

# Natural Language for Human-Robot Collaboration: Problems Beyond Language Grounding

Seth Pate, Wei Xu, Ziyi Yang, Maxwell Love, Siddarth Ganguri, Lawson L.S. Wong

Khoury College of Computer Sciences, Northeastern University  
{ pate.s , xu.wei3 , yang.ziyi2 , love.m , ganguri.s }@northeastern.edu , lsw@ccs.neu.edu

## Abstract

To enable robots to instruct humans in collaborations, we identify several aspects of language processing that are not commonly studied in this context. These include location, planning, and generation. We suggest evaluations for each task, offer baselines for simple methods, and close by discussing challenges and opportunities in studying language for collaboration.

## Significance

Humans generally want robots to do as they are told – this is, indeed, the second of Asimov’s laws. But if a robot can solve a problem better than we can, perhaps it should be telling us what to do.

Most language research in robotics has been on language **grounding**, in which a robot translates human instructions into actions (Anderson et al. 2018b,a; Fried et al. 2018; Majumdar et al. 2020; Li, Tan, and Bansal 2021; Wang et al. 2019). We take the reverse approach, identifying language skills with which robots may advise humans: **locating**, **planning**, and **speaking**. Developing these understudied skills will improve robot grounding, but it will also make for better robot collaborators.

In collaboration, each partner brings its own abilities. When robots are used for their strength, speed, and endurance, they need only understand language to follow instructions. But as robots get better at problem solving, they can offer us more than obedience.

For example, in industrial contexts like manufacturing and shipping, robots often operate without much language ability. But if an embodied agent could generate language, it could use its extensive memory, planning ability, and knowledge of its environment to provide its human teammate with reminders and instructions. These instructions are often helpful even if imperfect.

In this paper, we define, describe, and summarize literature on the language skills robots need to advise us. We set the collaboration problem in the context of navigation, suggest a common environment, and provide baselines with simple methods. We close with a discussion of the chal-

lenges and opportunities in creating robots that can give instructions as well as take them.

## General Problem Definition and Related Work

For sake of generality, we refer to our robot as an **agent**, and its human partner as a **user**. To give instructions, an agent needs several skills:

- **Location:** Interpret user’s **language**,  $l$ , to recognize the user’s **current state**  $s$  and **goal state**  $g$ .
- **Planning:** Find a **trajectory**  $x$  to bring the user from  $s$  to  $g$ .
- **Generation:** Draft natural language **instructions**  $w$  to express  $x$  to the user.

Ideally, the agent could repeat this cycle to collaborate in real time, as in fig. 1.

**Example Problem Context for Study** Agents can give instructions in many settings, for example cooking and equipment assembly. We chose navigation as an example context because it is an old problem with good data and benchmarks (Anderson et al. 2018b).

Moreover, navigation allows us to directly evaluate our instructions. Evaluating language is difficult because there is no single standard for good language (Zhao et al. 2021; Reiter and Belz 2009). But in navigation, we can compare instructions by giving them to a user and measuring its performance in meters and seconds. We can then use this evaluation to improve our methods.

Within navigation, we study the “vision-and-language navigation” (VLN) task, described in Anderson et al. (2018b). This task uses the Matterport3D environment, which divides real indoor settings into a graph of ‘**view-points**’ spaced about 3m apart (Chang et al. 2017). Each viewpoint has a unique identifier associated with a panoramic RGB-D image taken by a Matterport camera at that location in the environment. Anderson et al. (2018b) provide a simulator for the environment.

In this environment, VLN uses the Room-to-Room (R2R) (and similar) datasets (Anderson et al. 2018b), in which human ‘guides’ describes trajectories of paths in Matterport3D. Human ‘followers’ ground these descriptions, trying to recreate the original trajectory. The resulting dataset

