions if the robot is in one of them. Once the robot is at the turning point, it could change its direction using the `turn(direction)` function. Because component algorithm development of the control program was beyond the scope of this study, coordinates of the corridor's corners and the turning point areas are acquired from the virtual environment, reducing errors from noise in perception or control, and focusing on executing instructions. However, these can be obtained using prior methods [4].

TABLE I: **Data Analysis.** The table presents the route length (RL), mean, median, and standard deviation (SD) of word counts in the collected instructions, and their failure rates (FR). For the onsite instructions, we also report the alternative rate (AR), the rate of describing alternative routes.

	Route	RL	Iteration	Mean	Median	SD	FR	AR
Online Think-Out-Loud Instruction	University R1	40.27m	1	51.8	47.0	17.8	0.0%	-
			2	69.8	64.0	19.9	0.0%	-
	University R2	156.68m	1	81.3	81.0	24.9	9.09%	-
			2	98.3	99.0	31.2	3.03%	-
	Museum 5F R1	71.18m	1	81.4	78.0	36.3	17.39%	-
			2	88.9	90.0	32.3	17.39%	-
	Museum 5F R2	44.05m	1	60.1	53.5	21.5	9.09%	-
			2	71.0	61.0	33.9	4.55%	-
	Museum 7F R1	86.10m	1	98.2	91.5	42.5	13.64%	-
			2	96.7	90.0	42.2	18.18%	-
	Museum 7F R2	79.40m	1	71.3	68.0	25.8	4.35%	-
			2	95.0	85.0	47.4	0.00%	-
Onsite Memory-Based Instructions	University R1	40.27m	1	73.9	74.5	36.6	25.0%	10.0%
			2	102.9	94.5	51.1	25.0%	10.0%
	University R2	156.68m	1	131.0	115.5	73.2	40.0%	15.0%
			2	147.3	143.0	65.0	35.0%	15.0%
	Museum 5F R1	71.18m	1	68.2	64.0	27.4	76.19%	0.0%
			2	97.1	92.0	27.0	9.52%	0.0%
	Museum 5F R2	44.05m	1	65.5	51.0	42.7	45.45%	59.1%
			2	83.4	68.5	39.7	4.55%	63.6%
	Museum 7F R1	86.10m	1	68.7	69.5	27.5	54.54%	4.5%
			2	89.0	84.0	24.0	13.64%	9.0%
	Museum 7F R2	79.40m	1	79.5	69.0	40.0	52.38%	85.7%
			2	99.0	96.0	37.5	23.81%	90.1%

Additionally, we implemented an image recognition module `detect_from_RGB_image(object)`, which outputs bounding boxes of all detected objects, to manage landmark-related instructions such as *"turn after finding a chair."* While most existing object detection models are designed to identify objects from predefined classes, they are not capable of detecting arbitrary objects. Therefore, we used Grounding DINO [21], an open-vocabulary object detection model. Open-vocabulary object detection models output bounding boxes for any object by using the object's name as a query. With the object detection model selected, we then used CARLA's robot ego-centric RGB sensors to capture images. To address tasks requiring the robot to identify an object multiple times (*e.g.*, *"turn after passing four doors"*), we added tracking algorithms to avoid counting the same object in different frames as distinct entities. We further assume that in instructions that require finding landmark objects, the objects are located in close vicinity. For example, in the instruction *"turn after finding [object],"* the camera may capture the object at a considerable distance, but such instructions typically mean the object is close to the robot. To achieve this, we used CARLA's depth sensors to measure the distance to each detected object and filtered out those beyond four meters, ensuring that only nearby objects were considered.

