Think, Act, and Ask: Open-World Interactive Personalized Robot Navigation

Yinpei Dai[1], Run Peng[1], Sikai Li[1], Joyce Chai[1]

***Abstract*— Zero-Shot Object Navigation (ZSON) enables agents to navigate towards open-vocabulary objects in unknown environments. The existing works of ZSON mainly focus on following individual instructions to find generic object classes, neglecting the utilization of natural language interaction and the complexities of identifying user-specific objects. To address these limitations, we introduce Zero-shot Interactive Personalized Object Navigation (ZIPON), where robots need to navigate to personalized goal objects while engaging in conversations with users. To solve ZIPON, we propose a new framework termed Open-woRld Interactive persOnalized Navigation (ORION), which uses Large Language Models (LLMs) to make sequential decisions to manipulate different modules for perception, navigation and communication. Experimental results show that the performance of interactive agents that can leverage user feedback exhibits significant improvement. However, obtaining a good balance between task completion and the efficiency of navigation and interaction remains challenging for all methods. We further provide more findings on the impact of diverse user feedback forms on the agents' performance.**

I. INTRODUCTION

Recent years have seen an increasing amount of work on Zero-Shot Object Navigation (ZSON) where an embodied agent is tasked to navigate to open-vocabulary goal objects in an unseen environment [1], [2]. This task is crucial for developing robots that can work seamlessly alongside end-users and execute open-world daily tasks through natural language communication. A common practice in ZSON is to utilize pre-trained vision-language models (VLMs) out-of-the-box to ground natural language to visual observations, thus enabling the robots to handle unseen object goals without additional training [3], [4], [5], [6]. Despite recent advances, several limitations remain which hinder the real-world deployment of these agents.

First, previous works only focus on following individual instructions without considering feedback and interaction. In the realistic setting, instruction-following often involves back-and-forth interaction to reduce uncertainties, correct mistakes, and handle exceptions. For example, given an instruction *"go to the living room and find the toy airplane"*, if the robot goes to a wrong room, immediate feedback from the user will save the robot's endless search in the wrong room and direct it to the desired location. Therefore, it is important to build agents that can elicit and leverage language feedback from users during task execution to avoid errors and achieve the goal. Moreover, current navigation tasks are often designed to find any instance of the same object class [1], [7], [8]. However, in the real world, especially in a household, or office setting, objects can often be described by unique and personalized properties that are shared by people with the same background knowledge. For example, as shown in Figure 1, the robot is instructed to find *"Alice's computer"* (or *"the computer purchased last year"*). Just like humans, the robot needs to acquire and apply such personalized knowledge in communication with users. Even though current models can correctly identify a 'computer' based on their general perception abilities, it remains unclear how to build an agent that can swiftly adapt to a personal environment and meet users' personalized needs.

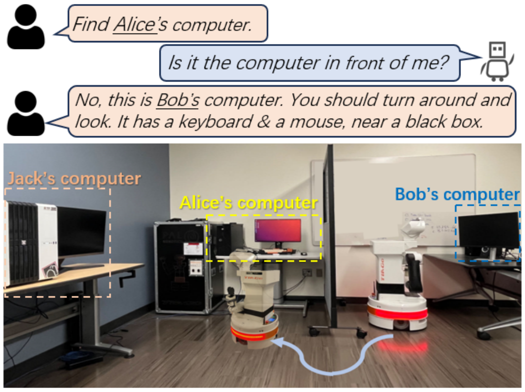

Find Alice's computer.
Is it the computer in front of me?
No, this is Bob's computer. You should turn around and look. It has a keyboard & a mouse, near a black box.
Jack's computer
Alice's computer
Bob's computer

Fig. 1: An example of zero-shot interactive personalized navigation. There are three computers in the room never seen by the robot before. The goal is to find *Alice's computer*. The robot starts by finding the wrong object and needs to communicate with the user and leverage the user feedback to locate the personalized goal.

To this end, we introduce *Zero-shot Interactive Personalized Object Navigation* (ZIPON), an extended version of ZSON. In ZIPON, the robot needs to navigate to a sequence of *personalized goal objects* (objects described by personal information) in an unseen scene. As shown in Figure 1, the robot can engage in conversations with users and leverage user feedback to identify the object of interest. Different from previous zero-shot navigation tasks [1], [4], [6], our task evaluates agents on two fronts: (i) language interactivity (understanding when and how to converse with users for feedback) and (ii) personalization awareness (distinguishing objects with personalized attributes such as names or product details). To the best of our knowledge, this is the first work to study personalized robot navigation in this new setting.

To enable ZIPON, we develop a general LLM-based framework for Open-woRld Interactive persOnalized Navigation (ORION). Specifically, ORION consists of vari-

[1] Computer Science and Engineering Division, University of Michigan. Emails: {daiyp, chaijy}@umich.edu.