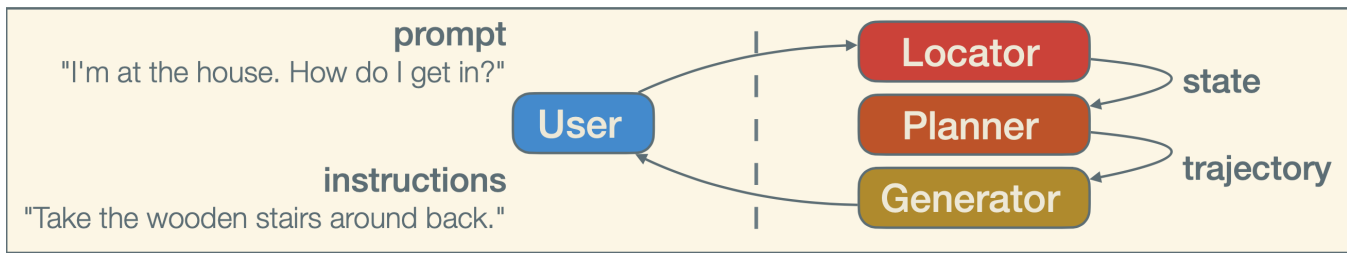


Figure 1: The collaboration problem and its tasks as a cycle.

is a collection of trajectories paired with instructions, where each trajectory is a series of viewpoints in Matterport3D,  $\mathbf{x} = (v_s, \dots, v_g)$ , and instructions are series of words,  $\mathbf{w} = (w_0, \dots, w_n)$ .

We can now define each language skill in context, summarizing the current literature and opportunities for new research.

**Location** The user begins at a **start viewpoint**  $v_s$ , describing this and its **goal viewpoint**  $v_g$  with language  $l$  to the **locator**. The locator must infer  $\hat{v}_s$  and  $\hat{v}_g$  from  $l$ .

Studying a similar problem, Tse and Campbell (2018) use a bag-of-words model and a Bayes filter to track a belief distribution of the agent’s location. This is a fast and simple method, but it cannot generalize to unseen environments, because it relies on language occurring alongside locations during training to estimate likelihoods. Moreover, the bag-of-words approach discards useful information about word order. Banerjee, Thomason, and Corso (2020) created a dataset for a navigation task involving location and grounding. They give a benchmark for the grounding task, but do not address location. This illustrates the gap we identify in this paper – language grounding is well studied at the cost of other skills.

**Planning** Given the locator’s guesses of  $\hat{v}_s$  and  $\hat{v}_g$ , the **planner** finds a feasible path between the estimated start-goal pair, a **trajectory**  $\mathbf{x} = (\hat{v}_s, \dots, \hat{v}_g)$ .

In navigation, we have excellent search strategies, so it is tempting to treat this problem as solved. In our baselines, we use A\* search. A more sophisticated planner, however, would take its user into account. An optimal path for one user might be a poor choice for another. The user of a wheelchair should not be told to take the stairs, for example. The planner could also consider how well it can describe a path, choosing a simple suboptimal path rather than risk confusion with a shorter one.

We might term this ‘user-aware planning’. This is an understudied concept, at least within navigation, and we highlight it as an opportunity for growth.

**Generation** The **generator** writes **instructions**  $w$  to describe  $\mathbf{x}$ . In Matterport3D, each viewpoint  $v_i$  is associated with a panoramic image  $p_i$ , which can be used for generation.

Because the language is conditioned on a trajectory, we cannot directly use unconditional probabilistic language models such as GPT-3 (Brown et al. 2020). Rather, our problem is more similar to another conditional generation task, image and video captioning (You et al. 2016). But unlike captions, instructions must come in order and might be quite detailed.

To solve this problem, a common method is to fill templates, such as ‘Turn [blank] at the [blank]’. Daniele, Bansal, and Walter (2017) used templates in a 3-D navigation task, chunking trajectories into series of ‘Compound Action Specifications’. These specifications reduce instructions to a pattern of orientation (which way to turn) and distance (how far to walk). Their templates were effective, but the method required engineering features by hand for that environment. Templates are also inflexible; an instruction might need details like ‘Remember that people in Britain drive on the other side.’

For the VLN task, Zhao et al. (2021) gives an excellent summary of generation in this context. They give standards for instructions and evaluate several generators, including their own template generator model, Crafty. They found generators like the sequence-to-sequence Speaker-Follower (Fried et al. 2018) to be relatively successful. However, VLN researchers mainly use their generators to bootstrap synthetic data and train better grounding models. This is a great application of a generator, but so far there is less interest in generating instructions for human users: there are many grounding models for VLN, but few generators, and those that do exist are far from human ability (VLN-Leaderboard 2018; Zhao et al. 2021).

## Data and Standards for Evaluation

We suggest some simple means to evaluate an agent’s ability to give instructions. Where possible, we use data and metrics from the VLN community. For the generation task, we use the unmodified R2R dataset, and test on its **validation seen** and **unseen** splits (Anderson et al. 2018b).