IV. Instruction Data Collection

A. Procedure

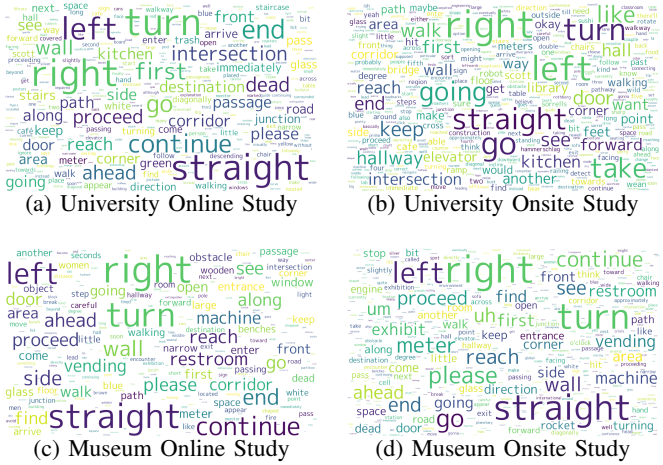
We conducted two studies, one online and one onsite, to collect natural language instruction data for routes at three locations: a floor across three buildings in a university and two floors in a museum. We designed the route as shown in Fig. 2. The studies were approved by our institutional review board (IRB), and informed consent was obtained from all participants. For each route, we obtained two rounds of instructions: one asking participants to describe the route to a blind person with a navigation robot naturally (first iteration) and another asking participants to describe the route after providing them with a brief description of the capability of the navigation robot (second iteration). The second instruction was collected to obtain more accurate memory-based instructions given by passersby. This was achieved by explaining the robot's capability (*e.g.*, being able to detect objects) to the participants. It simulates a scenario where a blind user may provide sighted passersby with robot information to obtain refined instructions. We expect that telling them about robots' capabilities would enable VLN models to achieve better performance.

In the first study, participants completed an online questionnaire designed to gather instructions that were similar to those in prior works. They were first presented with a scenario in which they communicated with a blind person accompanied by a navigation robot capable of following natural language instructions and 360° video walkthroughs of two routes. They were then asked to type instructions to the destination. They were allowed to re-watch the walkthrough videos at any time. We collected four instructions per participant. In total, 78 participants participated in the study, resulting in 312 instructions. The participants were gathered through university recruitment or through an online survey platform, and all were unfamiliar with the shown routes. The study was conducted in Japan, and the instructions were translated into English using GPT-4.

In the second study, we conducted an onsite in-person study. The aim of this study was to gather data that reflects the realistic scenario of sighted people describing the route from their memory. Thus, they did not watch the walkthrough video or experience the route during the study. The experimenter roleplayed as blind individuals, asked them for directions to the route destinations, and instructed them to describe the route verbally in two rounds. For the first iteration, we asked participants to describe the routes as naturally as possible. For the second iteration, to obtain more accurate instructions for the benchmark, in addition to explaining the robot's capabilities, the experimenter pointed out errors in the participants' given instructions, such as a missing turn, and asked them to explain the route again. For the university routes, we recruited sighted passersby and ensured that all participants were familiar with the route by using a pre-study check survey. In this study, each participant described a single route, resulting in two instructions per participant. In total, 40 participants participated in the study at the university, contributing 80 instructions. For the museum routes, we recruited staff or recent visitors who were familiar with the museum layouts. In this study, each participant described two routes, resulting in four instructions per participant. In total, 43 participants participated in the study at the museum, contributing 172 instructions.

B. Benchmark Analysis and Statistics

The mean, median, and standard deviation (SD) for the length of collected instructions are reported in Table I. First, we observed that the mean length and SD are longer for the second iteration in most cases, as participants tended to add



(a) University Online Study (b) University Onsite Study

(c) Museum Online Study (d) Museum Onsite Study

Fig. 3: **Word Clouds.** The onsite instruction data contains unique phrases that come from talking while recalling from the memory, such as *"uh," "maybe,"* and *"okay."*

more information on the second iteration. Also, we observed a tendency for instructions collected onsite to have higher lengths and more length variation. This is because, in the online study, participants described relevant and mostly accurate information about landmarks and turning points, while in the onsite study, many participants tried to be descriptive, relying on their memory, such as adding audio, olfactory cues, and conversational phrases such as *"I'm not 100% sure about this, but I think..."*.

