RREx-BoT: Remote Referring Expressions with a Bag of Tricks

Gunnar A. Sigurdsson Jesse Thomason Gaurav S. Sukhatme Robinson Piramuthu

Amazon Alexa AI

***Abstract*— Household robots operate in the same space for years. Such robots incrementally build dynamic maps that can be used for tasks requiring remote object localization. However, benchmarks in robot learning often test generalization through inference on tasks in unobserved environments. In an observed environment, locating an object is reduced to choosing from among all object proposals in the environment, which may number in the 100,000s. Armed with this intuition, using only a generic vision-language scoring model with minor modifications for 3d encoding and operating in an embodied environment, we demonstrate an absolute performance gain of 9.84% on remote object grounding above state of the art models for REVERIE and of 5.04% on FAO. When allowed to pre-explore an environment, we also exceed the previous state of the art pre-exploration method on REVERIE. Additionally, we demonstrate our model on a real-world TurtleBot platform, highlighting the simplicity and usefulness of the approach. Our analysis outlines a "bag of tricks" essential for accomplishing this task, from utilizing 3d coordinates and context, to generalizing vision-language models to large 3d search spaces.**

I. Introduction

Household robots persist in the home over time, enabling them to build up knowledge of rooms, furniture, and objects. For remote object grounding, an embodied agent is given a language description of a referent object, then conducts navigation and identification to localize that object. We consider remote object grounding for agents that, like household robots, have familiarity with the environment.

Remote object grounding approaches mostly assume an environment is *totally unseen*. However, methods for vision-and-language navigation frequently pre-explore environments [1], or use beam search over trajectory rollouts at inference time [2], [3]. We bring exploration to the remote object grounding task. Our problem formulation selects the best match for a referring expression from hundreds of thousands of objects in a environment (Figure 1).

With Remote Referring Expressions with a Bag of Tricks (RREx-BoT), we set a new state of the art on remote object grounding without pre-exploration (42.07% versus the 32.23% of AutoVLN [4]) and with pre-exploration (42.07% versus the 34.69% of OSMaN [5]) on the REVERIE [6] test set. RREx-BoT combines the strengths of two previous approaches. OSMaN [5] provides an agent explicitly with a gold standard map and object region candidates. The DUET model [7], by contrast, dynamically explores an unseen environment rather than taking advantage of a given map. RREx-BoT actively traverses the environment to learn a map, detecting object regions along the way, and identifies which detected object in the 3D environment most closely matches the provided language referring expression.

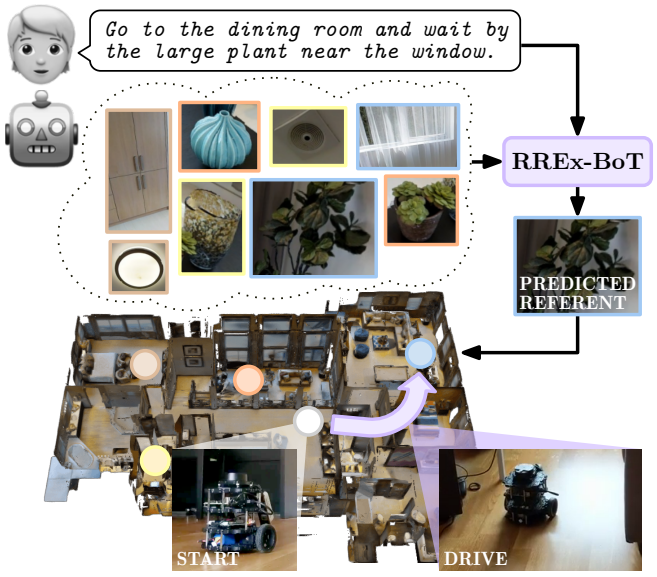

Fig. 1. RREx-BoT achieves state of the art performance on REVERIE and FAO by learning a ranking model on top of a generic vision-and-language backbone to score region proposals within a radius of an agent's starting position against a referring expression. The agent simply selects the highest scoring proposal as the referent and drives to it from the starting position. We deploy RREx-BoT out-of-the-box to a physical robot platform for real world language-guided object navigation.

We formulate a "bag of tricks" that enable RREx-BoT to *efficiently* utilize a large-scale, pretrained language-and-vision alignment model as a backbone to score hundreds of thousands of objects against an input referring expression.

- We generalize from 2d to 3d positional embeddings for object proposals in an embodied environment.
- During training and inference we add context through additional automatically detected regions.
- During training, we augment training with viewpoints from which the referent region is not visible to regularize the model against false positives.
- During inference, we explore viewpoints only up to a distance limit based on training data path lengths.
- During inference, we score region proposals in viewpoint groups with local neighborhood features to compare thousands of scores from hundreds of batches.

Figure 2 outlines our training and inference procedure.

II. Related Work

Remote object grounding is an instance of vision-and-language navigation (VLN), in which an embodied agent takes actions that move it through a simulated [8], [9] or real

environment [10] in response to a language instruction. VLN tasks span from single agent navigation to two agent task completion [11]. Here, we overview dataset and modeling efforts in the subset of VLN regarding navigation towards *object targets* based on language referring expressions.

Benchmarks for remote object grounding vary in the amount of visual information and language information given to an agent, as well as the nature of what is expected of the agent in response. In 360-VQA, for example, an agent must answer a language question by examining a panoramic scene [12]. While the agent in 360-VQA does not need to take navigation actions, it must go beyond identifying a referent object and also answer a question about that object, for example *What color is the vase to the right of the pictures?* By contrast, in Refer360° an agent adjusts its viewpoint, while still not performing navigation, in response to a *series* of language instructions localizing an object before predicting a bounding box answer [13].

Some works explore fidelity, continuous simulators [14], [15], utilize dialogue between two human players attempting to localize an agent in a scene based on object surroundings [16], or navigate to a remote object known to one player but not another [17]. We develop our bag of tricks on a *topological* simulator, where the agent only needs to reason over a discrete number of map locations, with single language instructions, rather than dialogue histories or interactions with oracles and simulated human partners.

Specifically, we focus on the Remote Embodied Visual Referring Expression in Real Indoor Environments (REVERIE) [6] and From Anywhere to Object (FAO) benchmarks [18]. We detail these in Section III.

