Coursework Submission Coversheet

Please use this coversheet for all coursework assessment submissions.

You should copy the completed coversheet into your assessment document as the first page and save as one single document for submission.

If you require full guidance on coursework submission, please consult the Code of Practice on Assessment (available in your Programme Organisation in Minerva). In particular:

- See section 3.3d for the full policy on penalties for exceeding assessment limits (e.g. word count).
- See section 3.3e for the full policy on penalties for late submission of coursework.

To complete this coversheet, please provide the following information:

Module code: OCOM5300M

Module title: Artificial Intelligence Project

Assessment title: Exploring a Small Vision Language Model Approach for Vision and Language Navigation Tasks in Continuous Environments

Your 9-digit Student ID number: 201692702

Word count or minute count: 6,179

Submission date: 15 January 2025

Please note that by submitting this piece of work you are agreeing to the University's Declaration of Academic Integrity.

Exploring a Small Vision Language Model Approach for Vision and Language Navigation Tasks in Continuous Environments

Wesley Chiu, Abdulrahman Altahhan

University of Leeds, School of Computing, ODL MSc in AI, UK.
{od22wc, a.altahhan}@leeds.ac.uk

Abstract. Vision and Language Navigation (VLN) is a rapidly evolving field of research that aims to enable an embodied agent to follow textual instructions given in natural language to navigate through an unseen environment to a goal position. Existing approaches to this task all into two main categories: "textual" models that aim to tackle the challenge of multi-modality in the textual space, and "integrated" approaches that integrate vision and language models in the latent space. Regardless of the approach, prior work in the field have typically used the largest and most powerful variants of Large Language Models and Vision Models available (70B+ parameters). This work aims to evaluate the suitability of using a lighter-weight variant of an existing open-sourced model as the primary VLM, which would allow it to be run "on-device" rather than outsourcing its inferencing to a networked compute. The experiments in a simulated environment demonstrates that the current ability of the lighter-weight model is not yet fit for purpose, failing to reach the success rates seen in approaches that use the full-sized variants.
Code is available at: github.com/wsychiu/OCOM5300M_ResearchProject

Keywords: Vision-and-Language Navigation, Vision Language Model, Large Language Model, Prompting

1 Introduction

Vision and Language Navigation (VLN) tasks, which tasks an embodied agent to follow a set of textual instructions given in natural language to navigate a previously unseen 3D indoor environment, have received increasing attention in recent years. Whilst initial research leveraged models such as RNNs and CNNs to process natural language and vision data, the introduction of Transformer-based models [1] has accelerated capabilities in Natural Language Processing (NLP) and Computer Vision (CV) - both key components in the VLN task. The rapid development of Large Language Models (LLMs) and Vision-Language Models (VLMs) has further intensified the volume and pace of research in VLN, with each advance in LLM and VLM competency benefiting the VLN task. The original VLN task, as proposed by Andersen, et al. [2], requires the agent to navigate between pre-defined nodes in the environment (referred to as a navigation

graph), which essentially reduces the VLN task to a vision-based graph-search problem; models trained in this manner have difficulty translating performance to real-world environments [3]. To help combat this gap, the VLN Continuous Environment (VLN-CE) task was introduced, and has no such predefined navigation points, requiring agents to take low-level actions to reach a navigable point in the environment. This introduces new challenges for the models to overcome, such as avoiding getting stuck on obstacles and dealing with distances [4].

Approaches to the VLN task (whether graph-based VLN or VLN-CE) that leverage existing LLMs and VLMs typically fall into two broad categories: "textual" models or "integrated" models. Textual models [5, 6, 7, 8, 9] typically use an LLM, such as OpenAI's ChatGPT [10], with a vision model or image captioning model, such as BLIP [11], by 'connecting' the two models through text. The vision model will generate a textual description of the agent's observations before sending this description to the LLM. Whilst such approaches are able to interface between the two models without extensive training or finetuning, the richness of image data may be lost as it becomes summarised in textual form [6]. Integrated models [12, 13, 14, 15] connect the vision and language model(s) in the latent space. These approaches have exhibited stronger performance in VLN tasks compared to the textual model approach (at the time of their release), but are less unable to take advantage of more competent LLMs or VLMs as they are released in the same 'plug-and-play' manner of textual models. This is due to the extensive training and finetuning the integrated architecture requires.

The development of Vision Language Models (VLMs) pretrained on a variety of tasks and with strong multi-modal performance [16, 17, 18, 19, 10] suggests that an alternative method that strikes a balance between the two approaches could be viable. Additionally, most prior work typically leverage larger, complex language and vision models, often with parameter counts in the tens of billions, requiring powerful hardware to support during inferencing. The development of Low Rank Adaptation (LoRA) finetuning [20] combined with model quantization [21] allows for a LLM or VLM to be trained and run on hardware with more limited memory. This is particularly relevant to the VLN task, as lighter-weight models can be run more readily on smaller robots with more limited hardware, broadening the application of these models.

