Appendix: Scaling Data Generation in Vision-and-Language Navigation

Zun Wang*♠[1,2] Jialu Li*[3] Yicong Hong*†[1]
Yi Wang[2] Qi Wu[4] Mohit Bansal[3] Stephen Gould[1] Hao Tan[5] Yu Qiao[2]
[1]The Australian National University [2]OpenGVLab, Shanghai AI Laboratory
[3]UNC, Chapel Hill [4]University of Adelaide [5]Adobe Research
wangzun@pjlab.org.cn, jialuli@cs.unc.edu, mr.yiconghong@gmail.com
Project URL: https://github.com/wz0919/ScaleVLN

We first describe the implementation details of our experiments in Sec. A, including pre-training objectives and details of REVERIE experiments. In Sec. B, we provide additional experiments about the effects of visual encoders, model initialization, and adding depth features. We then discuss the impact of ScaleVLN on different VLN agents and on learning the long-horizon VLN task (R4R). Leaderboard results of R2R and object grounding results for REVERIE are also included. Sec. C and Sec. D visualize our navigability graphs and the recovered images from Co-Modulated GAN [18].

A. Implementation Details (§4[1])

A.1. Pre-Training Objectives (§4.1)

We mainly employ three proxy tasks, MLM, MRM, and SAP, for pre-training the agent. Here we describe these proxy tasks in detail. The inputs for these tasks are instruction $\mathcal{W}$ and demonstration path $\mathcal{P}$. During training, we randomly sample one task for each iteration with equal probability.

Masked Language Modeling (MLM) involves predicting masked words based on textual context and the full trajectory. A special `[mask]` token is used to randomly mask out 15% of the tokens in $\mathcal{W}$. We predict the masked word distribution $p(w_i|\mathcal{W}_{\backslash i}, \mathcal{P}) = f_{\text{MLM}}(x'_i)$ through a two-layer fully-connected network, where $\mathcal{W}_{\backslash i}$ is the masked instruction and x'_i is the output embedding of the masked word w_i. The objective is to minimize the negative log-likelihood of predicting the original words: $\mathcal{L}_{\text{MLM}} = -\log p(w_i|\mathcal{W}_{\backslash i}, \mathcal{P})$.

Masked Region Modeling (MRM) is to predict labels for masked regions in history observations based on instructions and neighboring regions. To achieve this, we randomly remove view images in $\mathcal{P}$ with a 15% probability. For view images, the target labels are determined by an image classification model [6] pre-trained on ImageNet. To predict semantic labels for each masked visual token, we use a two-layer fully-connected network. The objective is to minimize the KL-divergence between the predicted and target probability distribution.

Single Action Prediction (SAP) aims to predict the next action based on the instruction and the given path. Following [5], we predict the probability for each candidate action in the action space via a two-layer fully-connected network. The objective is to minimize the negative log probability of the target view action $\mathcal{L}_{\text{SAP}} = -\log p_t(a^*_t|\mathcal{W}, \mathcal{P}_{<t})$.

A.2. Implementation Details of REVERIE (§4.1)

REVERIE data contains trajectories that lead to target objects specified by high-level instructions. Following AutoVLN [4], for every visible object at a viewpoint, we sample paths with an edge length between 4 and 9 that end at the viewpoint. We filter out objects that are more than 3 meters away from the central of the viewpoint, resulting in 518,233 paths from HM3D, and 311,976 paths from the Gibson environments. To generate instructions in REVERIE-style, we modify the GPT-2 architecture used in AutoVLN [4] by only encoding the target object in the final viewpoint as the prompt to generate the instructions. Our large-scale data augmentation paradigm creates 830,209 instruction-trajectory pairs for training. This size is ×38 larger than the original REVERIE dataset, and ×3.81 larger than the augmented dataset in AutoVLN [4].

We follow DUET and SIA [13] to pre-train the model with an additional Object Grounding (OG) task, which requires selecting a target from object candidates based on high-level instruction and observations along the path. We use CLIP ViT-H/14 [14] to extract the image features, and ViT-B/16 [6] pre-trained on ImageNet to extract the object features. We pre-train DUET for 100k iterations with a batch size of 128 and a learning rate of 5×10^{-5} on both

[1] Link to Section 4 in Main Paper.