This work is supported by NSF IIS-1949634, NSF SES-2128623, and Amazon.

ous modules for perception, navigation and communication. The LLM functions as a sequential decision maker to control the modules in a *think-act-ask* manner: In the *think* step, the LLM reflects the navigation history and reason about the next plans; In the *act* step, the LLM predicts an action to execute a module and the executed message is returned as context input for the next action prediction; In the *ask* step, the LLM generates natural language responses to interact with the user for more information. This framework allows us to incorporate different baselines to conduct extensive studies on ZIPON. Our empirical results have shown that agents that can communicate and leverage diverse user feedback significantly improve their success rates. We also find that the feedback providing hints of nearby landmark objects and step-by-step low-level navigation guidance improves the agent's performance most, while the feedback that contains enriched object descriptions and corrections brings relatively limited enhancement. However, all agents still struggle to balance task success with navigation and interaction efficiency. We hope these findings can facilitate future work on personalized navigation in the open world.

II. Related Work

Zero-Shot Object Navigation. Recently, there has been a growing interest in zero-shot object navigation using VLMs. One line of the approaches is exploration-based, where the agent moves around based on standard exploration algorithms and matches the ego-observations with language descriptions via VLMs [1], [3], [9]. Another line is the map-based method, where a spatial semantic map is built with VLM representations to enable natural language indexing [6], [10], [11]. Our framework integrates both methods for more flexibility and superior navigation performance.

LLMs as Dialogue Agents for Robots. LLMs have been extensively applied to enable human-robot conversations. InnerMonologue [12] used an LLM to form an inner monologue style to allow robots to ask questions. PromptCraft [13] explored prompt engineering practices for robot dialogues with ChatGPT. KNOWNO[14] measured the uncertainty of LLM planners for agents to ask for help when needed. In contrast, we use LLMs to operate different robot modules and frame it as a sequential decision-making problem.

Leverage Language Feedback. Autonomous agents that know how to adapt to language feedback can make more reliable decisions. Some research emphasizes real-time robot plan adjustments [15], [16]. Others harness language hints for assistance, such as CVDN [17], Pasture [4], and HARI [18]. However, no study has comprehensively compared different feedback types in the same navigation context.

Personalized Human-Robot Interaction. Building personalized robots is an active research area in human-robot interaction [19], [20], [21]. There are many works focused on enhancing personalized experience through non-verbal communications [22], [23], [24] and better interactive service design [25]. Personalized dialogue systems also gained increasing interest [26], [27], [28], where the persona is taken as conditional input to produce more characterized and sociable conversations. To our knowledge, previous work has not investigated interactive personalized navigation tasks.

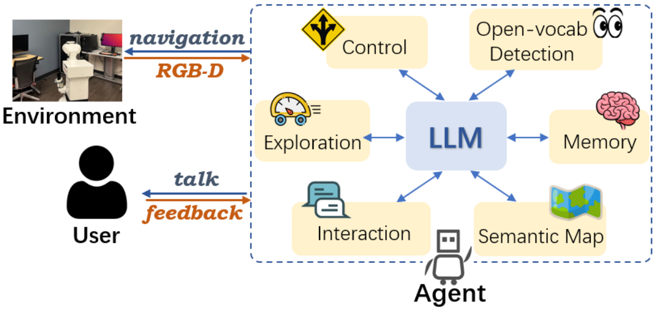


Fig. 2: The ORION framework architecture. The LLM makes sequential decisions to operate different modules to search, detect and navigate in the environment and talk with the user.

III. Interactive Personalized Navigation

We first introduce the zero-shot interactive personalized object Navigation (ZIPON) task, then explain the open-world interactive personalized navigation (ORION) framework.

A. The ZIPON Task

ZIPON is a generalized type of zero-shot object navigation [4]. Let $\mathcal{E}$ denote the set of all test scenes, and $\mathcal{G}$ denote the set of all personalized goals. Each goal $g \in G$ contains a tuple $g = (type, name, room, FB)$, where *type* is the class label for the object (e.g., 'bed', 'chair'), *name* is the personalized expression (e.g. 'Alice's bed', 'chair bought from Amazon') which is unique for every goal, *room* is the name of the room (e.g., 'Alice bedroom', 'living room') where g is located in, and *FB* is a dictionary to store all types of user feedback information (See Sec. IV-A). A navigation episode $\tau \in \mathcal{T}$ is a tuple $\tau = (e, g, p_0)$, where $e \in \mathcal{E}$, $g \in \mathcal{G}$, p_0 is the initial pose of the agent for current τ. The input observations are RGB-D images. Starting from p_0, the agent needs to find g by taking a sequence of primitive actions, where the action space is $\mathcal{A}_{task}$={TurnLeft 15°, TurnRight 15°, MoveForward 0.25m, Talk}. The first three are navigation actions, and the last one is a communicative action with the dialogue content to be generated by the agent. During the evaluation for τ, the agent can either move in the environment with navigation actions or interact with the user with Talk for more information. Whenever the agent believes it has reached the goal, it must issue Talk to stop moving and confirm with the user. The τ is terminated when the robot successfully finds the g or a maximum number $I_{\max}$ of interaction attempts (i.e., the total number of Talk actions) is attained. If the agent is within c meters of g and meets visibility criteria, the τ is successful.

B. Proposed ORION Framework

We propose ORION to solve ZIPON. As shown in Fig. 2, this framework comprises six modules: a control module to perform navigation movements, a semantic map module for natural language indexing, an open-vocabulary detection module to detect any objects with language descriptions, an exploration module to search the room, a memory module to store crucial information from user feedback, and an interaction module to talk with users. Central to these, the Large Language Model (LLM) serves as the primary controller, making sequential actions based on its strong reasoning abilities.