The average instruction length and route distance in our benchmark are greater than those in previous datasets. For example, the R2R dataset includes instructions averaging approximately 30 words and route distances of about 10 meters [1], and the RxR dataset features instructions averaging 78 words and route distances of 14.9 meters [22].

The word clouds of the collected instructions are shown in Fig. 3. For the university environment, although the samples collected in the onsite study are fewer, they include 521 different words compared to the 381 words found in the samples from the online study. The same trend was noted in the museum environment, with 611 different words found in the onsite study and 586 words in the online study. This shows the greater diversity in the instructions' wording when described from memory. Although the instructions from the online study were translated using LLM, we believe that these results hold in the instructions' original language.

In Table I, we also manually analyzed each instruction to determine if it contained significant errors, *i.e.*, the number of failures in describing the route correctly. One author first conducted an initial failure review, after which multiple authors engaged in a discussion to reach a consensus on all samples. Online think-out-loud instructions were classified as failures for reasons such as turning in the wrong direction; instructing a turn at the incorrect turning point; and suggesting unnecessary extra turns. For onsite memory-based instructions, the reasons for the failures were containing extra turns, directing to an incorrect direction, leading to a wrong destination, lacking essential turn information, turning to incorrect directions at a turn, and containing inaccurate environmental details.

Interestingly, while examining the instructions, we realized that in the real world, humans may be able to correct errors in the instructions. For example, according to some passersby, the robot should go through a corridor between the hexagon exhibitions and the rectangular exhibition immediately to the right for museum 5F R2. However, there is actually no corridor between these two exhibitions. But it is possible to imagine where the nonexistent corridor might lead and try to find a detour. Being able to recognize errors in memory-based instructions is vital for aiding blind people to follow instructions provided by passersby.

As shown in Table I, during the onsite study, we observed that participants sometimes described alternative routes compared to those we had initially anticipated and illustrated in Fig. 2, when they described the routes from their memory. Surprisingly, some participants described a different route in the second iteration compared to their initial description. This highlights the potential of humans to describe alternative routes in real-world scenarios and the need for VLN models to perform equally well in these alternative routes, thereby underscoring the naturalness of our benchmark.

In summary, our benchmark captures real-world complexity beyond mere geometric turns. We found that even routes with 90° turns are accompanied by richly varied natural language instructions. For example, $\sim$25% of participants described the two consecutive turns near the goal at Museum 5F R2 as *"proceed in the front right direction"*. In another example, one instruction described the consecutive turns in University R2 with only semantic cues such as *"[...], I believe and then you walk along the hallway connecting DEF hall and the GHI hall and then you'll arrive at your destination."* Memory-Maze provides a unique scenario-based evaluative environment by incorporating variation in intersection counts, route lengths, and landmark complexity.

V. Baseline VLN Model Implementation

Our baseline VLN model uses the control API specification from our benchmark so that we may focus more on its language interpretation and reasoning capabilities. First, we utilized LLM's capability to generalize to various tasks and comprehend complex natural language instructions, so that no additional training is required when deployed in a new environment. Second, our method requires only a single inference iteration to generate low-level navigation code for robot control, in contrast to existing models that perform multiple inferences during navigation, which may prolong navigation time. It also eliminates the need for a navigation graph by generating codes that directly interface with low-level navigation modules. The generation of navigation code potentially leads to the flexibility of integrating existing, well-established methods into various modules, such as for obstacle avoidance [2] or turning point detection [4], [6].