Models for remote object grounding include: iterative, sequence-to-sequence approaches; beam search methods to score potential trajectories; and those that maintain explicit environmental memory like maps.

Sequence-to-sequence approaches take in a language instruction to initialize a decoder that takes in image observations and emits actions [19]. The History Aware Multimodal Transformer [20] is a successful sequence-to-sequence model which maintains a latent observations history used as additional conditioning information during action prediction. A similar approach maintains only a single history state passed from one timestep to the next as a single token for an action-emitting transformer [21].

Another class of models first gather rollouts from sequence-to-sequence models of *possible* trajectories to follow from the start point conditioned on the language instruction. Such beam search models are typically applied in vanilla VLN, not remote object grounding, due to VLN instructions being guiding, low-level path descriptions. VLN-BERT is one such model, which rolls out possible paths based on a language-following model [22], then feeds each candidate to a fine-tuned vision-action-language transformer model that scores how well the language instruction matches the observation sequence. Airbert follows this same setup, but performs extensive domain-relevant fine tuning through data scraped from AirBnB [3].

We first explore the environment to construct an explicit map memory with object region proposals at each location. Such mapping-based approaches have been used for vision-and-language navigation via structured scene memory [23], and for object localization in 3D environments via hierarchical mechanical search [24]. Our RREx-BoT approach combines the strength of two existing mapping-based methods: DUET [7] and OSMaN [5]. In DUET, a topological map memory is built as the agent explores, and a combination of frontier exploration and instruction following is used to find the goal location. In OSMaN, the agent is given the gold map and bounding box detections of all objects on initialization, breaking the assumptions of the benchmarks on which it is evaluated while enabling directly selecting a referent object from among all possibilities. RREx-BoT builds a map through exploration, as in DUET, but does so locally, recording object detections along the way to enable a final global scoring, as in OSMaN.

III. Task Description

Agents must navigate from a starting location and select the referent object based on region proposals at a goal location given a language referring expression. Often, there are multiple goal positions from which the referent object is visible. We consider the discrete Matterport3D [26] simulation environment used in the Remote Embodied Visual Referring Expression in Real Indoor Environments (REVERIE) task [6]. Each navigable location is represented as a node in a navigation graph. At each timestep, the agent receives a panoramic RGBD observation and a camera location and produces either a navigation action or an object targeting action to select the inferred referent object. A panorama consists of images from any requested heading or elevation.

In the standard task formulation, the agent starts with no knowledge of the environment in each episode. We consider an agent operating long-term in the same environment, a "partially-observed" version of this task, where an agent first explores the local environment up to a distance limit from the starting location and stores information before having to solve embodied referring expression tasks.

IV. Method

We face two major challenges when applying an off-the-shelf, generic vision-and-language alignment scoring model to the task of remote, embodied visual referring expression resolution. First, such VL models are trained only on 2d images and associated language, but object referents in the current task are positioned in 3d space. Second, these models output match scores between image proposals and target language in batches, where *scores are relative within-batch* with respect to other batch regions given. We cannot fit hundreds of thousands of object proposal regions in a single batch, making ranking language-image match scores between batches non-trivial.

A. Background: Vision-Language Modeling

Our approach starts with a generic vision-language model formulation, where the visual input is represented with image

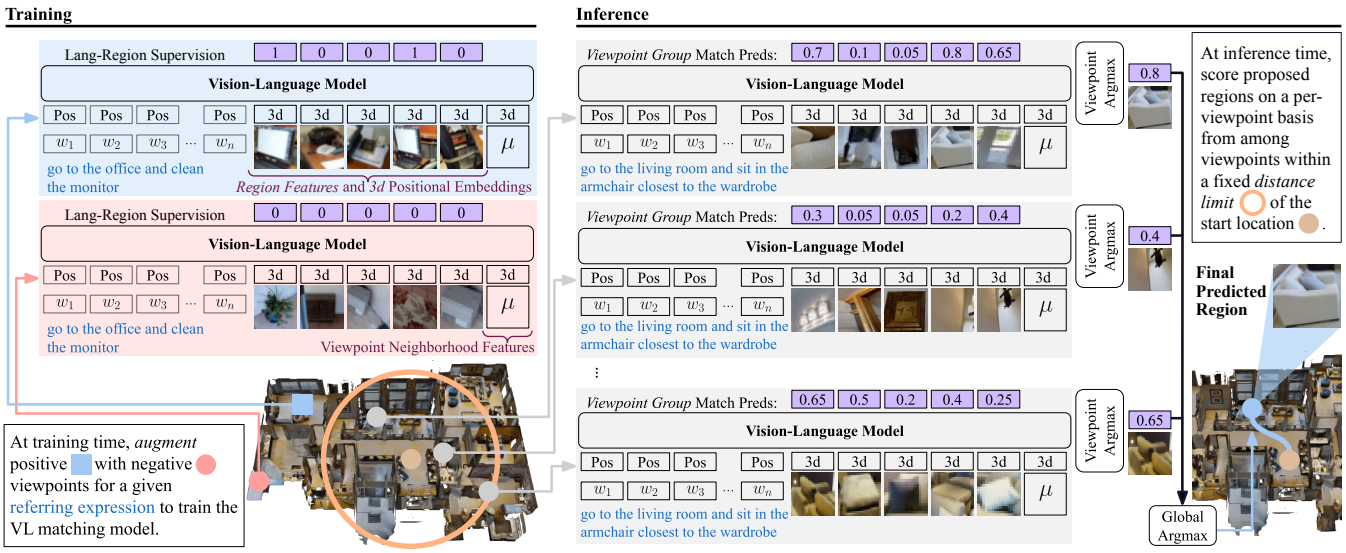

Fig. 2. We achieve state of the art performance on remote object grounding after fully exploring an environment via a generic vision-language model with minor modifications and several changes to the training setup. In particular, we start with a pretrained 2d vision-language backbone (ViLBERT [25]) and fine-tune it for remote object grounding by adding 3d positional embeddings for region proposals. We train the backbone to score the matching between a referring expression and region proposals drawn from viewpoints in 3d space where the true target object can or cannot be seen. Then, at inference time, we explore up to a fixed number of steps away from a starting viewpoint, scoring regions on a per-viewpoint basis, then pool all best-scoring regions across viewpoints to find a global best within the exploration frontier, which we predict as the referent.

region features. A region is a part of the image, such as a bounding box or object proposal, with corresponding feature and location information, typically obtained from an object detector or an object proposal generator. We focus on the task where text input $\mathbf{w}{=}w_0, \ldots, w_{T_w}$ is compared with a set of visual region features $\mathbf{v}{=}v_0, \ldots, v_{T_v}$, where T_w is the number of word tokens and T_v image tokens. To encode token positional information, each token (text or visual) is paired with a positional encoding, capturing linear order for text or bounding box size and location for image regions. This information is encoded in a latent feature and added to the corresponding token embedding. Scoring networks are pre-trained on image and text description pairs to reconstruct masked out input tokens, learning a joint distribution between vision and language. They can then be fine-tuned on tasks such as choosing a corresponding visual region to given text input [27], [25].