1) LLM Action Space: To make ORION more general, we design a unified action space for LLM to manipulate all modules. An *LLM action* is defined as a string indicating a python function, e.g., `FuncName(param1=value1, ...)`. The total LLM action space, denoted as $\mathcal{A}_{LLM}$, is a collection of all executable functions for LLM, extending beyond $\mathcal{A}_{task}$ to encompass more high-level abilities. On receiving the user utterance, the LLM will generate the next action to execute a specific module and take the returned function message as new input to predict the next action until the Talk action is chosen; then the agent will communicate with the user, asking for confirmation or more information. To best utilize the powerful reasoning abilities of LLMs, we propose a *think-act-ask* mechanism to prompt the LLM to generate actions. Specifically, given context, the LLM yields a JSON-format string like {“Thought”:$\cdots$, “Action”:$\cdots$} for the next *action*. This string will be parsed, and the “Action” part is used to operate modules for navigation or interaction. Below is an example of the LLM context, where text color differentiates input (black) and output (red).

```
... # definition of the task
... # definition of all LLM actions

1 Here is an example to perform sequential actions:
2 User Utterance: Find Alice’s computer.
3 {"Thought":"The user wants Alice’s computer, I should
     first search the memory to see if I found it before",
4  "Action":{"name":"retrieve_memory",
            "args":{"obj_str":"Alice’s computer"}}}
5 Function Return: Found 0 items in memory: []
... # internal process of the LLM
6 {"Thought":"I found a possible computer in the room, it
    might be the correct one, I shall ask the user",
7  "Action":{"name":"talk",
          "args":{"content":"Is this Alice’s computer?"}}}
8 Robot Response: Is this Alice’s computer?
9 User Utterance: No, it’s Bob’s computer. Keep searching
...
```

The initial two lines expound the ZIPON task and LLM actions (omitted for brevity). Line 1 begins the illustration of *think-act-ask* process. Line 2 is the first user input. Lines 3-4 show the LLM’s subsequent thought process and resultant action. Line 5 is the executed message from the last action. The line following line 5, omitted for space, signifies the internal procedure for generating sequential actions before user interaction, usually with multiple rounds. Line 6-7 shows the LLM action output for communication when the LLM thinks it’s time to talk. The robot will then interact with the user and continue to take action based on the new user input (line 9). Following this design, the LLM can manipulate all modules seamlessly after the user issues a goal and decide when and what to talk with the user.

2) Operated Modules: In this section, we delve into the utilities and LLM actions associated with each module.

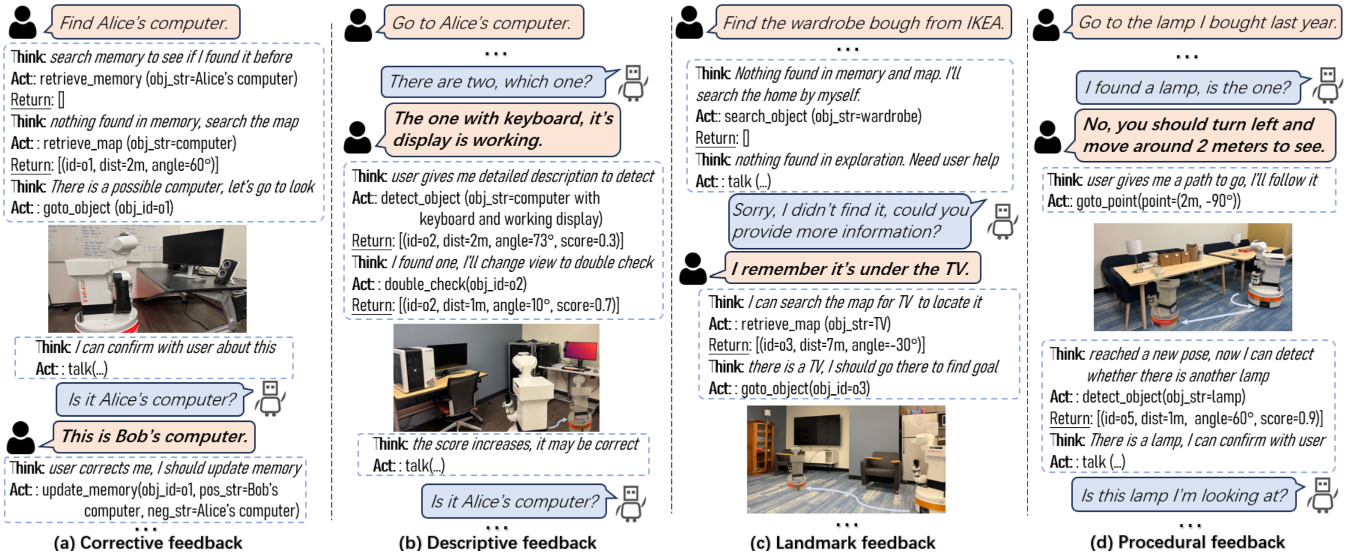
Control Module. It contains two low-level navigation actions, `Move(num)` and `Rotate(num)`, that cover the primitive navigation actions in $\mathcal{A}_{task}$. Concretely, `Move(1)` is MoveForward 0.25m, `Rotate(±1)` denotes TurnRight 15° and TurnLeft 15°, respectively. In addition, it has two high-level actions: `goto_point(point)`, that sequences low-level navigation actions to navigate to a point on a 2D occupancy map, and `goto_object(obj_id)` that directly navigates to an object with the given id.