We define the following as inputs to the agent: *natural language instruction*, the *sensor input* which includes the details obtained from sensors, *API specification* which consists of the commands and their explanations in Python that the

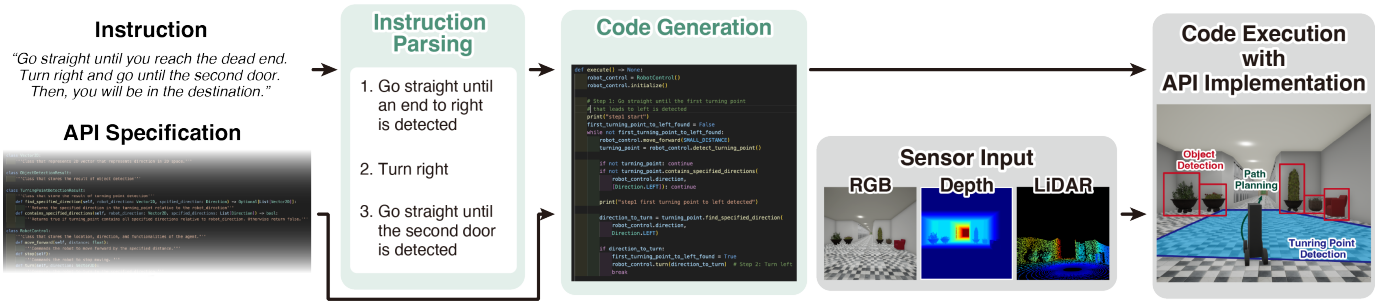


Fig. 4: **Method Overview.** Given a set of instructions from a sighted passerby, the LLM first parses it into an itemized format. Then, combined with the API specification, the LLM generates Python code directly to control the robot, which runs in the virtual environment using the simulated sensor inputs.

agent can use as described in Sec. III-B, *API implementation* which is the actual implementation of the API specification, and the *initial orientation* of the robot. We assume the initial orientation is predetermined, as the blind user can adjust it in place. We used the GPT-4 (gpt-4-1106-preview) model for the LLM. For the initial setup of the prompt, instructions from five participants in the online study were used as references to construct the prompt for the proposed systems. Fig. 4 shows the implementation overview.

A. Parsing Instruction

The system first parses a natural language instruction to step-by-step instructions using LLM. This was done to organize our benchmark's diverse natural language instruction and make it more interpretable before generating navigation codes. To achieve this, we prompt LLMs with a set of rules they should follow, such as the requirement to describe when and which turning point to turn, and which object the robot should detect, along with examples of possible input and expected output. After parsing, each navigation step is returned as a brief sentence. We employ a two-stage prompting method to guide LLM for more accurate outputs. We prompt LLM to provide a thought to guide the generation of the first output, then refine the output by incorporating a second thought, leading to the finalized output.

B. Navigation Code Generation

To generate the navigation code, we prompt LLM with an API specification that includes a range of commands for robot operations (*e.g.*, `move_forward(distance)` function). These commands are complete with docstrings of their usage explanation [18], [19] and instructions to generate Python codes that follow the provided specification. For `detect_from_RGB_image(object)`, our model uses an open vocabulary object detector internally (Sec. III-B) and flexibly determines which object to detect by generating an object argument. For example, for an input requesting the location of a red chair, the function would be invoked as: `detect_from_RGB_image("red chair")` by an LLM while generating a code. The API specification was formatted to the similar format of the previous work [18], [19], but with additional notes, such as how each function should be and not be used. We again employ the same two-stage prompting method. Finally, we execute the code using the API implementation.

VI. Experiment

Our benchmark simulates a scenario in which blind people ask sighted passersby to provide route guidance [4] from their memories. To evaluate how current models perform under this setting, we conducted an experiment.

A. State-of-the-Art Models

In our scenario, VLN agents are expected to demonstrate strong transferability, as blind users may navigate across diverse unseen locations by asking sighted passersby for directions. To evaluate this capability, we compare our model with two prior state-of-the-art methods that leverage foundation models and exhibit strong zero-shot performance: NavGPT [10] and NaVid [11].

NavGPT [10] demonstrates strong zero-shot transfer capability by leveraging an LLM, visual foundation model, and an object detector to iteratively select destinations within a navigation graph until the agent determines it has reached the goal. We used GPT-4o-mini for the LLM and for the visual foundation model, and the same Grounding DINO [21] for the object detector. As NavGPT requires a navigation graph to operate, we constructed navigation graphs over the environments following the R2R dataset [1].