B. Bag-of-Tricks for Vision-Language Model in 3d

Generic vision-and-language models are trained only on 2d image inputs. There are two challenges for applying such models in large, 3d spaces: positional encodings of 3d space and an explosion in regions of interest. Figure 2 highlights our "Bag of Tricks" to overcome these challenges.

a) 3d Positional Embeddings.: To use the models in 3d, we replace the 2d bounding box coordinates (x_0, y_0, x_1, y_1) with 3d coordinates and size information (x, y, z, r), where r is the estimated radius of the object (half the diagonal of the bounding box in 3d coordinates). We normalize the (x, y, z) location relative to the viewpoint $\boldsymbol{p}$ from which the region was observed to calculate 3d positional encodings.

b) Region Scoring and Viewpoint Sampling.: Pretrained models typically identify 10-100 regions of interest [25] in a 2d image. Thus, an embodied environment can have 10,000-100,000 regions.[1] Both training and inference, by contrast, proceed with small batches; in our case, 400 regions at a time can be processed. Further, referring expressions often include information about the surrounding environment, and so region scores must be contextualized with local information. Following an implementation detail in 2d vision-language models that add an average feature over all regions as a separate region [25], we add average features for each the 100 viewpoints closest to the viewpoint group (Figure 2 *Viewpoint Neighborhood Features*).

RREx-BoT needs to be able to predict which regions are relevant per batch, including when *none are relevant*. We first sample regions from a single ground truth viewpoint—a location from which the target object is visible. A sample can include zero or more ground truth regions, depending on how many of the bounding boxes cover the target object. Since there is a large number of potential regions, and potential viewpoints, we found best to focus on regions from the ground truth viewpoint during training. We use a sigmoid loss across these regions to train the model to select correct ground-truth regions while assigning low scores to distractor regions that are physically nearby:

$$\min_{f} \mathop{\mathbb{E}}_{\mathbf{w},\mathbf{p},\mathbf{y}} \left[\sum_{v_i \in \mathbf{p}} \mathcal{L}(s_i, y_i) \right], \tag{1}$$

where $\mathbf{p} \subseteq \mathbf{v}$ is a viewpoint, i.e. a subset of visual regions, $s_i \in \mathbf{s}$ from $\mathbf{s} = f(\mathbf{p}, \mathbf{w})$ is the output sigmoid score for each region v_i from the model f, and $y_i \in \mathbf{y}$ are ground truth labels for each region v_i used with a binary cross-entropy

[1]For example, a house with 250 viewpoints and 100 regions from 4 images in 360°, i.e. $250 \times 4 \times 100 = 100{,}000$

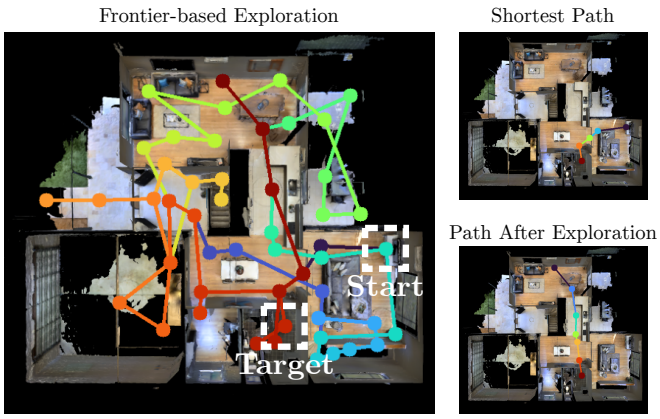

Fig. 3. During inference, RREx-BoT uses frontier-based exploration, or leverages a pre-explored map, then takes the shortest path to the target. Here, we use blue to red colors for the start to end of a trajectory including exploration.

loss $\mathcal{L}$. To better calibrate the model to score all regions during inference we add augmentation on the viewpoint level (Figure 2 *Viewpoint Augmentation*). That is, during training we either select the ground truth viewpoint, or a random *negative* viewpoint with a chance R. Random viewpoints often contain no ground truth regions of the target object, and so training with these regularize the scores for incorrect viewpoints during inference.

c) Non-Exhaustive Exploration: We start each inference by performing frontier-based exploration [28]. However, to avoid scoring potential regions from all over the house, we cap exploration to up to L steps from the starting point, with L estimated based on the training set of each task as a reasonable upper bound on how far away target objects are from starting positions based on training data. For a fair comparison with methods that do not pre-explore the house, the path the robot takes includes both the entire path taken by the frontier-based exploration, and then the estimated shortest path to the predicted destination. Figure 3 illustrates the frontier-based exploration followed by the direct path to the goal, compared to the shortest path. For each panoramic viewpoint, we extract 400 region proposals, and use depth channel information to place each region proposal in 3d space. The median depth value inside each region is used as the distance of the proposal from the camera origin.

d) Inference-time Viewpoint Grouping: To apply the trained model to 100,000 regions at inference time, we batch regions based on the viewpoint from which they were extracted. Thus, for a given inference text input $(w_0, \ldots, w_{T_w})$, we process the the visual region features $(v_0, \ldots, v_{T_v})$ from each viewpoint in separately (Figure 2 *Viewpoint Group Match Predictions*). The scores are compared across viewpoints, and the highest scoring region is selected (Figure 2 *Viewpoint Argmax* and *Global Argmax*).