This work proposes, as its main contribution, to explore the effectiveness of using a small, finetuned, quantized open-sourced model as the primary VLM (specifically, Qwen2-VL-7B) to tackle the VLN-CE task and examines its performance against other approaches.

2 Literature Review

Vision and Language Navigation (VLN) The ability for an embodied agent to navigate a previously unseen environment purely by natural language instruc-

tion is a field of research that has seen increasing interest since the task was formally introduced by Andersen et al. [**8578485**]. The VLN task requires that an embodied agent be able to navigate an unstructured and previously unseen environment by following navigation instructions provided in natural language. Early research focused on the use of Recurrent Neural Networks (RNNs) [**8578485**, 22, 23, 24]; however, with the introduction of the Transformer archicture [1] and the subsequent development of the Large Language Model (LLM) [25, 26], recent research has focused on leveraging the NLP capabilities of LLMs; these approaches can be broadly categorised into two different approaches: a text-based approach and an integrated approach.

Textual vs Integrated Approaches The text-based approach seeks to address the challenge of multi-modality by translating visual features from the agent's observations into textual space. LangNav [6] and NavGPT [7], for example, used image captioning models such as BLIP [11, 27] and DETR [28] to caption images and identify objects in those images; this textual information was then passed to GPT-4 for decision making and other navigation tasks; OpenNav [9] takes a similar approach with open-sourced models. Variations on this approach include DiscussNav [5], which uses a similar mechanism to enable discussions between its multiple domain experts, as well as NavCoT [8], which asks the LLM to predict information about observations from future states and structures its reasonings via chain-of-thought prompting. Integrated approaches also leverage LLMs and image captioning or vision transformer models [29], but integrates both models in the latent space through pre-training. These models are typically more performant than the text-based approaches, as the richness of the image data remains captured and transferred between the vision and language models [6]. Early works in VLN typically required the fusion of text and image encoding via pretraining, such as PREVALENT [30], the CMA model [4], and Airbert [31]. Recent models build on this work and also grow in complexity as researchers work to capture more proficiency in the VLN task. HAMT [14] combined BERT [32] encodings with a custom-trained Vision Transformer [29] and a form of spatial-temporal encoding. Nav-GPT2 [13] integrates LLMs with InstructBLIP [33], a VLM, via a vision encoder [34]. In a slightly different approach, SayCan [35] connects an LLM with a model trained via Reinforcement Learning to robotic affordance. These models, at time of release, were all State of the Art (SOTA), and continue to be referenced as benchmarks against new approaches; however, to obtain such proficiency, extensive pretraining and finetuning is required.

Specialist Modules Independent of the approach, VLN models typically contain multiple "modules" that specialise in particular tasks [36]; these are either separate models or different types of prompts fed to a single model. Common modules include the action-decision (or policy) module, progress monitor module, and a navigation history module. LangNav [6], for example, uses LLaMA2-7B to both select the action and update historical trajectory by using different

prompt templates. NavGPT [7] takes a similar approach by using a Prompt Manager module with GPT-4 to dynamically change the prompt in response to changes in the agent's state. Some approaches [14, 15] have improved progress monitoring by creating specialist architectures to enhance the usefulness of the historical trajectory, whilst other work [37, 38] seeks to retain historical trajectory as a graph.

Vision Language Models Early research in VLN required researchers to tackle the task of multi-modal input by integrating separately trained language and vision models. Further research in multi-modal models have developed more tightly integrated foundation models such as InstructBLIP [33], which allow researchers to finetune a pretrained multi-modal model. More recent research has expanded the competency of multi-modal foundation models, with notable closed-sourced models such as OpenAI's GPT4-V [10] and o1 [39] models, as well as Google's Gemini [19], whilst open-sourced models, such as LLaVa [40] and Qwen2-VL [17], provide a lower-cost and more flexible alternative. Qwen2-VL, in particular, has released three variants of its VLM, with the full model containing 72B parameters, whilst the smallest contains just 2B; the 7B parameter variant, however, aims to strike a balance between model size and performance [41].

Challenges in Continuous Environments A key challenge for VLN agents trained with graph-style navigation ("high-level actions") is the ability to transfer the learning into a real world environment; Andersen et al. [42] note a degradation in performance due to these factors. There have been some efforts to translate models trained in navigation-graph based VLN (also referred to as discrete VLN) to continuous environment, but a performance gap remains [43]. Continuous environments remove the implicit assumptions in navigation-graph style environments, requiring agents to consider new challenges such as environment topology and imperfect localisation in their decision-making process to select the next best "low-level action" to take [4]. To address these challenges, the originally proposed navigation-graph based Matterport3D Simulator [2] is now eschewed in VLN research in favour of the Habitat-Sim simulator [44, 45, 46], a continuous environment simular.