NaVid [11], a state-of-the-art VLN model that demonstrates strong generalization to unseen environments, employs a visual foundation model and operates without a navigation graph, relying solely on camera input, similar to our model. We strictly controlled the agent by following NaVid's established pipeline. We initialized its weights of LLM (Vicuna-7B) using their open-sourced checkpoint [11].

B. Metrics

For metrics, we employ success rate (SR), oracle success rate (OSR), and shortest path distance (SPD) [1], [23], and coverage weighted by length score (CLS) [24], [25]. As CLS computes the similarity of the path on the graph, it requires a dense navigation graph to map the navigated trajectory onto. Thus, we divided passable corridors into 50 cm square grids to serve as nodes on a graph and mapped predicted and ground truth paths onto it to calculate this metric [25]. For routes where participants described an alternative path, we used the described route as the ground truth.

TABLE II: **Performance of VLN Models.** We compare our method with state-of-the-art VLN models that fulfill the requirements relevant to our scenario.

Method	Condition Parser	Route	Online Think-Out-Loud Instructions SR↑	OSR↑	SPD↓	CLS↑	Onsite Memory-Based Instructions SR↑	OSR↑	SPD↓	CLS↑
NavGPT		University R1	0.04	0.09	37.54	0.05	0.02	0.04	40.58	0.05
NaVid		University R1	0.00	0.00	35.73	0.03	0.00	0.00	36.67	0.02
Proposed		University R1	0.20	0.24	**17.01**	**0.56**	0.23	0.33	21.04	0.46
Proposed	✓	University R1	**0.30**	**0.35**	19.66	0.49	**0.30**	**0.38**	**18.32**	**0.54**
NavGPT		University R2	0.00	0.00	162.10	0.01	0.00	0.00	161.13	0.01
NaVid		University R2	0.00	0.00	149.79	0.00	0.00	0.00	151.47	0.00
Proposed		University R2	0.00	0.00	93.66	0.32	**0.03**	**0.03**	117.52	0.20
Proposed	✓	University R2	**0.04**	**0.04**	**81.59**	**0.38**	**0.03**	**0.03**	**98.13**	**0.29**
NavGPT		Museum 5F R1	0.00	0.00	50.76	0.00	0.00	0.00	51.30	0.00
NaVid		Museum 5F R1	0.00	0.00	54.59	0.01	0.00	0.00	55.50	0.01
Proposed		Museum 5F R1	0.11	0.20	35.46	0.44	0.02	0.02	43.35	0.32
Proposed	✓	Museum 5F R1	**0.20**	**0.26**	**26.71**	**0.60**	**0.05**	**0.07**	**29.14**	**0.54**
NavGPT		Museum 5F R2	0.00	0.07	37.47	0.07	0.00	0.07	35.67	0.06
NaVid		Museum 5F R2	0.00	0.00	43.74	0.01	0.00	0.00	43.82	0.01
Proposed		Museum 5F R2	**0.05**	0.18	23.17	0.25	0.00	**0.02**	29.08	**0.37**
Proposed	✓	Museum 5F R2	**0.05**	**0.32**	**16.43**	**0.29**	0.00	**0.02**	**24.78**	**0.37**
NavGPT		Museum 7F R1	0.00	0.00	54.70	0.08	0.00	0.00	36.00	0.14
NaVid		Museum 7F R1	0.00	0.00	73.76	0.06	0.00	0.00	71.30	0.07
Proposed		Museum 7F R1	**0.02**	**0.02**	55.99	0.31	0.05	0.05	46.67	0.42
Proposed	✓	Museum 7F R1	**0.02**	**0.02**	**42.59**	**0.48**	**0.09**	**0.09**	**25.22**	**0.67**
NavGPT		Museum 7F R2	0.00	0.00	61.64	0.01	0.00	0.00	60.43	0.01
NaVid		Museum 7F R2	0.00	0.00	67.39	0.02	0.00	0.00	69.18	0.00
Proposed		Museum 7F R2	**0.15**	0.26	47.41	0.17	0.00	0.10	52.81	0.16
Proposed	✓	Museum 7F R2	0.07	**0.35**	**36.78**	**0.25**	**0.02**	**0.12**	**46.74**	**0.22**