V. Experiments

We demonstrate how a simple vision-language model can achieve state of the art results on the VLN benchmarks REVERIE [6] and FAO [18]. Highlighting the simplicity and usefulness, the supplementary material includes a video demonstration of this system running on a real-world TurtleBot3 platform by first pre-exploring an environment and then navigating to objects from a textual description.

A. Benchmarks and Metrics

We conduct experiments by using simulator-based datasets REVERIE [6] and FAO [18]. REVERIE contains 10,466 training instructions that contain 21 words on average, and the ground truth trajectories are 4-7 steps. Agent observations are 640x480 pixels with 60° vertical field-of-view (80° horizontal field-of-view). FAO[2] contains 26,790 training instructions that contain 47 words on average, and the ground truth trajectories are 2-21 steps (9.5 on average). We use the proposal bounding boxes provided in DUET [7] to improve the training data. As in DUET, we use observations of 600x600 pixels with 80° vertical and horizontal field-of-view, making our models directly comparable.

a) Evaluation Metrics: For navigation, we consider average path length in meters (*Trajectory Length, TL*), fraction of paths ending where the target is visible (REVERIE) or within 3 meters of it (FAO) (*Success Rate, SR*), best SR achieved along the path (*Oracle Success Rate, OSR*), and *SR* penalized by path length (*SPL*). For object grounding, we consider the episodes choosing the correct object (*Remote Grounding Succcess, RGS*), and *RGS* penalized by path length (*RGSPL*).[3] Grounding is successful if the chosen region bounding box has over 50% IoU with the ground truth object and is selected from a valid goal viewpoint.

B. Implementation Details

At a high level, RREx-BoT first visits all locations in a given scene, builds a list of proposed or observed regions of interest, and then uses that list to select the best referent region for a given referring expression. We fine-tune the ViLBERT vision-language model [25], which is pre-trained on the Contextual Captions [27] dataset, to score how well referring expressions match proposed regions. We choose ViLBERT because it is a generic VL backbone model with no task-specific architecture or objectives tuned for sequential decision making or the VLN setting.

We exclude all viewpoints that are more than L steps away from the starting viewpoint location of the agent in each episode. We set L=8 for REVERIE, the maximum length trajectory in training, and L=13 for FAO, based on a validation performance.

We use the region proposals provided by REVERIE itself for that setting, and by DUET [7] for FAO. For training, we augment these regions with new proposals predicted by Mask-RCNN [42] collected at headings 0°, 90°, 180°, 270° at 0° elevation to add additional contextual information to reach T_v=400 total. At inference we choose among only the provided proposals to select the final grounding.

[2] https://scenario-oriented-object-navigation.github.io/ also known as the SOON dataset.

[3] In FAO [18], RGSPL is referred to as SFPL; we follow DUET [7] and use the more common RGSPL name.

TABLE I. Compared to state-of-the-art methods on the REVERIE dataset, RREx-BoT outperforms all in Remote Grounding Success (RGS) and navigation success rate (SR). With pre-exploration, RREx-BoT also outperforms alternative pre-exploration methods on these metrics as well as their path-weighted variants (RGSPL, SPL). Test Unseen results are from the public leaderboard [29]. Table adapted from [7].

	Val Seen						*Val Unseen*						*Test Unseen*					
	Navigation				Grounding		Navigation				Grounding		Navigation				Grounding	
Method	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
Human	-	-	-	-	-	-	-	-	-	-	-	-	21.18	86.83	81.51	53.66	77.84	51.44
Seq2Seq[19]	12.88	35.70	29.59	24.01	18.97	14.96	11.07	8.07	4.20	2.84	2.16	1.63	10.89	6.88	3.99	3.09	2.00	1.58
RCM[30]	10.70	29.44	23.33	21.82	16.23	15.36	11.98	14.23	9.29	6.97	4.89	3.89	10.60	11.68	7.84	6.67	3.67	3.14
SMNA[31]	7.54	43.29	41.25	39.61	30.07	28.98	9.07	11.28	8.15	6.44	4.54	3.61	9.23	8.39	5.80	4.53	3.10	2.39
FAST-MATTN[32]	16.35	55.17	50.53	45.50	31.97	29.66	45.28	28.20	14.40	7.19	7.84	4.67	39.05	30.63	19.88	11.61	11.28	6.08
SCoA[33]	-	-	-	-	-	-	-	29.29	16.94	8.20	-	-	-	-	-	-	-	-
CKR[34]	12.16	61.91	57.27	53.57	39.07	-	26.26	31.44	19.14	11.84	11.45	-	22.46	30.40	22.00	14.25	11.60	-
SIA[35]	13.61	65.85	61.91	57.08	45.96	42.65	41.53	44.67	31.53	16.28	22.41	11.56	48.61	44.56	30.80	14.85	19.02	9.20
ORIST[36]	10.73	49.12	45.19	42.21	29.98	27.77	10.90	25.02	16.84	15.14	8.52	7.58	11.38	29.20	22.19	18.97	10.68	9.28
ProbES[37]	13.59	48.49	46.52	42.44	33.66	30.86	18.00	33.23	27.63	22.75	16.84	13.94	17.43	28.23	24.97	20.12	15.11	12.32
RecBERT[21]	13.44	53.90	51.79	47.96	38.23	35.61	16.78	35.02	30.67	24.90	18.77	15.27	15.86	32.91	29.61	23.99	16.50	13.51
Airbert[3]	15.16	48.98	47.01	42.34	32.75	30.01	18.71	34.51	27.89	21.88	18.23	14.18	17.91	34.20	30.28	23.61	16.83	13.28
HAMT[20]	12.79	47.65	43.29	40.19	27.20	25.18	14.08	36.84	32.95	30.20	18.92	17.28	13.62	33.41	30.40	26.67	14.88	13.08
HOP[38]	13.80	54.88	53.76	47.19	38.65	33.85	16.46	36.24	31.78	26.11	18.85	15.73	16.38	33.06	30.17	24.34	17.69	14.34
TD-STP[39]	-	-	-	-	-	-	-	39.48	34.88	27.32	21.16	16.56	-	40.26	35.89	27.51	19.88	15.40
RecBERT(O)[40]	14.40	67.81	65.50	60.12	48.28	43.95	22.20	40.24	32.58	25.90	22.01	17.52	22.91	44.44	36.13	26.95	22.08	16.75
DUET[7]	13.86	73.86	71.75	**63.94**	57.41	**51.14**	22.11	51.07	46.98	33.73	32.15	23.03	21.30	56.91	52.51	36.06	31.88	22.06
AutoVLN[4]	-	-	-	-	-	-	-	62.14	55.89	**40.85**	36.58	**26.76**	-	62.30	55.17	**38.88**	32.23	**22.68**
RREx-BoT	212.6	100.0	**77.37**	4.69	**62.97**	3.71	185.6	100.0	**61.06**	3.53	**43.74**	2.47	177.2	100.0	**65.18**	4.34	**42.07**	2.78
—Methods Utilizing Pre-Exploration—																		
OSMaN[5]	-	-	60.9	58.9	43.5	41.8	-	-	53.40	51.50	36.00	34.60	8.67	57.12	50.16	47.78	34.69	32.99
RREx-BoT(PE)	10.65	79.83	**77.30**	**75.95**	**62.97**	**61.81**	9.54	68.56	**61.01**	**58.83**	**43.68**	**42.24**	9.57	73.74	**65.19**	**62.04**	**42.05**	**40.12**