3 Methodology

The approach taken in this work, as illustrated in Figure 1, comprises of using a single VLM as the primary component of the model to enable the model to determine its current location in the environment, track its historical trajectory, and to make decisions on its next action. A small component of this architecture leverages an LLM to break down the natural language instructions into discrete "Navigation Checkpoints", which is in line with prior work [38, 6, 7].

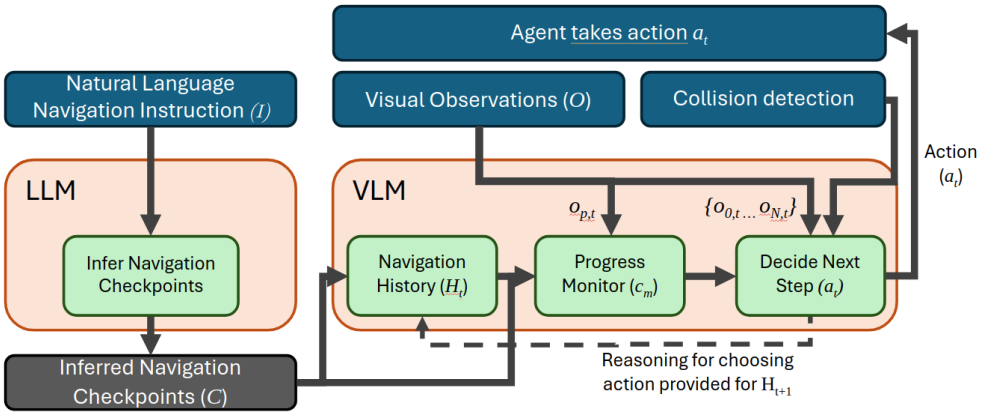

Fig. 1. The architecture of the model used in this work. The LLM is used only to create a structured list of Navigation Checkpoints that also act as sub-goals or medium-term goals for the agent to progress toward, and is useful for orienting the agent in instances where the final goal location may not be visible as well as for monitoring progress. The VLM, with its multi-modal capabilities, is used for all other tasks. Unlike prior work, with its pretrained vision and language capabilities, the VLM negates the requirement to run separate models for image captioning and reasoning tasks.

3.1 Problem Formulation

The VLN-CE task asks an embodied agent to autonomously navigate from a specified starting point to a goal location in an unseen environment by following a path described by natural language instructions. That is, given a natural language navigation instruction $\mathcal{I}$ and a starting state s_0, at each step t the agent must determine its current location in the environment and its current progress to take an action a_t that will lead it closer to the goal state s_G. The agent observes its environment through sensor observations O_t. In this work, O_t is comprised of colour images from N different views, each set at a different angle from the egocentric perspective of the agent. Additionally, the agent is also able to observe the environment through an egocentric panoramic image $o_{p,t}$. That is, the observations at each step is represented by $O_t = \{o_{p,t}, o_{0,t}, ..., o_{N,t}\}$. At each step, the agent must choose to take an action which corresponds to an image direction (excluding $o_{p,t}$), i.e. $a_t \in \{o_{0,t}, ..., o_{N,t}\}$, unless a specific direction is determined to result in a collision with the environment, in which case that particular direction is removed from the action set. The agent is deemed to have successfully reached s_G if the final state of the agent s_T, where T is the total number of steps taken, is within 3.0 metres of s_G.

3.2 Framework and Prompts

Unlike prior work that used separate Vision and Language models [7, 35] which require pre-training to combine the different models, this work proposes the use

of a singular VLM to manage both the image and textual inputs. The VLM is used for three different purposes: to track Navigation History, Decide the Next Navigation Checkpoint to work toward, and to Decide the Next Step. Only one task, creating Inferred Navigation Checkpoints, is not given to the VLM, and is instead given to the LLM.

Inferred Navigation Checkpoints The navigation instructions is given in natural language, and can include discourse markers that may not be relevant to the navigation itself (e.g. `"Okay, now ..."`) and may be confusing or difficult to track progress against. To address this, this work follows prior work [9, 13] and uses an LLM to convert $\mathcal{I}$ into a list of Navigation Checkpoints (C), which is easier for the model to track progress against. That is, for each $\mathcal{I}$, a series of M Navigation Checkpoints $C = \{C_0, ..., C_M\}$ are created such that by progressing from c_0 (the starting point of the instruction) to c_M will result in a successful trajectory (i.e. $s_T \leq s_G \pm 3.0$ metres). This can be represented as:

Navigation History A key component to the success of VLN agents is that at the time of deciding the optimal a_{t+1} the agent has the capability to retain in its context a history of its navigational trajectory [14, 15]. In this work, at each step, the VLM is prompted to generate h_t, which is a summary of the reasoning given for selecting a_t and the agent's current location based on $o_{p,t}$. The VLM is provided with C, $o_{p,t}$, a_t, the reasoning for select a_t, as well as the Navigation History $H_{t-1} = \{h_{t'}, ..., h_{t-1}\}$, where $t' = \max(t - 20, 0)$. Once generated, h_t is appended to the Navigation History, such that $H_t = \{h_{t'+1}, ..., h_t\}$. The Navigation History is limited to only the past 20 steps to manage VRAM limitations in the hardware; however, this also prevents actions taken much earlier in the trajectory, which are unlikely to be relevant to the current action, from entering the VLM's context window, thereby removing some noise from the data [15].