VII. Results and Discussion

A. Performance of the Proposed Method

As shown in Table II, our model outperforms NavGPT and NaVid. The baselines' suboptimal performances can be attributed to two factors: their deviation from the correct direction, and their premature decision that they had reached the goal. This is due to the fact that the baselines refer to the environment at each navigation step with an LLM. For example, if NavGPT makes a mistake even once during this process, it will be challenging for the model to recover the agent back to the correct path. Additionally, NaVid tends to make unnecessary frequent turns after initially following the route correctly for several iterations, likely due to the longer sequence of turns and longer instructions in our benchmark, which NaVid was not trained to handle. In contrast, our method achieves the desired outcome through a single iteration of code generation inference, removing the need to initiate inferences at every intermediate step for instructions like *"go straight for 100m and then turn right."*. We also observed that the instruction parsing module boosted the performance of our method in most cases.

B. Difficulty of the Benchmark

In Table II, it is observed that the performances from onsite memory-based instructions tended to be lower than those from online think-out-loud instructions, as it is more likely for the route instructions to contain errors due to human memory, and it is harder for the system to recover from errors. Overall, our results demonstrate the difficulty of the instruction data from human memory and the value of our benchmark.

Across all routes, both our model and the baselines showed suboptimal or low performance. One major reason was the difficulty in handling the varied and inaccurate input instructions. In longer routes, participants tended to inaccurately estimate lengths for certain path segments and not include sufficient information about the destination. Also, because our baseline contained modularized perception and control modules to focus on language parsing and reasoning capabilities, its suboptimal performance implies that the primary challenge in our benchmark lies in the complexity of the language instructions, such as inaccuracies or variations in wording, which were not present in previous benchmarks. Upon closer inspection, many instructions in our benchmark contained phrases that required a combined understanding of both natural language and the building's structure, which our proposed model failed to follow. One example was a phrase such as *"go along this path and turn right in the first intersection,"* which was often described at the starting point of University R1. The instruction skips the right turn in the first turning point by describing it as *"go along this path,"* because the building structure only allows a right turn at the immediate corner. As a result, the instruction starts by describing the first left turn where there are two possible directions to proceed. This variation in the levels of topological details further highlights the difficulty of our benchmark, which imitates real-world scenarios of blind people seeking navigation instructions.

TABLE III: **Effect of Instruction Refinement.** While in most cases refining instruction leads to an increase in performance, in certain cases, it was not always the case, due to redundant referral to surrounding objects.

Route	Condition Iteration	Online Think-Out-Loud Instructions SR↑	OSR↑	SPD↓	CLS↑	Onsite Memory-Based Instructions SR↑	OSR↑	SPD↓	CLS↑
University R1	1	**0.43**	**0.48**	**16.37**	**0.56**	0.25	**0.40**	20.41	0.52
University R1	2	0.17	0.22	22.94	0.41	**0.35**	0.35	**16.23**	**0.56**
University R2	1	**0.04**	**0.04**	86.82	0.36	0.00	0.00	**93.32**	**0.31**
University R2	2	**0.04**	**0.04**	**76.36**	**0.41**	**0.05**	**0.05**	102.95	0.28
Museum 5F R1	1	**0.26**	**0.30**	**25.80**	0.60	0.00	0.00	**27.21**	**0.56**
Museum 5F R1	2	0.13	0.22	27.61	**0.61**	**0.10**	**0.14**	31.07	0.52
Museum 5F R2	1	**0.05**	0.27	17.77	0.27	0.00	0.00	25.75	0.32
Museum 5F R2	2	**0.05**	**0.36**	**15.09**	**0.30**	0.00	**0.05**	**23.82**	**0.43**
Museum 7F R1	1	0.00	0.00	46.57	0.43	**0.09**	**0.09**	**23.77**	**0.68**
Museum 7F R1	2	**0.05**	**0.05**	**38.61**	**0.53**	**0.09**	**0.09**	26.66	0.65
Museum 7F R2	1	0.00	0.26	37.24	**0.26**	0.00	0.05	**46.71**	0.21
Museum 7F R2	2	**0.13**	**0.43**	**36.33**	0.24	**0.05**	**0.19**	46.76	**0.23**