TABLE II. Compared to state of the art on the FAO dataset [18], RREx-BoT outperforms all in grounding (RGS) and navigation (SR) metrics. With pre-exploration (PE), the path weighted RGSPL and SPL similarly improve, but there are no other PE model results against which to compare for FAO. Test Unseen results are from the public leaderboard [41].

	Val Unseen						*Test Unseen*					
	Navigation				Grounding		Navigation				Grounding	
Method	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑	TL↓	OSR↑	SR↑	SPL↑	RGS↑	RGSPL↑
GBE[18]	28.96	28.54	19.52	13.34	-	1.16	27.88	21.45	12.90	9.23	-	0.45
DUET[7]	36.20	50.91	36.28	22.58	6.02	3.75	41.83	43.00	33.44	21.42	-	4.17
AutoVLN[4]	-	53.19	41.00	**30.69**	-	**4.06**	-	48.74	40.36	**27.83**	-	**5.11**
RREx-BoT	250.4	97.37	**49.17**	3.65	**11.06**	0.83	339.3	98.16	**47.53**	3.34	-	0.64
RREx-BoT(PE)	15.60	56.40	**49.17**	**48.63**	**11.06**	**11.06**	15.38	52.07	**47.48**	**47.13**	-	**9.89**

We train for 160 epochs on REVERIE and 30 epochs on FAO, using a batch size of 10. We use a learning rate of 1e-5, weight decay of 0.01, and a 10%/90% linear warmup/decay schedule. We set the viewpoint augmentation chance to R=80%, meaning 80% of training viewpoints could contain no correct referent regions (Figure 2 *Training*). Each model was trained on a single NVIDIA T4 GPU, and training took 4 days. Source code for replicating the results in this paper will be made available.

C. Benchmark Results

RREx-BoT outperforms state of the art methods on REVERIE and FAO, with and without pre-exploration, and is easily deployed to a real robot system.

a) REVERIE Results: Table I compares RREx-BoT to many others. At inference time, RREx-BoT first explores all viewpoints within L steps before scoring regions and navigating to the predicted referent. We also include a RREx-BoT variant that pre-explores (PE) the environments and so can navigate directly to the selected referent, similar to the OSMaN [5] model. The *Test Unseen* results are obtained from the public leaderboard, where our RREx-BoT is, at the time of writing, state-of-the-art in terms of *SR* and *RGS*, and *RREx-BoT Pre-Explore* is state-of-the-art in terms of *SPL* and *RGSPL* among pre-exploration models [29].

Base RREx-BoT achieves low *SPL* and *RGSPL* metrics that consider path length due to frontier exploration. The oracle success rate *OSR* is 100.0% because all locations where the referent *can be* are a part of the exploration path with L=8. *RREx-BoT (PE)* evaluations when exploration is not included as part of the path, similar to OSMaN [5]. We exceed OSMaN on path weighted *SPL* and *RGSPL* metrics, demonstrating that in cases where the agent has been operating in the same environment continuously, a simple model can significantly outperform specialized models.

b) FAO Results: Table II shows results on the unseen validation set and the held-out challenge test set of the FAO dataset [18].[4] As with REVERIE, RREx-BoT achieves state of the art results. The FAO dataset includes longer paths than REVERIE and noisier, more challenging training data, but RREx-BoT adapts to this setting and reach a convincing performance. The oracle success rate *OSR* is lower than

[4]The regular remote grounding success (RGS) metric, without path penalty, was not presented in the original FAO paper [18], or on the leaderboard. Running the model evaluation from the DUET [7] open-source code included that number however, and that is included here.

100% because our exploration budget of L=13 is lower than that needed to reach all targets. Setting L=21 would reach 100% OSR, but L=13 yields better performance because of a more tractable search space over regions.

c) Home Robot Results: We evaluate RREx-BoT on a TurtleBot3 Burger with an attached RGBD camera running ROS nodes for coordinate frames, navigation, and camera updates. The robot first drives around the environment and running Mask-RCNN on RGBD images to create regions (which have 3d coordinates from the depth). Then, we use RREx-BoT to select a region proposal from among those stored given a text query. The selected regions' 3d coordinates are passed to the ROS navigation system. The robot was able to build a map of the environment, use RREx-BoT to identify the correct object based on a query, and use the navigation system to navigate to that target. In the supplementary material we include a video demonstration of the robot performing mapping, remote grounding, and successful navigation to a referent target given speech input.

D. Performance Analysis and Ablations

Figure 4 shows correct and incorrect RREx-BoT predictions in REVERIE. To better understand where RREx-BoT does and does not work well, we analyze its performance on REVERIE in detail, including ablations of various aspects of each entry in our "bag of tricks."

a) Analysis on REVERIE: Highlighting the challenge for generalizing an image vision-language model to 100,000s of regions, we found that if we limit evaluation to only regions from a viewpoint where the goal is visible,[5] RREx-BoT can reach a 98.5% RGS on the training set. In contrast, this performance is 78.7% on the validation seen, and 62.9% on the validation unseen set.