HM3D and Gibson environments. We compare three model checkpoints at 30k, 40k, and 50k and pick the one with the highest fine-tuning performance. Then we fine-tune DUET for 150k iterations, with batch size 32 and learning rate 2×10^{-5} on a single NVIDIA A100 GPU.

B. Additional Experiments (§4)

Here we provide additional experiments to investigate the effect of visual encoder, model initialization, and depth features. We also experiment with different model architectures (*i.e.*, HAMT [3]) on R2R dataset, and show object grounding results for the REVERIE task.

B.1. Effect of Visual Encoders (§4.2)

We study the effect of visual encoders in Table 1. Here we adopt CLIP's ViT backbone with different model sizes and input patches (*i.e.*, Base/16, Large/14, and Huge/14). We can see that the vision encoder has a major influence on SPL, suggesting the agent can make fewer wrong steps and is capable of efficient navigation.

Visual Encoders	R2R Val-Seen				R2R Val-Unseen			
	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
CLIP-ViT-B/16	12.41	**2.02**	80.51	74.88	13.16	2.53	78.08	68.31
CLIP-ViT-L/14	12.62	2.16	80.04	74.06	13.13	2.50	78.08	68.97
CLIP-ViT-H/14	12.53	2.15	**81.19**	**76.83**	12.61	**2.49**	**78.20**	**69.71**

Table 1: Effect of visual encoders.

B.2. Effect of Initialization (§4.2)

Table 2 presents the performance of initializing the navigation agent with different pre-trained models in pre-training. We discovered that utilizing BERT to initialize the language encoder does not enhance downstream performance, and even harms the performance on the validation unseen set. We attribute this to the vast domain gap between uni-modal BERT's language representations and CLIP's visual representation. Results could be improved by initializing the model with LXMERT's language encoder [15], and even more by utilizing both the language encoder and cross-modal encoder from LXMERT, indicating that incorporating pre-trained vision-and-language models could benefit agent performance.

B.3. Effect of Depth Modality (§4.2)

We also explored leveraging depth information to improve visual representations as described in Table 3. In line with previous methods such as [12, 11, 8, 1], we directly concatenate the depth features from DDPPO [17] (a ResNet backbone pre-trained on PointGoal navigation with depth inputs) and the RGB features (from CLIP ViT-B/16) to create the visual representations. Our findings indicate that when not using HM3D as the augmented environment, the agent's SR is significantly better if learning from the additional depth input. However, this conclusion changes when HM3D environments are involved: the agent's SR with RGBD was slightly lower than with RGB-only. We suspect that as the data is scaled up with more visual observations and language instructions, the agent may not require additional depth information to assist decision-making.

Language Encoder Initialization	R2R Val-Seen				R2R Val-Unseen			
	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
Random	12.87	2.29	78.75	72.61	12.69	2.72	75.65	67.00
BERT	12.43	2.29	79.04	73.72	12.95	2.76	75.01	66.57
LXMERT (lang.)	11.73	**2.07**	**80.22**	**75.65**	13.17	2.67	75.86	67.36
LXMERT (lang.+cross.)	12.63	2.27	79.24	73.34	12.83	**2.62**	**76.59**	**67.74**

Table 2: Effect of different initialization, where *LXMERT (lang.)* means only initialize the language encoder with LXMERT, and *LXMERT (lang.+cross.)* means initialize both the langauge encoder and cross modal encoder with LXMERT.

HM3D Aug	Sensor	R2R Val-Seen				R2R Val-Unseen			
		TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
×	RGB	13.28	**2.51**	76.89	69.71	13.53	3.06	72.92	**62.82**
	RGBD	14.16	2.54	**77.18**	**69.76**	15.14	**3.02**	**74.12**	62.54
✓	RGB	12.63	2.27	79.24	73.34	12.83	**2.62**	**76.59**	67.74
	RGBD	11.24	**2.12**	**79.73**	**75.45**	12.93	2.63	76.46	**68.52**

Table 3: Effect of adding depth modality.