Furthermore, we realized that in the real-world, humans may be able to correct errors in the instructions. For example, some passersby instructed the robot to take a nonexistent corridor between the hexagon and the rectangular exhibitions near the museum 5F R2. Although the corridor did not exist, imagining its intended destination allowed locating an alternative route. Similarly, when participants provided incorrect turns, landmarks described later in the instructions helped identify and correct these oversights.

C. Effect of Refining Instruction

Table III, reports the performance of our proposed method across different instruction iterations. The first iteration corresponds to the most natural, memory-based instruction, while the second represents memory-based instructions that are more accurate and contain features that may better assist the VLN agent in navigation. Generally, we found that refining the instruction led to a slight performance improvement. This suggests that, when deploying VLN-equipped robots, it is beneficial to assist sighted passersby in recalling routes more and in conveying environmental information in a format that is more compatible with robotic interpretation. However, for Museum 5F R1 and University R1, the tendency was not always the case. This happened because participants tended

to describe more objects during the second iteration, which contained greater variation in object descriptions. For example, one participant described only turning point-related information at the first iteration, while in the second iteration, the participant also described objects to ignore, such as, *"then, you will come across an intersection with a door on the left and an intersection with doors on both sides, but ignore them and continue straight ahead."*

VIII. Conclusion and Future Work

This work proposed Memory-Maze. We found that realistic instructions collected in the onsite environment, where participants had to rely on human memories, were longer with greater variation in words, and contained more errors compared to the instructions collected online. Upon qualitative inspection, we observed evidence of the tendency for memory-based instruction to be more difficult for the model to handle, such as the ones that required understanding of *"go along this path."* This suggests that future VLN models should consider a more adaptive map representation where nodes and turns are not strictly defined, or a more flexible approach to accommodate varying topological descriptions.

For future work, we aim to explore the interactive aspect with users and robots. For example, the robot could also guide the instruction from passersby to be better or rephrase it itself, potentially leading to better performance. We also plan to convert our baseline into a closed-loop architecture that continuously verifies and refines its interpretation of instructions using real-time sensor input.

Lastly, although the size of our benchmark is comparable to existing benchmarks [26], [27], its size remains limited in order to be used as a dataset for training. One possible approach is to modify the design of the online study so that annotators can only observe the environment prior to providing annotations, but this would only partially replicate the characteristics of real-world memory-based data. Another potential method is to leverage LLMs with in-context learning to augment the benchmark. However, further investigation is needed to ensure that LLM-generated data can accurately mimic the characteristics of our dataset.

ACKNOWLEDGEMENT

We deeply thank Hironobu Takagi for engaging in the discussion of this project. We also thank participants in our experiment, from Carnegie Mellon University, Waseda University and Miraikan.

References

[1] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. Van Den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *CVPR*, 2018.

[2] J. Guerreiro, D. Sato, S. Asakawa, H. Dong, K. M. Kitani, and C. Asakawa, "Cabot: Designing and evaluating an autonomous navigation robot for blind people," in *ASSETS*, 2019.

[3] S. Liu, A. Hasan, K. Hong, R. Wang, P. Chang, Z. Mizrachi, J. Lin, D. L. McPherson, W. A. Rogers, and K. Driggs-Campbell, "Dragon: A dialogue-based robot for assistive navigation with visual language grounding," *RA-L*, 2024.

[4] M. Kuribayashi, T. Ishihara, D. Sato, J. Vongkulbhisal, K. Ram, S. Kayukawa, H. Takagi, S. Morishima, and C. Asakawa, "Pathfinder: Designing a map-less navigation system for blind people in unfamiliar buildings," in *CHI*, 2023.