We found that 30.8% of the errors on the unseen validation were from a valid viewpoint (versus 20.2% in DUET), and if a trajectory ended in a valid viewpoint there was a 71.6% chance it would have remote grounding success (versus 68.7% in DUET). This difference demonstrates the advantage of the global search after pre-exploration, since there are fewer failures from invalid viewpoints.

Figure 5 compares RREx-BoT with DUET [7] on the *Validation Unseen* split. We look at success as a function of: the room type of the target object; number of words in the referring expression (*Instruction Length*); shortest distance to target in meters (*Distance*); number of viewpoints where the ground truth object can be found (*Valid Viewpoints*); number of viewpoints within L=8 distance from the starting viewpoint (*Viewpoints Within Range*). We find that RREx-BoT is relatively robust to the distance and size of the search space, but could be improved with more robust language understanding for longer language inputs.

b) Ablation Studies: We consider implementation choices for each "trick" used by RREx-BoT. We examine the pre-exploration setting to study upper bound of RREx-BoT with ablated components. Table III evaluates RREx-BoT(PE): with no 3d information (– *3D Positional Encoding*), with no additional region proposals for additional context (– *Context Proposals*), with all regions considered (– *Distance Limit*), without any viewpoints that do not contain the target during training (– *Augmentation*), with regions processed in random batches instead of by viewpoint (– *Viewpoint Grouping*), and using pre-trained ViLBERT without fine-tuning on the REVERIE data (– *Fine-Tuning*).

[5]Corresponding to the training scenario without augmentation.

TABLE III. Ablations of RREx-BoT "tricks" on REVERIE.

	Val Seen		*Val Unseen*	
Method	SR↑	RGS↑	SR↑	RGS↑
RREx-BoT	**77.37**	**62.97**	**61.06**	**43.75**
– Region Positional Enc.	75.90	61.07	58.90	40.50
– Context Proposals	74.63	59.94	52.49	35.64
– Distance Limit	72.17	59.24	45.38	31.50
– Augmentation	30.15	27.62	16.39	14.14
– Viewpoint Grouping	33.59	13.77	23.69	10.17
– Fine-Tuning	10.89	2.60	12.67	4.23

TABLE IV. RREx-BoT performance on REVERIE unseen validation set as we adjust hyperparameters and ablate components within each "trick." We examine the negative viewpoint augmentation rate (Aug; 80% used by selected model), region sampling strategies (Regions), replacing our Viewpoint Relative Coordinates with different kinds of 3d positional encodings for regions (3d Enc), removing 3d context information input to ViLBERT (3dC), and changing how we group regions and limit viewpoints considered during inference (Infer).

	Method Ablation	SR↑	RGS↑
	RREx-BoT	**61.06**	**43.75**
Aug	50% Negative Viewpoints	57.71	39.48
	90% Negative Viewpoints	61.06	42.66
Regions	+ Bootstrapping	60.72	41.18
	+ Env. Dropout 50%	56.49	39.05
	+ Env. Dropout 80%	55.58	38.68
3d Enc	Start Relative Coord.	59.67	39.99
	Absolute Coord.	60.12	40.98
	BBox Coord	60.21	41.81
	No Region Positional Enc	58.90	40.50
3dC	– Context Proposals	52.49	35.64
	– Viewpoint Nbhd. Feat.	58.19	39.68
Infer	– Viewpoint Grouping	23.69	10.17
	– Distance Limit	45.38	31.50
	+ Two-Step Inference	53.80	33.77

We find that grouping by viewpoint is critical to simplify the search problem, and augmentation with viewpoints that may not contain referent regions is important to suppress false positives when scoring 100,000s of proposal regions. Since the evaluation setup for REVERIE and FAO includes a start position, and the trajectories do not typically cover the entire house, it is also important to include a budget on the search space. There are noticeable, if less dramatic, improvements when using 3d positional encoding and when adding additional in-context proposals from an object detector. We investigate each of these findings in more detail.

How to find needles in haystacks? Bridging the gap between the training setup (400 regions) and the final inference setup (100,000 regions) requires carefully managing overfitting of the model. In Table IV (*Aug*) we first vary the viewpoint augmentation and see that decreasing to 50% (from RREx-BoT's 80%) loses performance, while