Semantic Map. Following [6], we collect RGB-D images to build vision-language maps by reconstructing the point cloud and fusing the LSeg [29] features from each point to yield a top-down neural map denoted as $M_{sem} \in \mathbb{R}^{L \times W \times C}$. Here, L and W are the map length and width, respectively. C is the CLIP [30] feature dimension. The LLM action for this module, `retrieve_map(obj_str)`, utilizes the CLIP text encoder to obtain the embedding $t \in \mathbb{R}^{C}$ for an object description string. It then identifies a similarity matching area $A_{sem} \in \mathbb{R}^{L \times W}$ in the map and extracts suitable contours. These contours are returned as a list of tuples (obj_id, distance, angle), indicating the assigned ID, distance and angle of the contour’s center relative to the agent’s pose.

Open-vocabulary Detection. This module is to detect any object from an RGB image using its language description. We use the grounded-SAM [31] due to its good performance, but any other detection models can be used here. Once detected, the segmented pixels are transformed into 3D space and projected to the 2D occupancy map to acquire the detected area $A_{det} \in \mathbb{R}^{L \times W}$. The module has two actions: (i) `detect_object(obj_str)` that returns a list of detected objects written as (obj_id, distance, angle, detection_score) tuple, and (ii) `double_check(obj_id)` which repositions to a closer viewpoint of an object and detect again.

Exploration Module. This module uses frontier-based exploration (FBE) [32] to help the agent search unseen scenes. Following [3], the agent starts with a 360° spin and builds a 2D occupancy grid map for exploration. The LLM action for this module is `search_object(obj_str)`, where FBE determines the next frontier points for the control module to take `goto_point` and the detection module to perceive on-the-fly with `detect_object`. Upon detecting objects, this module returns the detection message. If invoked again, it continues to explore the room until FBE stops.

Memory. This module stores crucial user information with two neural maps, M_{pos} and M_{neg}, both of which have the same size as M_{sem} and hold CLIP features. M_{pos} saves user-affirmed information, while M_{neg} records user denials. Both maps begin with zero initialization and are updated through the action `update_memory(obj_id, pos_str,neg_str)`. For example, if the user confirms *“yes, it’s Alice’s desk”*, then the LLM sets *“Alice’s desk”* as the positive string and adds its CLIP text feature into the object area associated with the given object id in M_{pos}. Conversely, a denial like *“no, it’s not cabinet”* leads to the CLIP feature of *”cabinet”* being stored in M_{neg} as the negative string. Object areas derive from either A_{sem} or A_{det}. Another action, `retrieve_memory(obj_str)`, matches the CLIP text embedding of the input object description string with features in two maps, then returns retrieved objects as (obj_id, distance, angle) tuples from M_{pos} while avoiding those in M_{neg}.



Find Alice's computer.

Think: *search memory to see if I found it before*
Act:: retrieve_memory (obj_str=Alice's computer)
Return: []
Think: *nothing found in memory, search the map*
Act: : retrieve_map (obj_str=computer)
Return: [(id=o1, dist=2m, angle=60°)]
Think: *There is a possible computer, let's go to look*
Act: : goto_object (obj_id=o1)

Think: *I can confirm with user about this*
Act: : talk(...)

Is it Alice's computer?

This is Bob's computer.

Think: *user corrects me, I should update memory*
Act: : update_memory(obj_id=o1, pos_str=Bob's computer, neg_str=Alice's computer)

...

(a) Corrective feedback

Go to Alice's computer.

...

There are two, which one?

The one with keyboard, it's display is working.

Think: *user gives me detailed description to detect*
Act:: detect_object (obj_str=computer with keyboard and working display)
Return: [(id=o2, dist=2m, angle=73°, score=0.3)]
Think: *I found one, I'll change view to double check*
Act: : double_check(obj_id=o2)
Return: [(id=o2, dist=1m, angle=10°, score=0.7)]

Think: *the score increases, it may be correct*
Act: : talk(...)

Is it Alice's computer?

...

(b) Descriptive feedback

Find the wardrobe bough from IKEA.

...

Think: *Nothing found in memory and map. I'll search the home by myself.*
Act:: search_object (obj_str=wardrobe)
Return: []
Think: *nothing found in exploration. Need user help*
Act: : talk (...)

Sorry, I didn't find it, could you provide more information?

I remember it's under the TV.

Think: *I can search the map for TV to locate it*
Act: : retrieve_map (obj_str=TV)
Return: [(id=o3, dist=7m, angle=-30°)]
Think: *there is a TV, I should go there to find goal*
Act: : goto_object(obj_id=o3)

...

(c) Landmark feedback

Go to the lamp I bought last year.

...

I found a lamp, is the one?

No, you should turn left and move around 2 meters to see.

Think: *user gives me a path to go, I'll follow it*
Act:: goto_point(point=(2m, -90°))

Think: *reached a new pose, now I can detect whether there is another lamp*
Act: : detect_object(obj_str=lamp)
Return: [(id=o5, dist=1m, angle=60°, score=0.9)]
Think: *There is a lamp, I can confirm with user*
Act: : talk (...)