B.4. ScaleVLN with Different VLN Models (§4.2)

To evaluate the generalization ability of our ScaleVLN dataset, we also apply the augmented data to train different VLN agents, including Seq2Seq [2], EnvDrop [16], and HAMT [3]. The HAMT model is pre-trained and fine-tuned with the same data and configurations as we pre-trained the DUET model, while we follow similar configurations of Seq2Seq and Envdrop to the original papers. All three agents are trained with the CLIP ViT-B-16 feature. The results are shown in Table 4. Compared to using only PREVALENT [7] for augmentation, All three models significantly benefit from incorporating the ScaleVLN dataset, with 12.2%, 3.8%, 5.5% absolute increase in SR for Seq2Seq, EnvDrop, and HAMT, respectively. This shows that ScaleVLN strengthens models' generalization ability. Note that Seq2Seq and Envdrop perform better on Val-Seen when using PREVALENT, mainly caused by overfitting the training environments.

B.5. ScaleVLN for Long-Horizon VLN (§4.2)

We evaluate the impact of our dataset on a long-horizon VLN dataset, R4R [10]. R4R extends the R2R dataset by concatenating two adjacent trajectories in R2R, resulting in

Model	Pre-Train Data	Fine-Tune Data	R2R Val-Seen			R2R Val-Unseen		
			NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑
Seq2Seq [2]	-	R2R, PREV	**3.89**	**58.18**	**38.49**	6.32	37.34	23.21
	-	R2R, ScaleVLN	4.78	49.85	36.32	**5.20**	**47.51**	**34.81**
Envdrop [16]	-	R2R, PREV	**3.65**	**66.12**	**61.72**	4.41	59.22	52.35
	-	R2R, ScaleVLN	3.70	65.23	59.06	**3.99**	**63.01**	**54.93**
HAMT [3]	R2R, PREV	R2R, PREV	2.58	74.93	71.52	3.69	64.90	60.11
	R2R, PREV, ScaleVLN	R2R	**2.15**	**79.53**	**76.64**	3.43	67.56	62.32
	R2R, PREV, ScaleVLN	R2R, ScaleVLN	2.43	76.40	73.30	**3.07**	**70.46**	**65.12**

Table 4: Influence of ScaleVLN on different VLN models.

longer navigation trajectories not biased by the shortest path prior. We directly fine-tune our pre-trained HAMT models from Table 4 on R4R. Compared to pre-training with only R2R and PREVALENT, adding our ScaleVLN dataset in the pre-training stage leads to a consistent gain, yielding +2.7% SR, +1.5% nDTW and +2.7% SDTW [9]. As suggested by the large improvement in nDTW between the ground-truth path and the executed path, our ScaleVLN data not only facilitate the model to reach the target but also follow the path described by the given instruction.

Pre-Train Data	Fine-Tune Data	R4R Val-Unseen				
		NE↓	SR↑	CLS↓	NDTW↑	SDTW↑
R2R, PREV	R4R	6.19	41.52	57.89	51.21	30.00
R2R, PREV, ScaleVLN	R4R	**6.09**	**44.20**	**59.55**	**52.77**	**32.73**

Table 5: Effect of ScaleVLN on learning R4R.

B.6. Leaderboard Results of R2R (§4.4)

We report the top seven submissions on the test-unseen leaderboard of R2R[2] (Table 6). When ranking with success rate, we can see that (a) most methods have extremely low SPL (1%) due to using beam search to find the optimal paths. Even so, our single-run result (*EarlyToBed*) outperforms them by a large margin. When ranking with SPL (b), some methods pre-explored the test environments but their results are still much worse than ours. Apart from human followers, we are currently ranked first on the leaderboard.

Team	NE↓	SR↑	SPL↑
human	1.61	86	76
EarlyToBed (ours)	2.27	80	70
LILY°	2.54	78	1
Airbert°	2.50	78	1
Shortest-Path-Prior°	3.55	74	1
UU_77	3.00	74	63
TAIIC°	2.99	74	1

(a) Top 7 in SR.

Team	NE↓	SR↑	SPL↑
human	1.61	86	76
EarlyToBed (ours)	2.27	80	70
TAIICX†	3.00	73	69
Active Exploration†	3.30	70	68
sponge	3.26	71	67
Auxiliary Reasoning†	3.96	68	65
SE-Mixed	3.52	70	65

(b) Top 7 in SPL.

Table 6: R2R leaderboard results (28.JUL.2023). °: Beam search. †: Pre-exploration.

B.7. REVERIE Object Grounding Result (§4.4)

We report the success rate of remote object grounding (RGS) and its path length-weighted result (RGSPL). As shown in Table 7, ScaleVLN achieves state-of-the-art performance on object grounding task on the test leaderboard, comparable to the previous best method AutoVLN [4].