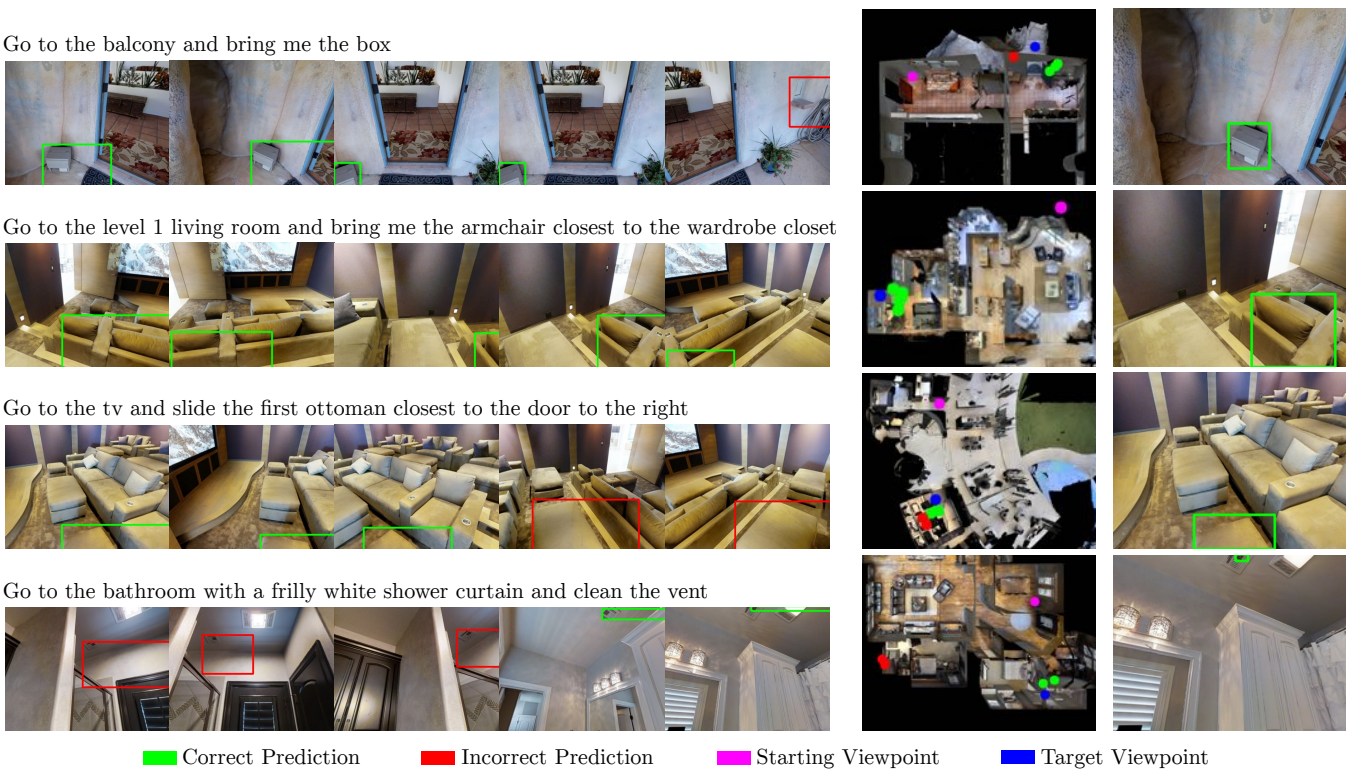

Fig. 4. Qualitative results from our model. The visualizations show a view of the ground truth target object given the text instructions, along with the top 5 predictions from the model from left to right. We also show a top-down map of the floor containing the target object with the starting location (pink), goal location (blue), correct predictions (green), incorrect predictions (red). The top 3 rows are successful remote grounding (top 1 predicted object is correct) whereas the last row is unsuccessful (top 3 predictions incorrect).

increasing to 90% has a minor negative effect. In Table IV (*Regions*), we consider sampling with replacement the regions in each viewpoint during training (+ *Bootstrapping*), and environment dropout [43] where we randomly replace regions during training with regions from the training set (+ *Env. Dropout 50/75%*), which seem to not help in this case.

How to encode 3d positions? We explore ways of encoding the 3d information. In Table IV (*3d Enc*) we show results for using the bounding box coordinates in the panorama space (*BBox Coord.*), absolute 3d coordinates (*Absolute Coord.*), 3d coordinates relative to the start of the trajectory (*Start Relative Coord.*), and constant 3d coordinates unrelated to the region (*No Region Positional Enc*). The different types of 3d information perform similarly, with *Viewpoint Rel. Coord.*, used by RREx-BoT, performing slightly better.

How to add 3d context? Investigating how to incorporate 3d context (surrounding information) when making a prediction we look at the following settings in Table IV (*3dC*): no additional object proposals (– *Context Proposals*), and no average features from neighboring viewpoints (– *Viewpoint Nbhd. Feat.*). While having additional object proposals is important, we found additional context less useful.

How to reduce the 3d search space? Finally, we consider ways of reducing the search space during inference, since 100,000 regions can create many false positives. In Table IV (*Infer*) we note that removing the *Viewpoint Grouping* and *Distance Limit* has a drastic effect on the final model. However, adding a *Two-Step Inference* approach, were we first classify among the viewpoints (average feature for each) and then between the regions in best viewpoint, reduces the search space too much, and causes a drop in performance. Interestingly, we found that including 3d positional encoding was not enough for the network to learn the "distance limit" and ignore everything beyond a certain threshold. This result is likely because a constraint of L steps in the simulator does not translate well into Euclidean distance in 3d space.

VI. Conclusion

We present RREx-BoT, a remote grounding model that builds on a generic vision-language backbone and a "bag of tricks" for training and inference for this embodied task. We set state of the art remote grounding accuracy on both REVERIE and FAO, and deploy RREx-BoT on a physical robot. We hope that these results serve as recommendations to the community for which tricks to apply when doing exhaustive vision-learning retrieval for remote grounding. Furthermore, this may help guide research for applying vision-language models to other large, geometric domains.

Acknowledgements: The authors would like to thank Steinar Þorvaldsson for supporting the hardware demo and Vicente Ordonez and the anonymous reviewers for feedback on the manuscript.

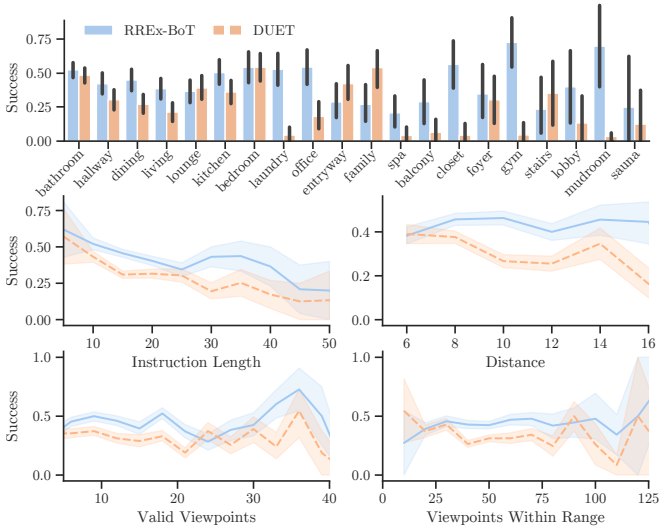

Fig. 5. Analysis of RGS (*Success*) of RREx-BoT on REVERIE unseen validation, as a function of various parameters. The shaded region is the 95% bootstrapping confidence interval of the estimate, where small sample sizes result in larger error bounds. DUET [7] is included as a baseline.

REFERENCES

[1] H. Wang, W. Wang, T. Shu, W. Liang, and J. Shen, "Active visual information gathering for vision-language navigation," in *ECCV*, 2020.

[2] A. Majumdar, A. Shrivastava, S. Lee, P. Anderson, D. Parikh, and D. Batra, "Improving vision-and-language navigation with image-text pairs from the web," in *ECCV*, 2020.

[3] P.-L. Guhur, M. Tapaswi, S. Chen, I. Laptev, and C. Schmid, "Airbert: In-domain pretraining for vision-and-language navigation," in *ICCV*, 2021.