Is this lamp I'm looking at?

...

(d) Procedural feedback

Fig. 3: The dialogue examples for different language feedback types. Corrective feedback indicates errors in the robot's object identification. Descriptive feedback details the goal's appearance and status. Landmark feedback points out salient nearby objects. Procedural feedback suggests a rough navigation path to the goal. The content within the dashed box represents the internal *think-act-ask* processes the LLM undergoes. The content in the coloured boxes denotes the interactions between the user and the robot. Ellipsis marks indicate the omission of intermediate human/robot dialogue contents and LLM internal process for space brevity.

Interaction. This module is for the communicative interface between the robot and the user. This work uses a textual interface where users interact with the robot by typing texts. The LLM action is `talk(content)`, which is the same TALK in $\mathcal{A}_{task}$ with the content to be generated by the LLM.

IV. EXPERIMENTS

In this section, we begin by introducing the experimental design for ZIPON in Sec. IV-A. Then, we elaborate on the experiments conducted in simulated environments in Sec. IV-B and demonstrate our method on real robots in Sec. IV-C.

A. Experimental Design

Natural language interaction is the core feature of our ZIPON task. When the robot makes mistakes or asks for help, the user can guide the robot to find the goal through diverse forms of language feedback. Therefore, we conduct experiments to examine the influence of different types of user feedback on ORION and baseline alternatives.

User Feedback Types. Four common types[1] of user language feedback are used in our experiments: 1) *corrective feedback*, which tells the robot what object it has actually reached if that's not the correct goal; 2) *descriptive feedback*, which provides more details about the goal object's appearance, status, functions, etc; 3) *landmark feedback*, which gives the object landmarks like other common and salient objects near the current goal object; and 4) *procedural feedback*, which offers language-described approximate routes to instruct the robot to approach the goal from the current pose step by step. We hope through controlling the form of user feedback during the interaction, some interesting findings can be generated. Fig. 3 illustrates examples that the ORION takes actions under different user feedback settings.

Baselines. Three strong zero-shot object navigation baselines are adapted to our flexible ORION framework.

- CLIP-on-Wheels (CoW) [3] uses FBE to explore and localize objects in images with CLIP saliency maps using Grad-CAM[33]. We apply these to the exploration module and open-vocabulary detection module. However, CoW does not have a semantic map and memory module originally.
- VLMap [6] fuses LSeg features for each image and 3D reconstructs spatial maps for natural language indexing. We apply it to the semantic map module, but VLMap does not contain detection, memory and exploration modules.
- ConceptFusion (CF) [10] adopts a similar map creation technique but maintains original CLIP features to capture uncommon objects with higher recall than VLmap. We utilize the CF map in the semantic map module, while other modules remain the same as VLMap.

All compared methods are connected with the same LLM (GPT-4-8k-0613) to schedule different modules to navigate and interact with users.

Evaluation Metrics. A good interactive navigation agent should be efficient (i.e., achieve the goal as fast as possible) and pleasant (i.e., bother the user as little as possible). Therefore, we use (i) Success Rate (SR), defined as SR $= \frac{1}{N}\sum_{i=1}^{N} S_i$ to evaluate the percentage that the agent reaches the correct goals, $S_i \in \{0, 1\}$ is the binary indicator of success for g_i, N is the total number of goals; (ii) Success Rate weighted by the Path Length (SPL) [7], defined as SPL $= \frac{1}{N}\sum_{i=1}^{N} S_i \frac{l_i}{max(a_i, l_i)}$, where the l_i denotes the ground truth shortest path length, and a_i denotes the actual path length; (iii) Success Rate weighted by the Interaction Turns (SIT), defined as SIT $= \frac{1}{N}\sum_{i=1}^{N} S_i \frac{1}{I_i}$, where $I_i \geq 1$ is the number of interactions between the agent and user for g_i. SPL and SIT indicate the navigation and interaction efficiency respectively.

[1] We do not consider emotional or social aspects in this work.

Method	No Interaction			Yes/no Feedback			Corrective Feedback			Descriptive Feedback			Landmark Feedback			Procedure Feedback		
	SR	SPL	SIT	SR	SPL	SIT	SR	SPL	SIT	SR	SPL	SIT	SR	SPL	SIT	SR	SPL	SIT
Human	–	–	–	–	–	–	94.5	75.7	81.9	94.5	75.4	84.8	95.0	76.6	86.0	97.2	78.9	86.9
COW	15.4	8.4	15.4	36.8	24.2	35.8	38.5	21.6	29.0	53.5	22.4	33.2	43.9	21.3	30.2	59.0	22.5	32.8
VLmap	23.9	20.3	23.9	41.8	30.8	31.2	43.6	32.6	35.2	44.4	31.5	35.8	53.3	35.8	39.0	67.5	**52.7**	44.1
CF	21.3	13.7	21.3	47.9	25.6	31.1	52.1	**37.5**	36.5	47.0	33.5	34.1	59.9	**40.1**	40.6	68.4	49.4	39.3
ORION	**28.2**	**24.9**	**28.2**	**54.2***	**35.5**	**37.8**	**59.0***	34.2	**39.1**	**63.7***	**37.4**	**44.2***	**69.5***	**40.1**	**46.8**	**80.3***	51.1	**52.8***

TABLE I: Results of different methods in single-feedback settings with simulated users. The asterisk * indicates a statistically significant improvement of ORION over all three baselines for each column (wilcoxon test; $p < 0.05$).