Models	REVERIE Val-Unseen				REVERIE Test-Unseen			
	SR↑	SPL↑	RGS↑	RGSPL↑	SR↑	SPL↑	RGS↑	RGSPL↑
SIA [13]	31.53	16.28	22.41	11.56	30.80	14.85	19.02	9.20
HAMT [3]	32.95	30.20	18.92	17.28	30.40	26.67	14.88	13.08
DUET [5]	46.98	33.73	32.15	23.03	52.51	36.06	31.88	22.06
AutoVLN [4]	55.89	40.85	**36.58**	**26.76**	55.17	38.88	32.23	22.68
DUET+ScaleVLN(ours)	**56.97**	**41.84**	35.76	26.05	**56.13**	**39.52**	**32.53**	**22.78**

Table 7: Object grounding performance on REVERIE.

C. Comparison of Navigability Graphs (§3.2)

We visualize the navigability graphs produced by AutoVLN [4] and our method for several HM3D environments in Figure 1. We can see that our graphs are denser, covering more regions, have viewpoints away from obstacles, and are fully traversable in open space.

D. Recover High Quality Images (§3.2)

As introduced in Main Paper §3.2, we apply the Co-Modulated GAN [18] to recover the corrupted images rendered from the HM3D and Gibson environments. Specifically, we first render a panorama of shape 512×1024 from the 3D mesh at each viewpoint. Then, we crop four images of shape 512×512 centered at 0°, 90°, 180° and 270° of the panorama (with overlapping), and recover them separately. Note that, in VLN, the panoramic observation at a viewpoint is represented by 36 single-view images at 12 viewing angles and three elevations [2]. We directly extract their corresponding regions from the four recovered images to obtain these single-view images for pre-training an agent.

Table 8 visualizes the difference between the rendered images and our recovered images. First, we can see that our method can recover missing regions, including outdoor scenes such as sky and trees (Example 1 & 4) and indoor scenes such as floor and walls (Example 6). Besides, the recovered images usually have less blurry or distorted areas, and the object boundaries are much clearer and sharper. For instance, the ceiling light in Example 2, the chairs in Example 3, and the door frames in Example 5. Even for the highly corrupted images from Gibson (Examples 4–6), we can see that the method can still recover the scene to a reasonable quality.

[2] R2R test server: https://eval.ai/web/challenges/challenge-page/97/leaderboard/270.