The R2R dataset cannot be used directly to train a locating agent, because it does not associate language to locations. Instead, we started with the Room-Across-Room (RxR) (Ku et al. 2020), which is similar but includes pose traces and times for each word  $w_i$  in  $w$ . We split up each instruction, matching each phrase of  $w_0 \dots w_j$  to its closest graph location  $v$ , to make our language dataset.

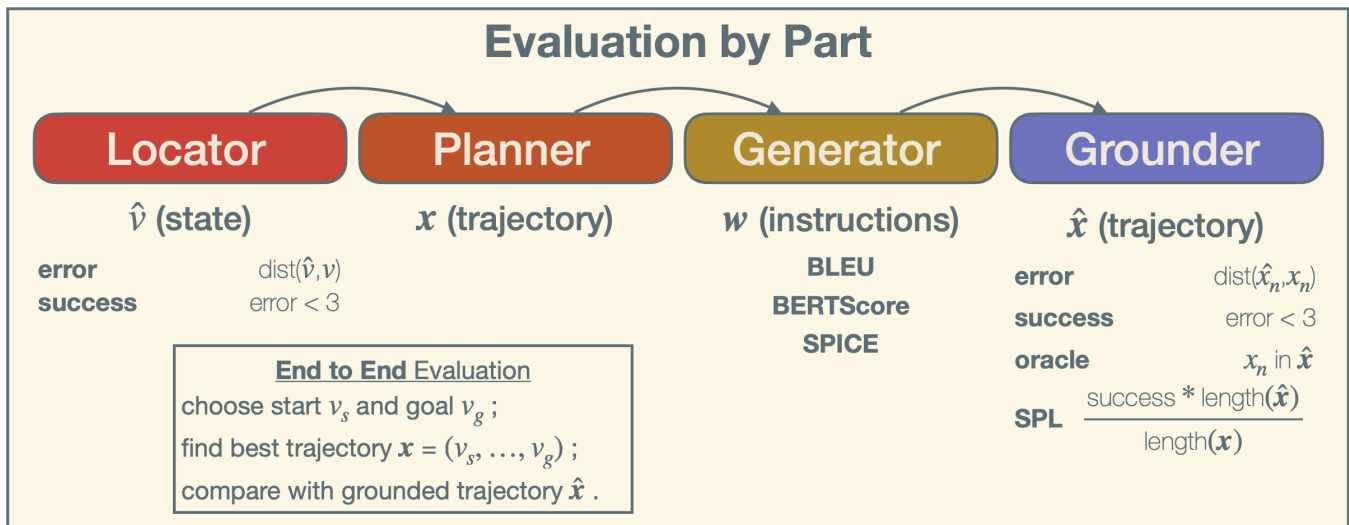


Figure 2: Notation and metrics for evaluating tasks individually and as a pipeline.

**Location Evaluation** We evaluate the locator on its **error**, the graph distance within the Matterport simulator between its viewpoint  $\hat{v}$  and the target viewpoint  $v$ . We also report **success rate**; the task is a success if the error falls within a 3m threshold, a convention for this dataset (Anderson et al. 2018a).

**Planning Evaluation** In the sparse graph of the Matterport environment, there tends to be one clearly best plan for any two points, so evaluating the planner made little sense here. In other environments, we could evaluate planned trajectories by length, elevation change, etc. We could also test ‘user-aware’ planners by giving them two arbitrary locations and running their resulting plan through a fixed generator and grounder. Different planners would elicit different performance from the generator and grounder, giving us a basis for comparison.

**Generation Evaluation** Generators are usually evaluated by comparing their output to reference translations using metrics like BLEU (Papineni et al. 2002) and BERTScore (Zhang et al. 2020). There are also models like SPICE (Anderson et al. 2016), which evaluate image captions. These measures were not meant to evaluate instructions *per se*, and are often of limited value (Zhao et al. 2021).

Zhao et al. (2021) developed a neural network ‘compatibility model’ to assign instructions and trajectories a similarity score, which correlated well with human evaluation. We do not use their model here; instead, we evaluate our generator directly on how well the user grounds its instructions, producing **grounded trajectory**  $\hat{x} = (\hat{x}_0, \dots, \hat{x}_n)$ .