Method	SR	SPL	SIT
COW	62.4	33.7	36.2
VLMap	71.8	54.8	48.0
CF	73.5	52.3	44.2
ORION	**83.8***	**56.6**	**53.5**
w/o mem	81.2	54.8	51.2
w/o exp	75.9	51.6	49.7
w/o det	72.5	47.7	42.2
w/o map	69.2	37.5	41.3

TABLE II: Results of different methods (upper part) and ablations (lower part) in the mixed-feedback setting with simulated users. * means statistical significance as in Tab. I

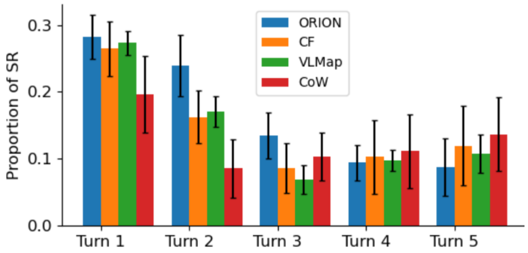


Fig. 4: Distribution of task success rates based on interaction turns (turn 1-5) in the mixed feedback setting for all compared methods.

B. *Evaluation with Simulated Environment*

In the simulated environment, we build simulated users to scale up the experiments and compare different methods under various feedback settings.

1) Experimental Setup: We use the Habitat simulator [34] and the high-quality realistic indoor scene dataset HM3D v0.2 [35] for experiments. Ten scenes are randomly selected from the validation set. A total of 14,159 RGB-D frames are collected for semantic map creation, and 117 goal objects are randomly selected for evaluation. To construct the personalized goals, we annotate the chosen objects with various types of personal information, such as people's names (e.g., *Alice's computer*), manufacturers of products (e.g., *chair from IKEA*) and purchase dates (e.g., *bed bought last year*), so that each g in $\mathcal{G}$ can be uniquely identified. We manually write object descriptions for each g for the descriptive feedback. For the procedural feedback, we generate the ground-truth geodesic path and translate it into language sentences to indicate an approximate route, e.g., *"turn left, go forward 5 meters, turn right"*. The fast-marching method [36] is used for low-level motion planning. The map size LxW is 600x600, where each grid equals 0.05m in the environment. The RGB-D image frame is 480x640 with a camera fov 90 degrees. The depth range is set in [0.1m,10m] for map processing. CLIP-ViT-B-32 is used for text feature extraction and semantic matching for the semantic map and memory module. An episode is successful when the agent is within 1.5 meters and the mass centre of g is in the ego-view image. $I_{\max}$ is 5 for each goal.

2) User Simulator: As real user interactions can be tedious and time-consuming, simulated users are often used as a substitute to evaluate dialogue agents [37], [38], [39]. We use another LLM (dubbed as user-LLM) as the backbone to build a user simulator to interact with robots. At each turn, all relevant ground-truth information is sent as input to the user-LLM in a dictionary, which includes the dialogue context, personalized goal information, task success signals, robot detection results, etc. Then, the user-LLM generates suitable user utterances for the robot, guided by appropriate prompt examples. We chose GPT-3.5-turbo-0613 since we found it adequate to produce reasonable user utterances. Each scene is run 5 times with random seeds to get average results.

3) Evaluation Results: Tab. I shows the results of all methods under single-feedback settings, where we control the input with only one type of feedback to be sent to the user-LLM to generate the user utterance. For comparison, we add 'No Interaction' setting, which means the robot can not interact with users, and 'Yes/no Feedback' setting, which means the user only gives yes/no response as minimal information to indicate whether the agent has reached the goal without any extra feedback information. We also include the results of human-teleoperated agents as the upper bound.

Comparison on different types of feedback. As shown in Tab. I, using each type of feedback contributes to a performance increase to some degree, highlighting the significance of natural language interactions. Specifically, the procedure feedback brings the most significant improvement, with an increase of 21-26% SR across all methods compared to the Yes/no Feedback. We found this because it can suggest a route or a destination point even though it is not precise. This guidance helps agents transition to nearby areas, substantially narrowing the scope needed to locate the goal. The landmark feedback also largely boosts the performance since the provided nearby objects could be easier to find than the goal objects, thus helping the robot reach the goal's neighbourhood. Comparatively, the descriptive feedback does not yield much benefit but still boosts methods that have the exploration module, resulting in a 16.7% and 10.5% increase in SR for CoW and ORION, respectively. We conjecture this is because those methods use open-vocabulary detection models to process online ego-view RGB images, so the fine-grained visual semantics of the objects can be maintained to align with rich language descriptions, whereas map-based methods like VLMap may lose these nuances during map creation. The corrective feedback brings the least improvement since it only improves when the wrong objects the agent found for the current goal could happen to be some correct objects of other goals in the same scene.

Besides the single-feedback settings we also evaluated in the mixed-feedback setting, where the user-LLM receives all types of feedback information to generate utterances. As observed in the upper section of Tab. II, all methods can be future improved from single-feedback settings, showing approximately 3-5% in both SR and SIT. This indicates that the richer the information a robot can access during interaction, the better its performance in the environment.