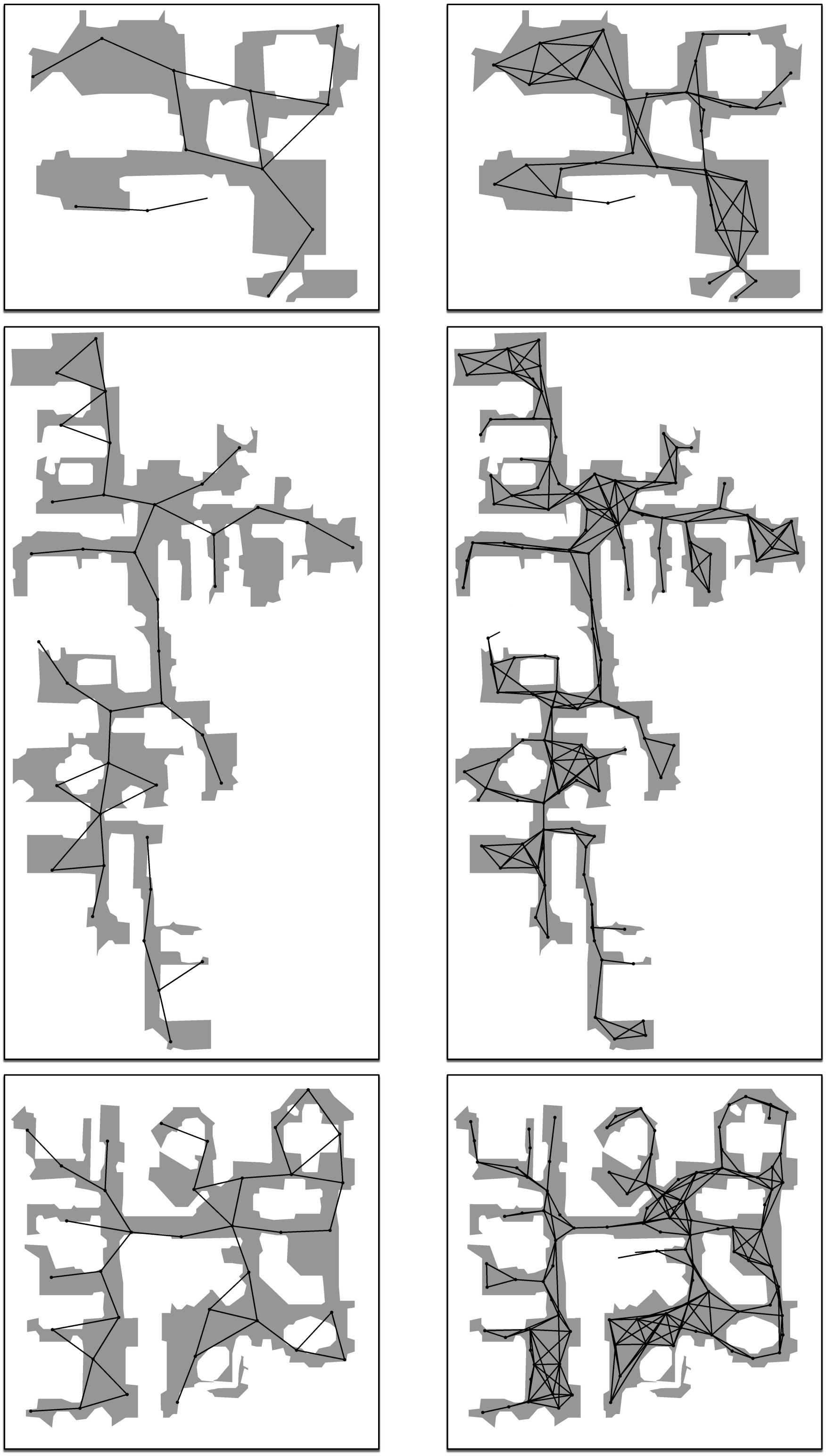


Figure 1: Comparison of navigability graphs between AutoVLN [4] and our ScaleVLN.