[5] K. Müller, C. Engel, C. Loitsch, R. Stiefelhagen, and G. Weber, "Traveling more independently: A study on the diverse needs and challenges of people with visual or mobility impairments in unfamiliar indoor environments," *TACCESS*, 2022.

[6] M. Kuribayashi, K. Uehara, A. Wang, S. Morishima, and C. Asakawa, "Wanderguide: Indoor map-less robotic guide for exploration by blind people," in *CHI*, 2025.

[7] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *CoRL*, 2017.

[8] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi, "Touchdown: Natural language navigation and spatial reasoning in visual street environments," in *CVPR*, 2019.

[9] W. Wu, T. Chang, X. Li, Q. Yin, and Y. Hu, "Vision-language navigation: A survey and taxonomy," *NCAA*, 2023.

[10] G. Zhou, Y. Hong, and Q. Wu, "Navgpt: Explicit reasoning in vision-and-language navigation with large language models," in *AAAI*, 2024.

[11] J. Zhang, K. Wang, R. Xu, G. Zhou, Y. Hong, X. Fang, Q. Wu, Z. Zhang, and H. Wang, "Navid: Video-based vlm plans the next step for vision-and-language navigation," *RSS*, 2024.

[12] V. Ranganeni, M. Sinclair, E. Ofek, A. Miller, J. Campbell, A. Kolobov, and E. Cutrell, "Exploring levels of control for a navigation assistant for blind travelers," in *HRI'23*, 2023.

[13] Y. Qi, Q. Wu, P. Anderson, X. Wang, W. Y. Wang, C. Shen, and A. v. d. Hengel, "Reverie: Remote embodied visual referring expression in real indoor environments," in *CVPR*, 2020.

[14] X. Song, W. Chen, Y. Liu, W. Chen, G. Li, and L. Lin, "Towards long-horizon vision-language navigation: Platform, benchmark and method," in *CVPR*, 2025.

[15] D. Shah, B. Osiński, S. Levine, *et al.*, "Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *CoRL*, 2023.

[16] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *ICML*, 2022.

[17] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *ICRA*, 2023.

[18] H. Biggie, A. N. Mopidevi, D. Woods, and C. Heckman, "Tell me where to go: A composable framework for context-aware embodied robot navigation," in *CoRL*, 2023.

[19] D. Surís, S. Menon, and C. Vondrick, "Vipergpt: Visual inference via python execution for reasoning," in *ICCV*, 2023.

[20] J. Krantz, A. Gokaslan, D. Batra, S. Lee, and O. Maksymets, "Waypoint models for instruction-guided navigation in continuous environments," in *CVPR*, 2021.

[21] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," in *ECCV*, 2024.

[22] A. Ku, P. Anderson, R. Patel, E. Ie, and J. Baldridge, "Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding," in *EMNLP*, 2020.

[23] J. Gu, E. Stefani, Q. Wu, J. Thomason, and X. Wang, "Vision-and-language navigation: A survey of tasks, methods, and future directions," in *ACL*, 2022.

[24] V. Jain, G. Magalhaes, A. Ku, A. Vaswani, E. Ie, and J. Baldridge, "Stay on the path: Instruction fidelity in vision-and-language navigation," in *ACL*, 2019.

[25] G. Ilharco, V. Jain, A. Ku, E. Ie, and J. Baldridge, "General evaluation for instruction conditioned navigation using dynamic time warping," *arXiv*, 2019.

[26] C. Fu, Y. Dai, Y. Luo, L. Li, S. Ren, R. Zhang, Z. Wang, C. Zhou, Y. Shen, M. Zhang, *et al.*, "Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis," in *CVPR*, 2025.

[27] J. Krantz, S. Banerjee, W. Zhu, J. Corso, P. Anderson, S. Lee, and J. Thomason, "Iterative vision-and-language navigation," in *CVPR*, 2023.