Comparison on different methods From Tabs I and II, it's evident that no single method consistently outperforms across all metrics in ZIPON tasks. Key challenges include: (i) *Balancing task success with navigation efficiency*. While ORION can achieve a high SR, it often lags in SPL, e.g., in Tab. I, it surpasses VLMap in SR by over 10% but is only 2% ahead in SPL. This is because it can explore and actively move in the room to detect, often involving more steps to find the goal. But map-based methods take direct movements to retrieved objects, regardless of their accuracy. (ii) *Balancing task success with interaction efficiency*. Compared with human tele-operated agents, all methods suffer from low SIT performance, indicating a large dependence on interaction rather than finding the goal objects more efficiently. Fig 4 further illustrates this, showing that a considerable proportion of SR requires large interaction turns for all methods.

Ablations and more analysis. Tab. II's lower section presents ablation results for ORION in the mixed feedback setting. Here, 'w/o mem' excludes the additional memory feature; 'w/o exp' omits the exploration module but retains the 360° spin; 'w/o det' replaces the grounded-SAM model with a basic k-patch mechanism [3] where the CLIP matches texts to image patches; and 'w/o map' removes the semantic map module. The results emphasize the crucial role of the semantic map, marked by the most significant SR drop in ORION w/o map. Both ORION w/o exp and ORION w/o det yield comparable outcomes, suggesting the importance of a robust detection model paired with active exploration. Interestingly, the memory module has a limited impact, even with its capability to retain user information. We hypothesize this is because, during the zero-shot evaluation, previously stored goals aren't retested. To further explore this, we conduct a second time ZIPON evaluation using the memory accumulated from the first time. Consequently, ORION gets scores of 91.5% in SR, 63.9% in SPL, and 65.3% in SIT.

C. Evaluation with Real Robots

We also perform real-world experiments with the TIAGo robot for the indoor ZIPON using the ORION framework.

1) Experimental Setup: We select 20 goal objects in a room for navigation. Each object is assigned a unique person's name unless it's paired with another object. For instance, Alice's computer would be on Alice's table. We then manually provide 3-5 sentences for each goal as the descriptive feedback. Nine salient objects (e.g., fridge) in the room are used as landmarks. Simple instructions like *"3 meters to your left"* are used for the procedural feedback. The built-in GMapping [40] and move_base package are used for SLAM and path planning. To create the semantic map, we construct a topology graph that includes the generic classes of all landmark objects and large goal objects for simplicity. Experiments are run twice to get average results. Other setups remain the same with simulated experiments.

Feedback Type	SR	SPL	SIT
No Interaction	37.5	35.8	37.5
Corrective	67.5*	51.8*	53.6*
Descriptive	67.5*	48.6	48.0
Landmark	**72.5***	**66.8***	54.8*
Procedural	**72.5***	62.8*	**61.9***

TABLE III: Results on real robots. * means statistically significant difference compared to 'No Interaction' (wilcoxon test; p<0.05)

2) Evaluation Results: Tab. III displays the results of ORION under different feedback settings. We can see that leveraging user feedback enhances overall performance with similar trends in simulated environment results. Specifically, the landmark feedback yields substantial improvement in SR and SPL, as the selected landmarks are easily identifiable in the room, thus effectively narrowing down the search. While the procedural feedback provides only basic navigational cues about the goal, it still greatly enhances SIT. Given the room's straightforward layout, the robot often encounters potential goals during evaluation; therefore, the corrective feedback also largely improves the results by rectifying misidentified objects for ORION to update its memory. Comparatively, the descriptive feedback brings the least SPL and SIT as matching real-world observations with language descriptions using VLMs is still quite challenging. Common failure cases stem from two main sources: 1) low-level movement errors, which negatively impact task success performance; and 2) inaccurate detection, which fails to identify the correct objects in the robot's ego-view images.

V. Conclusion

This work introduces Zero-shot Interactive Personalized Object Navigation (ZIPON), an advanced version of zero-shot object navigation. In this task, a robot needs to navigate to personalized goal objects while engaging in natural language interactions with the user. To address the problem, we propose ORION, a general framework for open-world interactive personalized navigation, where the LLM serves as a decision-maker to direct different modules to search, perceive, navigate in the environment, and interact with the user. Our results in both simulated environments and the real world demonstrate the utilities of different types of language feedback. They also point out the challenges to obtaining a good balance between task success, navigation efficiency, and interaction efficiency. These findings will provide insights for future work on language communication for human-robot collaboration. This work is only our initial step in exploring LLMs in personalized navigation and has several limitations. For example, it does not handle broader goal types, such as image goals, or address multi-modal interactions with users in the real world. Our future efforts will expand on these dimensions to advance the adaptability and versatility of interactive robots in the human world.

References

[1] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in Neural Information Processing Systems*, 35:32340–32352, 2022.

[2] Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. *arXiv preprint arXiv:2301.13166*, 2023.

[3] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Clip on wheels: Zero-shot object navigation as object localization and exploration. *arXiv preprint arXiv:2203.10421*, 3(4):7, 2022.

[4] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[5] Boyuan Chen, Fei Xia, Brian Ichter, Kanishka Rao, Keerthana Gopalakrishnan, Michael S Ryoo, Austin Stone, and Daniel Kappler. Open-vocabulary queryable scene representations for real world planning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11509–11522. IEEE, 2023.