Deciding Next Checkpoint Each Navigation Checkpoint c_m acts as an medium-term goal for the agent to navigate toward, which is particularly important in trajectories where s_G is not visible from s_t. At each step, the VLM is provided with C, O_t, and H_t and is then prompted to state the list of all the Navigaton Checkpoints the agent has already achieved (c_0 to c_{m-1}), the Navigation Checkpoint the agent should now be working toward c_m, and the instructions associated with c_m.

Deciding Next Step The VLN-CE task requires the agent to decide at each step which of the low-level actions would be most likely to lead to c_m. At each step, the VLM is provided with $\{o_{0,t}, ..., o_{N,t}\}$ and the reasoning associated with c_m described in the "Deciding Next Checkpoint" section above. The VLM is then prompted to repeat the instructions for c_m and to select the o_n that is most aligned with that Navigation Checkpoint. During prompting, the list of available actions for that step (A_t) is modified to remove an action if the simulation detects that a collision with the environment would occur should that

step be taken; this is an emulation of a Laser Distance Scanner providing collision information to the agent and provides some environmental feedback to the agent.

3.3 Finetuning Data Generation for Continuous Environment

Each natural language navigation instruction provided is paired with a set of waypoints that describes the path that the original annotator used to create the instructions; this path can be referred to as the "gold label" trajectory that the agent should follow to reach the goal in as direct an approach as possible. This work uses the Room-across-Room (RxR) dataset [47], which was created using the Matterport3D simulator [48], a navigation-graph based simulator. As the provided waypoints do not describe a path that is relevant to the VLN-CE task, a new "gold label" trajectory and prompt finetuning data that is suitable for a continuous environment will need to be created.

Gold Label Trajectory The original trajectories from the RxR dataset were used to determine a new "gold label" trajectory for use in the continuous environment (CE-Trajectory). The agent was placed at the starting state s_0, and at each step t, all potential actions ($A_t = \{a'_{t,0}, ..., a'_{t,N}\}$) were assessed by determining the potential state that each action would result in ($S'_{t+1} = \{s'_{t+1,0}, ..., s'_{t+1,N}\}$). The next best action was determined in a greedy manner by the state $s'_{t+1,n}$ that was closest to the next waypoint in the original trajectory; this was repeated until $s_T \leq s_G \pm 3.0$ metres. Some of the RxR annotations and waypoints describe trajectories and actions that the agent specified in this work cannot take or would require a material deviation from the original trajectory to achieve. For example, some of these instructions may include: `"Hop over the coffee table"`, `"Hop over the table"`, or navigating to an outdoor staircase. To avoid these trajectories, any trajectories where $T > 50$ were rejected, resulting in 3,304 eligible CE-Trajectories.

Generating Training Prompts The training data required for finetuning the VLM requires that the training data include the prompts for the `user` and `assistant` roles. These prompts were generated by recreating the experience described in the "gold label" trajectories and capturing the prompts used in the Navigation History and Progress Monitor modules. A slightly adjusted version of the Decide Next Step module, the Rationalise Next Step, was created to *rationalise* the decision behind the provided action a_t at each step. This resulted in over 7,800 generated datapoints being created. The same process was used to create the calibration dataset for use in quantization, though only 20 datapoints were created for calibration.

4 Experiment

4.1 Datasets and Environment Simulation

Dataset The Matterport3D [48] dataset is a well known and commonly used dataset in VLN research. It comprises of 194,400 colour and depth (RGB-D) images to create panoramas at 10,800 different (internal) locations from 90 different residential buildings. These panoromas allows a Matterport3D Simulator user to view the environment in full in 360˚, and also allows users to "teleport" from one panoramic "node" to another. These panoramic nodes form the "navigation-graph" used in traditional or discrete VLN models. The RxR dataset was used for the natural language instructions, which comprises of over 126,000 natural language instructions (annotations) across three different languages to describe traversal through 16,500 different paths in the Matterport3D dataset. For this work, only the en-US guide annotations were considered, which is comprised of 13,992 annotations across 13,992 unique paths. Finally, the annotations are separated into the following splits: *train*, *seen val*, and *unseen val*. In order to align with prior work, the *train* split was used for generating finetuning data, whilst the *unseen val* split was used to test the model.

