2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Grounding Language to Landmarks in Arbitrary Outdoor Environments

Matthew Berg*, Deniz Bayazit*, Rebecca Mathew, Ariel Rotter-Aboyoun, Ellie Pavlick, Stefanie Tellex[1]

***Abstract*— Robots operating in outdoor, urban environments need the ability to follow complex natural language commands which refer to never-before-seen landmarks. Existing approaches to this problem are limited because they require training a language model for the landmarks of a particular environment before a robot can understand commands referring to those landmarks. To generalize to new environments outside of the training set, we present a framework that parses references to landmarks, then assesses semantic similarities between the referring expression and landmarks in a predefined semantic map of the world, and ultimately translates natural language commands to motion plans for a drone. This framework allows the robot to ground natural language phrases to landmarks in a map when both the referring expressions to landmarks and the landmarks themselves have not been seen during training. We test our framework with a 14-person user evaluation demonstrating an end-to-end accuracy of 76.19% in an unseen environment. Subjective measures show that users find our system to have high performance and low workload. These results demonstrate our approach enables untrained users to control a robot in large unseen outdoor environments with unconstrained natural language.**

I. INTRODUCTION

As autonomous systems improve on outdoor robots, such as self-driving vehicles and drones, it becomes increasingly necessary to develop models that translate high-level, often ambiguous instructions to low-level inputs for the autonomous system. For example, a passenger might instruct a self-driving vehicle to *"Avoid the red bridge on the way to the office"* or to *"Go through the red bridge before heading to CVS."* Such natural language commands present multiple structural and semantic layers that the robot's autonomous system cannot understand.

Existing approaches to this translation problem assume a language model trained over a map of the exact environment in which the robot will be deployed [1, 2, 3, 4]. This lack of generality prevents the robot from navigating to areas on the map where the language model has not been trained. In addition, current approaches require grounding all of the natural language to a predefined, fixed set of possible predicates, which is overly strict and limits generalization. Such approaches also focus towards training a language model on a limited vocabulary that is specific to a given map, forgoing the highly developed semantic depth of publicly available global mapping data. This limitation curbs the user's ability to refer to landmarks by using semantic descriptors, like *"red bridge"* or *"ice cream store."*

Fig. 1: **Simulated Skydio R1 in Tulsa, Oklahoma.** This map was not shown during training and the model succeeds at performing 76.19% of the tested natural language commands in this environment.

In this paper, we present a system that allows a person to command a drone with natural language in an environment never-before-seen to the drone. The system is constructed from a language model and planning model. In the language model, natural language is parsed into a structured logical form necessary for planning. We use Linear Temporal Logic (LTL), which represents atomic propositions over a linear timeline. We exclusively use the LTL atoms for our logical form, allowing the natural language to stay in its unstructured state, such as *"Go to the big blue bear but avoid the main green"* grounding to $\mathtt{F}(\textit{big blue bear} \wedge \neg \textit{main green})$. Keeping natural language in the logical form allows us to leverage more flexible neural models better suited to resolving ambiguous language while simultaneously maintaining a structured command representation in the planner. Critically, this retention of natural language reduces the predefined predicates our system requires to logical operators (e.g. AND, NOT). As a result, our model can seamlessly handle unseen referring expressions to landmarks, allowing it to generalize to entirely novel environments and commands.

In the planning model, the grounded LTL formulae are supplied to a planner that has access to a predefined semantic map of the robot's environment, generated from OpenStreetMap (OSM) [5]. The landmarks names from the LTL formulae are resolved to navigational coordinates. These coordinates become part of a motion plan that is uploaded to a simulated Skydio R1 drone.