As in location, we compare the final position of the grounded trajectory  $\hat{x}_n$  with the final position of the actual trajectory  $x_n$  to find **error**. In addition to **success rate**, there is the more generous **oracle success rate**, measuring whether the agent **ever** encountered the goal. Finally, there

is success weighted by path length, **SPL**, which penalizes agents for taking a longer path than necessary (Anderson et al. 2018a).

The R2R dataset includes human descriptions of trajectories  $x$ ; we denote these ground-truth descriptions as  $\underline{x}$ . By using a fixed model to ground first those human instructions, then our generator’s description of the same  $x$ , we get a sense of how close our generator comes to human ability. We use a single pretrained grounding model, the Speaker-Follower model (Fried et al. 2018), as the fixed grounder. It is no longer state of the art, but is easily implemented, and good enough for a reasonable comparison of a generator. A better evaluation would include an ensemble of different grounding models, as well as a human evaluation on the simulator.

**End-to-End Evaluation** As location, planning, and generation are parts of a single process, we suggest an end-to-end evaluation as depicted in fig. 2. This allows us to test an agent’s performance in a closed loop, ideally with a human user. We tested our baseline models end-to-end as shown, but they performed worse than random agents, so we omit the results table here.

## Baseline Results and Discussion

We report empirical results on location and generation using some baseline models, highlighting the large performance gap between existing models and ideal.

### Location

- **Bag of Words:** Empirically estimates  $p(v_i|l)$  from word frequency in the training text, then greedily chooses from among all viewpoints. By design, cannot generalize to unseen viewpoints.

Table 1: Baseline results for generation. Navigation metrics obtained by grounding generated language using a fixed grounding model (see "Data and Standards for Evaluation"). Human-level success rates are  $\approx 86\%$ .

	Error (m) ↓		Success (%) ↑		Oracle Succ. (%) ↑		SPL (%) ↑	
	seen	unseen	seen	unseen	seen	unseen	seen	unseen
Random Grounder	9.66	9.46	17.05	19.41	22.35	22.47	16.07	16.92
Human Instructions	3.33	6.64	66.70	35.20	74.05	44.70	60.69	28.13
Seq2Seq	3.58	6.68	64.11	33.84	71.76	44.40	59.18	27.04
Reinforce (BLEU)	<b>3.42</b>	6.61	<b>65.00</b>	32.31	<b>72.35</b>	44.32	<b>59.24</b>	25.80
Reinforce (BERT)	3.69	6.62	64.68	35.52	69.71	44.70	58.42	28.23
Reinforce (Agent)	3.87	<b>6.38</b>	59.90	<b>36.44</b>	69.41	<b>47.13</b>	54.90	<b>29.18</b>

- **RNN/ResNet:** Encodes image  $p_i$  at viewpoint  $v_i$  with a ResNet (He et al. 2016) trained on ImageNet (Deng et al. 2009), producing  $h_v$ . Encodes word embeddings of  $l$  with an LSTM (Hochreiter and Schmidhuber 1997) to make a context vector  $h_w$ . Trains a deep neural network to produce  $y = f(h_v, h_w)$ , choosing the highest  $y$ .
- **Stacked Cross Attention:** SCAN (Lee et al. 2018) uses attention to align parts of the viewpoint image  $p$  with individual words  $l \in l$ , giving an overall similarity score. Chooses the most similar viewpoint.

Table 2: Baseline results for location. Success rates are too low to locate users using language.

	Error (m) ↓		Success (%) ↑	
	seen	unseen	seen	unseen
Random	9.58	8.31	6.56	8.00
Bag of Words	29.14	-	<b>8.08</b>	-
RNN/ResNet	8.81	8.13	19.85	20.71
SCAN	8.06	7.93	<b>29.82</b>	<b>27.34</b>

## Generation

- **Sequence to Sequence:** A sequence-to-sequence LSTM with attention (Fried et al. 2018). Minimizes cross-entropy loss of output tokens,  $w$  against target tokens,  $\underline{w}$ .
- **REINFORCE:** Starting with the pretrained generator above, uses the REINFORCE method (Williams 1992) to backpropagate an undifferentiable loss to the model. We test two losses based on the standard language metrics BLEU and BERTScore. A third loss (Agent) explores the idea of training a generator directly to produce better instructions. We use a pretrained grounding model to take actions in the simulator according to the generated instructions, and return loss equal to the distance between the goal location  $v_g$  and the agent's final location  $\hat{x}_n$ .