Environment Habitat-Sim [44, 45, 46] was used as the simulated environment, which converts the navigation graph-based Matterport3D dataset into continous environments. The environment was also set to allow sliding, which allows the agent to slide along an obstacle rather than being stuck against it. The agent in the simulator was given a colour sensor, through which colour images could be captured for the agent to observe the envrionment in. The agent was able to observe its environment in 6 different directions: Forward (0˚, straight ahead), Forward-left (45˚to its left, as measured from the forward direction), Left (90˚to its left), Forward-right (45˚to its right), Right (90˚to its right), and Behind (180˚from its forward direction); each observation has a 90˚horizontal and vertical field of view (FOV). In order to reduce file size and reduce training and inference time, the image dimensions were set at 64x64 pixels. The agent is able to take an action that corresponds to each of these observations, where "Forward" would result in a forward step of 1.0 metres, whilst all other actions would result in the agent turning to face that direction (e.g. "Forward-left" would result in turning 45˚to the left). The agent was also provided a colour sensor to create a panoramic colour observation, with an effective horizontal FOV of 360˚and a vertical FOV of 90˚, which was 320x64 pixels.

Evaluation Metrics The standard VLN evaluation metrics [2] include Success Rate (SR), the proportion of trajectories that ended within 3 metres of the goal location; Oracle Success Rate (OSR), the success rate regardless of whether the agent chose to stop at the goal location; Navigation Error (NE), the average distance between the agent's final location and the goal location in metres; and Trajectory Length (TL), the average distance travelled by the agent in metres. The exhaustive list of VLN evaluation metrics also include Success Rate

penalised by Path Length (SPL) to assess trajectory efficiency; Normalized Dynamic Time Warping (NDTW), the comparison between the ground-truth and actual path; and Success Rate penalised by NDTW (SNDTW). However, in this work, we will assess only the first four metrics, as model performance was such that the other metrics are not required to determine the effectiveness of the model or to help inform directions for future work.

4.2 Implementation Details

This work used a finetuned and quantized open-sourced VLM as the primary model to navigate an environment. All finetuning, quantization and inferencing of the model was run on a desktop computer with an AMD Ryzen 7 7800x3D CPU, 32GB of RAM, and an NVIDIA RTX4090 with 24GB of VRAM. Qwen2-VL-7B-Instruct [16, 17] was chosen as the base model for the VLM due to its strong performance relative to other open-sourced models [41, 49] of similar parameter size. The model was finetuned using Low Rank Adaptation (LoRA) [20], and quantized using Activation-aware Weight (AWQ) optimisation [21]. Both model finetuning and qunatization were conducted through LLaMa-Factory [50]. OpenAI's GPT-4 [10] was chosen as the LLM, which was used solely to convert the natural language navigation instructions from the RxR dataset into a structured list of Navigation Checkpoints as it demonstrated slightly stronger text-to-text generation than the VLM.

The finetuning dataset comprised of 150 trajectories from the `train` split of the RxR dataset, which is similar in magnitude to prior work [9] and allows inference time to be kept relatively low. All 150 trajectories were generated within 9 hours. Finetuning with LoRA took 16 minutes and quantization required 10 minutes. Finally, 152 trajectories were randomly sampled from the `val unseen` split of the RxR dataset, which required approximately 11 hours to complete; the number of trajectories was reduced from the full 1,517 trajectories available to ensure inferencing time remained reasonable.

4.3 Experiment Results

Results The performance of the model as implemented underperformed all but one of the earliest VLN-CE methods, CMA [4]. The performance of MapGPT [38] relative to NavGPT-2 [13], where both are discrete environment models, suggests that a zero-shot VLM approach can outperform a model with specialist architecture finetuned for VLN tasks. The performance of DiscussNav [5] indicates that zero-shot prompting can be performant on the VLN-CE task. This suggests that the underperformance observed in this model could stem from poor prompting strategies, or that the Qwen2-VL-7B model is insufficient for a task of this complexity, or both.

Qualitative Results An examination of the outputs of the model demonstrates some of the limitations of the model.

Table 1. Comparison of the model in this work against prior work in RxR dataset. Qwen2-VL-7B, as implemented in this work performs well below other models regardless of approach to multi-modality. Navigation Error as a proportion of Trajectory Length is high, suggesting that the model is not choosing to stop at the correct location. *: Results from VLN-CE task.