[6] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[7] Karmesh Yadav, Santhosh Kumar Ramakrishnan, John Turner, Aaron Gokaslan, Oleksandr Maksymets, Rishabh Jain, Ram Ramrakhya, Angel X Chang, Alexander Clegg, Manolis Savva, Eric Undersander, Devendra Singh Chaplot, and Dhruv Batra. Habitat challenge 2022, 2022.

[8] Karmesh Yadav, Jacob Krantz, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Jimmy Yang, Austin Wang, John Turner, Aaron Gokaslan, Vincent-Pierre Berges, Roozbeh Mootaghi, Oleksandr Maksymets, Angel X Chang, Manolis Savva, Alexander Clegg, Devendra Singh Chaplot, and Dhruv Batra. Habitat challenge 2023, 2023.

[9] Dhruv Shah, Błażej Osiński, Sergey Levine, et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on Robot Learning*, pages 492–504. PMLR, 2023.

[10] Nur Muhammad Mahi Shafiullah, Chris Paxton, Lerrel Pinto, Soumith Chintala, and Arthur Szlam. Clip-fields: Weakly supervised semantic fields for robotic memory. In *RSS*, 2023.

[11] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 815–824, 2023.

[12] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *arXiv preprint arXiv:2207.05608*, 2022.

[13] Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20, 2023.

[14] Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. *arXiv preprint arXiv:2307.01928*, 2023.

[15] Pratyusha Sharma, Balakumar Sundaralingam, Valts Blukis, Chris Paxton, Tucker Hermans, Antonio Torralba, Jacob Andreas, and Dieter Fox. Correcting robot plans with natural language feedback. In *RSS*, 2022.

[16] Yuchen Cui, Siddharth Karamcheti, Raj Palleti, Nidhya Shivakumar, Percy Liang, and Dorsa Sadigh. No, to the right: Online language corrections for robotic manipulation via shared autonomy. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 93–101, 2023.

[17] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR, 2020.

[18] Khanh X Nguyen, Yonatan Bisk, and Hal Daumé Iii. A framework for learning to request rich and contextually useful information from humans. In *International Conference on Machine Learning*, pages 16553–16568. PMLR, 2022.

[19] K. Dautenhahn. Robots we like to live with?! - a developmental perspective on a personalized, life-long robot companion. In *RO-MAN 2004. 13th IEEE International Workshop on Robot and Human Interactive Communication (IEEE Catalog No.04TH8759)*, pages 17–22, 2004.

[20] Caitlyn Clabaugh and Maja Matarić. Robots for the people, by the people: Personalizing human-machine interaction. *Science robotics*, 3(21):eaat7451, 2018.

[21] Mehdi Hellou, Norina Gasteiger, Jong Yoon Lim, Minsu Jang, and Ho Seok Ahn. Personalization and localization in human-robot interaction: A review of technical methods. *Robotics*, 10(4):120, 2021.

[22] Jorge de Heuvel, Nathan Corral, Lilli Bruckschen, and Maren Bennewitz. Learning personalized human-aware robot navigation using virtual reality demonstrations from a user study. In *2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 898–905. IEEE, 2022.

[23] Jorge de Heuvel, Nathan Corral, Benedikt Kreis, and Maren Bennewitz. Learning depth vision-based personalized robot navigation from dynamic demonstrations in virtual reality. In *IROS*, 2023.

[24] Jennifer Goetz, Sara Kiesler, and Aaron Powers. Matching robot appearance and behavior to tasks to improve human-robot cooperation. In *The 12th IEEE International Workshop on Robot and Human Interactive Communication, 2003. Proceedings. ROMAN 2003.*, pages 55–60. Ieee, 2003.

[25] Min Kyung Lee, Jodi Forlizzi, Sara Kiesler, Paul Rybski, John Antanitis, and Sarun Savetsila. Personalization in hri: A longitudinal field experiment. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 319–326, 2012.

[26] Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*, 2019.

[27] Maria Schmidt and Patricia Braunger. A survey on different means of personalized dialog output for an adaptive personal assistant. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*, pages 75–81, 2018.

[28] Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. Learning personalized end-to-end goal-oriented dialog. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6794–6801, 2019.

[29] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. Language-driven semantic segmentation. In *International Conference on Learning Representations*, 2022.

[30] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip. *Zenodo*, July 2021. If you use this software, please cite it as below.

[31] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[32] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pages 146–151. IEEE, 1997.

[33] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[34] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[35] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport

3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021.

[36] James A Sethian. A fast marching level set method for monotonically advancing fronts. *proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[37] Simon Keizer, Milica Gasic, Filip Jurcicek, François Mairesse, Blaise Thomson, Kai Yu, and Steve Young. Parameter estimation for agenda-based user simulation. In *Proceedings of the SIGDIAL 2010 Conference*, pages 116–123, 2010.

[38] Florian Kreyssig, Iñigo Casanueva, Paweł Budzianowski, and Milica Gašić. Neural user simulation for corpus-based policy optimisation of spoken dialogue systems. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 60–69, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[39] Bo-Hsiang Tseng, Yinpei Dai, Florian Kreyssig, and Bill Byrne. Transferable dialogue systems and user simulators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 152–166, Online, August 2021. Association for Computational Linguistics.

[40] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46, 2007.