**Discussion** Our baseline results show that there is ample room for improvement across tasks. We close with a few points to summarize and extend our work.

To train our locator, we had to do surgery on our language dataset. Since the language skills we describe are understudied in robotics, we must augment existing data, as well as collect new data.

Although we spent little time on planning, it is part of the collaborative process and should be studied alongside it. In planning, the instruction agent is not only choosing the best path for navigation, it is choosing the best path to tell the user. Capturing user outcomes in a cost function will help us train better planners.

Finally, we note that our tasks of location, planning, and generation are closely tied to grounding. Generators can make better training data for grounding, but grounding is helpful in training generators, by providing an objective evaluation mechanism for scoring generated language.

This is good reason to build robots that understand both sides of the communication process, opening new opportunities for human-robot collaboration.

## References

- Anderson, P.; Chang, A.; Chaplot, D. S.; Dosovitskiy, A.; Gupta, S.; Koltun, V.; Kosecka, J.; Malik, J.; Mottaghi, R.; Savva, M.; and Zamir, A. R. 2018a. On evaluation of embodied navigation agents. *arXiv:1807.06757*.
- Anderson, P.; Fernando, B.; Johnson, M.; and Gould, S. 2016. SPICE: Semantic propositional image caption evaluation. In *European Conference on Computer Vision*.
- Anderson, P.; Wu, Q.; Teney, D.; Bruce, J.; Johnson, M.; Sünderhauf, N.; Reid, I.; Gould, S.; and Van Den Hengel, A. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Banerjee, S.; Thomason, J.; and Corso, J. J. 2020. The RobotSlang Benchmark: Dialog-guided Robot Localization and Navigation. In *Conference on Robot Learning*.
- Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *Neural Information Processing Systems*.
- Chang, A.; Dai, A.; Funkhouser, T.; Halber, M.; Niessner, M.; Savva, M.; Song, S.; Zeng, A.; and Zhang, Y. 2017.

- Matterport3D: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision*.
- Daniele, A. F.; Bansal, M.; and Walter, M. R. 2017. Navigational instruction generation as inverse reinforcement learning with neural machine translation. In *ACM/IEEE International Conference on Human-Robot Interaction*.
- Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Fried, D.; Hu, R.; Cirik, V.; Rohrbach, A.; Andreas, J.; Morency, L.-P.; Berg-Kirkpatrick, T.; Saenko, K.; Klein, D.; and Darrell, T. 2018. Speaker-follower models for vision-and-language navigation. In *Neural Information Processing Systems*.
- He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.
- Ku, A.; Anderson, P.; Patel, R.; Ie, E.; and Baldrige, J. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Empirical Methods in Natural Language Processing*.
- Lee, K.-H.; Chen, X.; Hua, G.; Hu, H.; and He, X. 2018. Stacked cross attention for image-text matching. In *European Conference on Computer Vision*.
- Li, J.; Tan, H.; and Bansal, M. 2021. Improving Cross-Modal Alignment in Vision Language Navigation via Syntactic Information. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Majumdar, A.; Shrivastava, A.; Lee, S.; Anderson, P.; Parikh, D.; and Batra, D. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *European Conference on Computer Vision*.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: a method for automatic evaluation of machine translation. In *Annual meeting of the Association for Computational Linguistics*.
- Reiter, E.; and Belz, A. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics* 35(4): 529–558.
- Tse, R.; and Campbell, M. 2018. Human–Robot Communications of Probabilistic Beliefs via a Dirichlet Process Mixture of Statements. *IEEE Transactions on Robotics* 34(5): 1280–1298.
- VLN-Leaderboard. 2018. Vision and Language Navigation Leaderboard. <https://eval.ai/web/challenges/challenge-page/97/leaderboard/270>. Accessed: 2020-07-11.
- Wang, X.; Huang, Q.; Celikyilmaz, A.; Gao, J.; Shen, D.; Wang, Y.-F.; Wang, W. Y.; and Zhang, L. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.
- Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3): 229–256.
- You, Q.; Jin, H.; Wang, Z.; Fang, C.; and Luo, J. 2016. Image captioning with semantic attention. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K. Q.; and Artzi, Y. 2020. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.
- Zhao, M.; Anderson, P.; Jain, V.; Wang, S.; Ku, A.; Baldrige, J.; and Ie, E. 2021. On the evaluation of vision-and-language navigation instructions. In *Conference of the European Chapter of the Association for Computational Linguistics*.