Method	Approach	Model(s)	↑SR	↑OSR	↓NE	TL
Human [2]		86	90	1.61	11.85	
DiscussNav* [5]	Text	GPT-4, InstructBLIP	43	61	5.32	9.69
LangNav[6]	Text	LLaMA2-7B, BLIP, DETR	34	45	7.10	–
NavCoT [8]	Text	LLaMA2-7B, BLIP	40	48	6.26	9.95
NavGPT [7]	Text	GPT-4, BLIP-2	34	42	6.46	11.45
Open-Nav* [9]	Text	LLaMA3.1-70B, SpatialBot, RAM	19	23	6.70	7.68
CMA* [4]	Integrated	Bi-Direct. LSTM, ResNet50	0.3	0.4	7.37	8.64
HAMT [14]	Integrated	Finetuned BERT, ViT-B/16	66	–	3.62	11.46
NavGPT-2 [13]	Integrated	FlanT5-XXL (11B), ViT-g/14	68	74	3.37	13.68
VLN BERT [12]	Integrated	V&L BERT	63	–	3.93	12.01
VLN BERT* [43]	Integrated	V&L BERT, ResNet	23	–	7.66	7.42
MapGPT [38]	VLM	GPT-4V	48	58	5.62	–
This work*	**VLM**	**Qwen2-VL-7B**	**3**	**11**	**11.88**	**12.64**

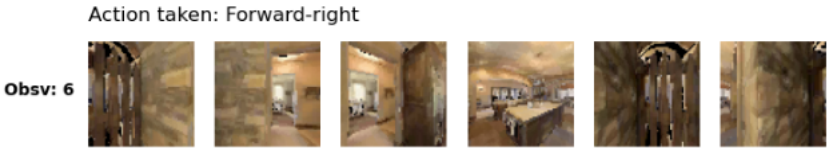

Fig. 2. Observation taken at Steps 6 of a trajectory. The agent is hallucinating, generating text that suggests that the first image shows a kitchen sink.

In Figure 2, the agent has identified that it is working on Navigation Checkpoint 1, which is to "`Turn your body to the left to face the sink area`"; however, it then describes Image 1 (left-most image in Figure 2) as, "`Image 1 shows the sink area, which is the target for Navigation Checkpoint 1...`". This is an example of the how the Navigation Checkpoints list provided to the Decide Next Step module has been added to the context of the text generation and negatively influenced the image description and action decision. Such hallucinations may be a result of the limits of the smaller Qwen2-VL-7B model, and the full-size model (with 72B parameters) may not suffer from the same issues.

Fig. 3. Observations taken at Steps 33 and 34 of a trajectory. The agent chooses to turn 180°a total of 10 times consecutively, switching between the observations seen in Step 33 and Step 34 and appears stuck in a cycle.

Another example is seen in a trajectory where the agent appears stuck in a cycle. At Step 33 ("Obvs: 33") in Figure 3, the action reasoning generated by the agent is:

```
I am currently working towards Navigation Checkpoint: 6
Navigation Checkpoint 6 instructions are to: Go through the open door
Image that best aligns with Navigation Checkpoint 6: Image 6 shows the
open door that leads to the next room, which is the next step in the
instructions
Selected image direction: Behind
```

However, the agent is mistaking Navigation Checkpoint 5 instructions (which it had already followed in Step 31) as Navigation Checkpoint 6; furthermore, Image 6 (the right-most image in Obvs: 33) shows an image of a closed door and not an open one as the VLM suggests. Finally, Navigation Checkpoint 6 does not appear in the list of Navigation Checkpoints. In step 34 (Obvs: 34), the agent repeats most of the same reasoning, except for the final line:

```
Why Image 6 was selected: Image 6 shows a hallway with a rug and a door,
```

`which aligns with the instruction to go through the open door`

In this instance, the VLM is suggesting that the agent can see a hallway in Image 6 (the right-most image), despite the observation suggesting a bedroom instead. The agent then repeats the reasoning for step 33 and step 34 in a cycle for 10 steps until the stochastic nature of the VLM results in choosing the action "`Forward`" in step 43. This episode, which is repeated in other trajectories, suggests that the prompting strategies used in the modules may have further room for refinement; in particular, the Navigation History module is not conveying the context of the agent being stuck in a loop and driving more exploratory behaviour.

5 Conclusion and Future Work

In this work, we developed a zero-shot small VLM-based approach to solve for the multi-modal nature of the VLN task by leveraging the vision and language capabilities of Qwen2-VL-7B. We introduced a method of generating VLN-CE training data from ground-truth data and annotations created from discrete, navigation-graph based environments, which we then used to finetune the VLM. Despite this work, it was demonstrated that the use of a single, small (7B parameter) VLM to drive a VLN agent in a continuous environment is unable to match the performance of approaches that use significantly larger models (70B+ parameters). Moving on from this work, it would be important to evaluate the performance of this approach using the Qwen2-VL-72B variant of the model to determine the performance gains from moving to a larger model. There is also significant scope for future work on refining and improving this approach. One potential area for investigation is the creation of dedicated finetuned models for each module; currently the approach described in this work uses the same VLM, which was finetuned with data relevant to all VLM modules. Investigation into adopting a discussion-based reasoning approach could also yield benefits, as demonstrated in prior work. Finally, the approach used in this work can be adapted easily to other models; as VLM development continues and increasingly competent models are released, there is potential for this approach to see improved performance.