[4] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev, "Learning from unlabeled 3d environments for vision-and-language navigation," in *ECCV*, 2022.

[5] V. Cirik, "Understanding language referring to the visual world," Ph.D. dissertation, LTI, SCS, Carnegie Mellon University, 2022.

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

[7] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev, "Think global, act local: Dual-scale graph transformer for vision-and-language navigation," in *CVPR*, 2022.

[8] M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the talk: Connecting language, knowledge, and action in route instructions," in *AAAI*, 2006.

[9] D. L. Chen and R. J. Mooney, "Learning to interpret natural language navigation instructions from observations," in *AAAI*, 2011.

[10] V. Blukis, Y. Terme, E. Niklasson, R. A. Knepper, and Y. Artzi, "Learning to map natural language instructions to physical quadcopter control using simulated flight," in *CoRL*, 2019.

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

[12] S.-H. Chou, W.-L. Chao, W.-S. Lai, M. Sun, and M.-H. Yang, "Visual question answering on 360° images," in *WACV*, 2020.

[13] V. Cirik, T. Berg-Kirkpatrick, and L.-P. Morency, "Refer360°: A referring expression recognition dataset in 360° images," in *ACL*, 2020.

[14] X. Li, D. Guo, H. Liu, and F. Sun, "REVE-CE: Remote embodied visual referring expression in continuous environment," *RAL*, 2022.

[15] J. Krantz, S. Banerjee, W. Zhu, J. J. Corso, P. Anderson, S. Lee, and J. Thomason, "Iterative vision-and-language navigation," *arXiv preprint arXiv:2210.03087*, 2022.

[16] M. Hahn, J. Krantz, D. Batra, D. Parikh, J. M. Rehg, S. Lee, and P. Anderson, "Where are you? localization from embodied dialog," in *EMNLP*, 2020.

[17] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer, "Vision-and-dialog navigation," in *CoRL*, 2019.

[18] F. Zhu, X. Liang, Y. Zhu, Q. Yu, X. Chang, and X. Liang, "SOON: Scenario oriented object navigation with graph-based exploration," in *CVPR*, 2021.

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

[20] S. Chen, P.-L. Guhur, C. Schmid, and I. Laptev, "History aware multimodal transformer for vision-and-language navigation," in *NeurIPS*, 2021.

[21] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, "VLN BERT: A recurrent vision-and-language bert for navigation," in *CVPR*, 2021.

[22] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell, "Speaker-follower models for vision-and-language navigation," in *NeurIPS*, 2018.

[23] H. Wang, W. Wang, W. Liang, C. Xiong, and J. Shen, "Structured scene memory for vision-language navigation," in *CVPR*, 2021.

[24] A. Kurenkov, R. Martín-Martín, J. Ichnowski, K. Goldberg, and S. Savarese, "Semantic and geometric modeling with neural message passing in 3d scene graphs for hierarchical mechanical search," *arXiv preprint arXiv:2008.07792*, 2020.

[25] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *NeurIPS*, 2019.

[26] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3D: Learning from RGB-D data in indoor environments," *3DV*, 2017.

[27] P. Sharma, N. Ding, S. Goodman, and R. Soricut, "Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning," in *ACL*, 2018.

[28] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *CIRA*, 1997.

[29] Y. Qi, Q. Wu, P. Anderson, X. Wang, W. Y. Wang, C. Shen, and A. van den Hengel, "REVERIE challenge," https://eval.ai/web/challenges/challenge-page/606/leaderboard/1683, accessed: 2022-10-26.

[30] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Wang, and L. Zhang, "Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation," in *CVPR*, 2019.

[31] C. Ma, J. Lu, Z. Wu, G. AlRegib, Z. Kira, R. Socher, and C. Xiong, "Self-monitoring navigation agent via auxiliary progress estimation," in *ICLR*, 2019.

[32] L. Ke, X. Li, Y. Bisk, A. Holtzman, Z. Gan, J. Liu, J. Gao, Y. Choi, and S. S. Srinivasa, "Tactical rewind: Self-correction via backtracking in vision-and-language navigation," *CVPR*, 2019.

[33] Y. Zhu, Y. Weng, F. Zhu, X. Liang, Q. Ye, Y. Lu, and J. Jiao, "Self-motivated communication agent for real-world vision-dialog navigation," in *ICCV*, 2021.

[34] C. Gao, J. Chen, S. Liu, L. Wang, Q. Zhang, and Q. Wu, "Room-and-object aware knowledge reasoning for remote embodied referring expression," *CVPR*, 2021.

[35] X. Lin, G. Li, and Y. Yu, "Scene-intuitive agent for remote embodied visual grounding," in *CVPR*, 2021.

[36] Y. Qi, Z. Pan, Y. Hong, M. Yang, A. van den Hengel, and Q. Wu, "The road to know-where: An object-and-room informed sequential bert for indoor vision-language navigation," in *ICCV*, 2021.

[37] X. Liang, F. Zhu, L. Li, H. Xu, and X. Liang, "Visual-language navigation pretraining via prompt-based environmental self-exploration," *arXiv preprint arXiv:2203.04006*, 2022.

[38] Y. Qiao, Y. Qi, Y. Hong, Z. Yu, P. Wang, and Q. Wu, "HOP: History-and-order aware pre-training for vision-and-language navigation," in *CVPR*, 2022.

[39] Y. Zhao, J. Chen, C. Gao, W. Wang, L. Yang, H. Ren, H. Xia, and S. Liu, "Target-driven structured transformer planner for vision-language navigation," in *ACM-MM*, 2022.

[40] Z. Zhan, L. Lin, and G. Tan, "Object-aware navigation for remote embodied visual referring expression," *Neurocomputing*, 2023.

[41] F. Zhu, X. Liang, Y. Zhu, Q. Yu, X. Chang, and X. Liang, "SOON challenge," https://eval.ai/web/challenges/challenge-page/1275/leaderboard/3235, accessed: 2022-12-16.

[42] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *ICCV*, 2017.

[43] H. Tan, L. Yu, and M. Bansal, "Learning to navigate unseen environments: Back translation with environmental dropout," in *ACL*, 2019.