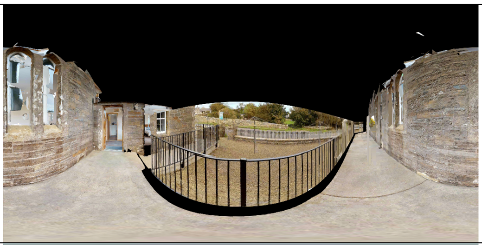
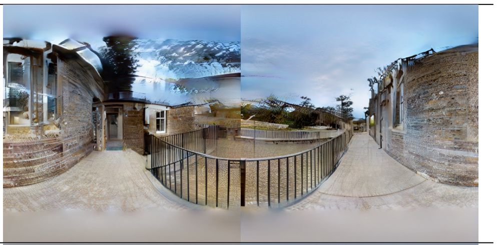
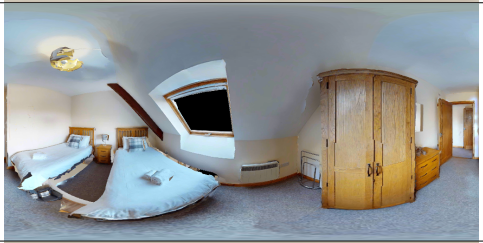
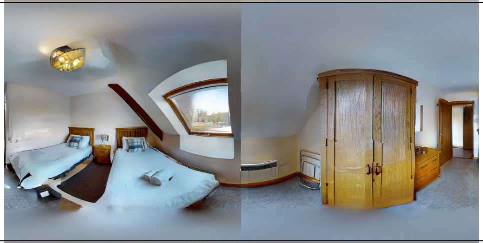
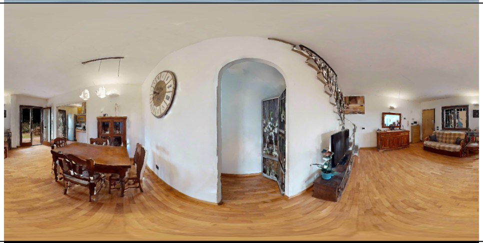
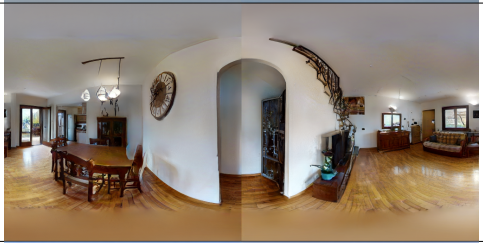
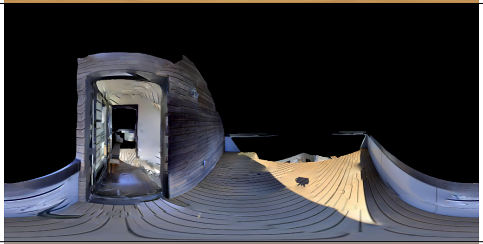
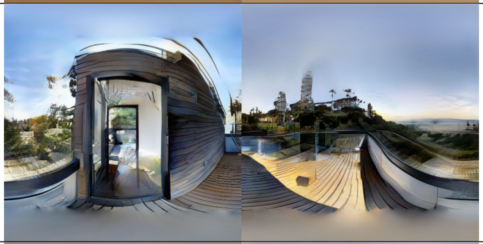
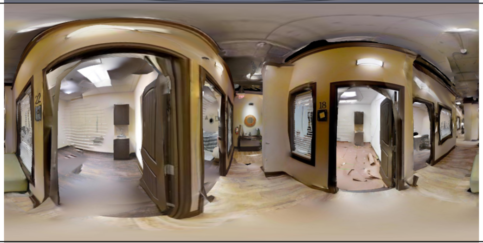
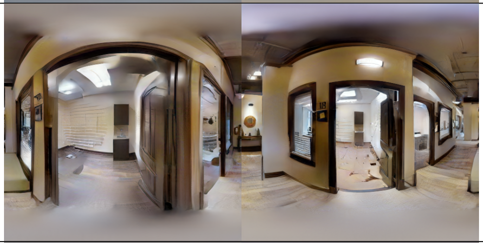
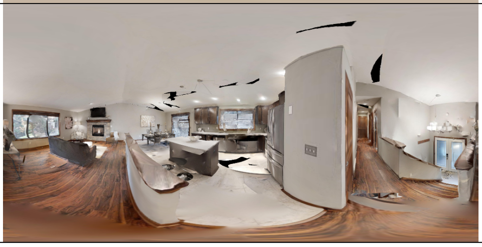
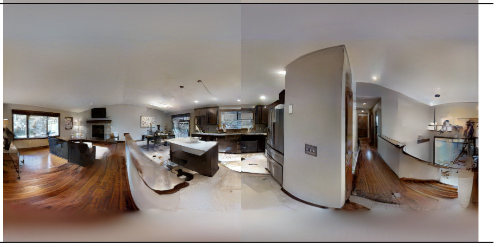
Examples	Environments	Rendered	Recovered
1	HM3D		
2	HM3D		
3	HM3D		
4	Gibson		
5	Gibson		
6	Gibson		

Table 8: Qualitative examples of our recovered images from HM3D and Gibson environments. The vertical line at the middle of panorama is caused by directly sticking two independently recovered images at 0° and 180°, which will not appear in the resulting augmented data, as explained in Appendix §D.

References

[1] Dong An, Zun Wang, Yangguang Li, Yi Wang, Yicong Hong, Yan Huang, Liang Wang, and Jing Shao. 1st place solutions for rxr-habitat vision-and-language navigation competition (cvpr 2022). *arXiv preprint arXiv:2206.11610*, 2022. 2

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018. 2, 3

[3] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34:5834–5847, 2021. 2, 3

[4] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Learning from unlabeled 3d environments for vision-and-language navigation. In *European Conference on Computer Vision*, pages 638–655. Springer, 2022. 1, 3, 4

[5] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16537–16547, 2022. 1, 3

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[7] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146, 2020. 2

[8] Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. 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*, pages 15439–15449, 2022. 2

[9] Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*, 2019. 3

[10] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1862–1872, 2019. 2

[11] Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15162–15171, 2021. 2

[12] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, 2020. 2

[13] Xiangru Lin, Guanbin Li, and Yizhou Yu. Scene-intuitive agent for remote embodied visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7036–7045, 2021. 1, 3

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1

[15] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019. 2

[16] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of NAACL-HLT*, pages 2610–2621, 2019. 2, 3

[17] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[18] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large scale image completion via co-modulated generative adversarial networks. *arXiv preprint arXiv:2103.10428*, 2021. 1, 3