Acknowledgements Aside from the API being used in the this work to generate the list of Navigation Checkpoints, OpenAI's ChatGPT-4o was also used (URL: https://chatgpt.com) to assist with code debugging, trouble-shooting, and for verifying understanding of models from prior work.
I would also like to thank Dr. Altahhan and the University of Leeds for the offer of ARC4 HPC Cluster compute; whilst this work chose ultimately not to use the resource, it was useful to have this resource in reserve if other available compute became unavailable.

References

[1] Ashish Vaswani et al. *Attention is All you Need*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[2] Peter Anderson et al. "Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3674–3683. DOI: 10.1109/CVPR.2018.00387.

[3] Peter Anderson et al. "Sim-to-Real Transfer for Vision-and-Language Navigation". In: *Proceedings of the 2020 Conference on Robot Learning*. Ed. by Jens Kober, Fabio Ramos, and Claire Tomlin. Vol. 155. Proceedings of Machine Learning Research. PMLR, 16–18 Nov 2021, pp. 671–681. URL: https://proceedings.mlr.press/v155/anderson21a.html.

[4] Jacob Krantz et al. *Beyond the Nav-Graph: Vision-and-Language Navigation in Continuous Environments*. 2020. arXiv: 2004.02857 [cs.CV]. URL: https://arxiv.org/abs/2004.02857.

[5] Yuxing Long et al. "Discuss before moving: Visual language navigation via multi-expert discussions". In: *arXiv preprint arXiv:2309.11382* (2023).

[6] Bowen Pan et al. *LangNav: Language as a Perceptual Representation for Navigation*. 2024. arXiv: 2310.07889 [cs.CV]. URL: https://arxiv.org/abs/2310.07889.

[7] Gengze Zhou, Yicong Hong, and Qi Wu. "NavGPT: Explicit Reasoning in Vision-and-Language Navigation with Large Language Models". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 7. 2024, pp. 7641–7649. DOI: 10.1609/aaai.v38i7.28597.

[8] Bingqian Lin et al. *NavCoT: Boosting LLM-Based Vision-and-Language Navigation via Learning Disentangled Reasoning*. 2024. arXiv: 2403.07376 [cs.CV]. URL: https://arxiv.org/abs/2403.07376.

[9] Yanyuan Qiao et al. *Open-Nav: Exploring Zero-Shot Vision-and-Language Navigation in Continuous Environment with Open-Source LLMs*. 2024. arXiv: 2409.18794 [cs.RO]. URL: https://arxiv.org/abs/2409.18794.

[10] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[11] Junnan Li et al. *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*. 2022. arXiv: 2201.12086 [cs.CV]. URL: https://arxiv.org/abs/2201.12086.

[12] Yicong Hong et al. "VLN BERT: A recurrent vision-and-language bert for navigation". In: *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*. 2021, pp. 1643–1653.

[13] Gengze Zhou et al. "NavGPT-2: Unleashing Navigational Reasoning Capability for Large Vision-Language Models". In: *Computer Vision – ECCV 2024*. Ed. by Aleš Leonardis et al. Cham: Springer Nature Switzerland, 2025, pp. 260–278. ISBN: 978-3-031-72667-5.

[14] Shizhe Chen et al. "History aware multimodal transformer for vision-and-language navigation". In: *Advances in neural information processing systems* 34 (2021), pp. 5834–5847.

[15] Keji He et al. "Memory-Adaptive Vision-and-Language Navigation". In: *Pattern Recognition* 153 (2024), p. 110511. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2024.110511. URL: https://www.sciencedirect.com/science/article/pii/S0031320324002620.

[16] Jinze Bai et al. "Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond". In: *arXiv preprint arXiv:2308.12966* (2023).

[17] Peng Wang et al. "Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution". In: *arXiv preprint arXiv:2409.12191* (2024).

[18] An Yang et al. "Qwen2 Technical Report". In: *arXiv preprint arXiv:2407.10671* (2024).

[19] Gemini Team et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2024. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805.

[20] Edward J. Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *CoRR* abs/2106.09685 (2021). arXiv: 2106.09685. URL: https://arxiv.org/abs/2106.09685.

[21] Ji Lin et al. "AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration". In: *Proceedings of Machine Learning and Systems*. Ed. by P. Gibbons, G. Pekhimenko, and C. De Sa. Vol. 6. 2024, pp. 87–100. URL: https://proceedings.mlsys.org/paper_files/paper/2024/file/42a452cbafa9dd64e9ba4aa95cc1ef21-Paper-Conference.pdf.

[22] Chih-Yao Ma et al. "The Regretful Agent: Heuristic-Aided Navigation Through Progress Estimation". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 6725–6733. DOI: 10.1109/CVPR.2019.00689.

[23] Xin Wang et al. "Reinforced Cross-Modal Matching and Self-Supervised Imitation Learning for Vision-Language Navigation". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 6622–6631. DOI: 10.1109/CVPR.2019.00679.

[24] Xiujun Li et al. *Robust Navigation with Language Pretraining and Stochastic Sampling*. 2019. arXiv: 1909.02244 [cs.CL]. URL: https://arxiv.org/abs/1909.02244.

[25] Alec Radford. "Improving language understanding by generative pre-training". In: (2018).

[26] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[27] Junnan Li et al. *BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models*. 2023. arXiv: 2301.12597 [cs.CV]. URL: https://arxiv.org/abs/2301.12597.

[28] Xizhou Zhu et al. *Deformable DETR: Deformable Transformers for End-to-End Object Detection*. 2021. arXiv: 2010.04159 [cs.CV]. URL: https://arxiv.org/abs/2010.04159.

[29] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[30] Weituo Hao et al. "Towards Learning a Generic Agent for Vision-and-Language Navigation via Pre-Training". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 13134–13143. DOI: 10.1109/CVPR42600.2020.01315.

[31] Pierre-Louis Guhur et al. "Airbert: In-Domain Pretraining for Vision-and-Language Navigation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 1634–1643.

[32] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423/.

[33] Wenliang Dai et al. *InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning*. 2023. arXiv: 2305.06500 [cs.CV]. URL: https://arxiv.org/abs/2305.06500.

[34] Yuxin Fang et al. "EVA: Exploring the Limits of Masked Visual Representation Learning at Scale". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 19358–19369. DOI: 10.1109/CVPR52729.2023.01855.

[35] Alex Irpan et al. "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances". In: 2022. URL: https://arxiv.org/abs/2204.01691.

[36] Wansen Wu, Tao Chang, and Xinmeng Li. *Vision-Language Navigation: A Survey and Taxonomy*. 2022. arXiv: 2108.11544 [cs.CV]. URL: https://arxiv.org/abs/2108.11544.

[37] Shizhe Chen et al. "Think Global, Act Local: Dual-scale Graph Transformer for Vision-and-Language Navigation". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 16516–16526. DOI: 10.1109/CVPR52688.2022.01604.

[38] Jiaqi Chen et al. "MapGPT: Map-Guided Prompting with Adaptive Path Planning for Vision-and-Language Navigation". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar.

Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 9796–9810. DOI: 10.18653/v1/2024.acl-long.529. URL: https://aclanthology.org/2024.acl-long.529/.

[39] OpenAI. *Learning to reason with LLMs*. https://openai.com/index/learning-to-reason-with-llms/. [Online]. Accessed: 2025-01-10. 2024.

[40] Haotian Liu et al. *Visual Instruction Tuning*. 2023. arXiv: 2304.08485 [cs.CV]. URL: https://arxiv.org/abs/2304.08485.

[41] Hugging Face. *Open VLM Leaderboard*. https://huggingface.co/spaces/opencompass/open_vlm_leaderboard. [Online]. Accessed: 2025-01-11. 2024.

[42] Peter Anderson et al. *Sim-to-Real Transfer for Vision-and-Language Navigation*. 2020. arXiv: 2011.03807 [cs.CV]. URL: https://arxiv.org/abs/2011.03807.

[43] Yicong Hong et al. "Bridging the Gap Between Learning in Discrete and Continuous Environments for Vision-and-Language Navigation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 15439–15449.

[44] Manolis Savva et al. "Habitat: A Platform for Embodied AI Research". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[45] Andrew Szot et al. "Habitat 2.0: Training Home Assistants to Rearrange their Habitat". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021.

[46] Xavi Puig et al. *Habitat 3.0: A Co-Habitat for Humans, Avatars and Robots*. 2023.

[47] Alexander Ku et al. "Room-Across-Room: Multilingual Vision-and-Language Navigation with Dense Spatiotemporal Grounding". In: *Conference on Empirical Methods for Natural Language Processing (EMNLP)*. 2020.

[48] Angel Chang et al. "Matterport3D: Learning from RGB-D Data in Indoor Environments". In: *International Conference on 3D Vision (3DV)* (2017).

[49] Trong-Hieu Nguyen-Mau et al. "Enhancing Visual Question Answering with Pre-trained Vision-Language Models: An Ensemble Approach at the LAVA Challenge 2024". In: *Proceedings of the Asian Conference on Computer Vision (ACCV) Workshops*. Dec. 2024, pp. 275–286.

[50] Yaowei Zheng et al. "LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*. Bangkok, Thailand: Association for Computational Linguistics, 2024. URL: http://arxiv.org/abs/2403.13372.