We perform both a user evaluation and corpus-based evaluation of this model. Our in-person user evaluation

* These authors contributed equally to this work.
[1] Brown University, Providence, RI, 02912, USA. Emails: {matthew_berg, deniz_bayazit, rebecca_mathew, ariel_rotter-aboyoun, ellie_pavlick}@brown.edu, stefie10@cs.brown.edu

demonstrates an accuracy of 76.19% in an environment not shown during training and a mean NASA-Task Load Index (NASA-TLX) performance score of 14.85 points out of 20 points. For the corpus-based approach, we present 1540 challenging natural language commands collected on Amazon Mechanical Turk (AMT) which describe trajectories containing one or two landmarks from 22 unique maps[1]. Using this data, we show an accuracy of 45.91%.

II. RELATED WORK

Natural language presents an intuitive means of communication with robots, particularly those with autonomy systems that rely on higher-level human guidance. There has been extensive work on developing models which translate natural language to lower-level input for these autonomy systems. Previous work has focused on grounding the complete natural language command into a symbolic form for the motion planner [6, 7, 8, 9]. To handle complex instructions, Tellex et al. [1] created a probabilistic graphical framework for grounding natural language commands to landmarks and other entities in a map. In addition, neural sequence-to-sequence (Seq2Seq) models that ground natural language to symbolic forms have been proposed [4, 10, 11]. However, these approaches make the dual assumption that there exists a small number of landmarks in the map and that the language model can be trained on these landmarks directly. In contrast, our approach uses a map with millions of landmarks and does not assume that a language model can be trained on all of them.

Importantly, natural language can refer to entities not only via explicit names, but also via general descriptions. For example, one might say *"Go to the medicine store"* instead of *"Go to CVS."* There exists a body of work on grounding semantic information in natural language to logical forms [2, 12, 13, 14, 15]. To create more domain-independent groundings, Cheng et al. [12] demonstrates a neural semantic parser that uses an intermediate form containing natural language. Misra et al. [13] presents a framework for grounding novel verbs to logical forms by leveraging available information in the environment. Similar to these works, we use natural language in a logical form and leverage information in the map to ground unknown words. However, our approach includes natural language in the fully grounded logical form, and leverages semantic utterances with information in the map to ground novel landmarks. This combination allows us to interpret references to landmarks that the robot's model has never seen during training while also grounding complex commands with constraints and subgoals.

A variety of approaches exist for combining natural language with robot instruction following in a map with landmarks [16, 17, 18, 19]. Dzifcak et al. [18] presents a framework for grounding natural language commands into a logical form representing goals and actions, while Kollar et al. [17] directly parses the natural language command into a logical form of figure (subject of sentence), verb, landmark, and spatial relation. Our work is positioned between the two, coupling a goal-based logical form with landmarks directly parsed from the natural language command.

[1] https://github.com/h2r/Language-to-Landmarks-Data

III. APPROACH

Our system allows a person to command a drone with natural language in a never-before-seen environment. The system can interpret natural language commands, including references to nearby landmarks, with no training data for the environment. A graphical representation of our system is shown in Figure 2.

The language model grounds natural language commands to LTL formulae. The LTL structure is created by CopyNet [20], a Seq2Seq model capable of copying out of vocabulary (OOV) words. To ground natural language landmark referring expressions to landmarks in a map unseen to the language model, we use a resolution model that draws semantic information from a mapping database. The final output of the language model is an LTL formula with natural language in the logical form, e.g. $\texttt{F}\,(\textit{CVS} \wedge \texttt{F}\,(\textit{red bridge}))$.

The LTL formula is then passed to the planning model. The planning model uses a map generated from OSM, partitioned into Voronoi cells [21]. The partitioned map along with the LTL formula are supplied to the AP-MDP planner [4]. This planner extracts goals and constraints from the LTL formula to create a motion plan as a series of latitude and longitude points.

A. Linear Temporal Logic

As our language model is not constrained to any map region or landmarks, it is necessary to encode goals and constraints of the natural language command in a domain-independent way. To accomplish this, we turn to LTL, a domain-independent formalism whose syntax can encode goals and constraints of the robot's path. By allowing for encoding of both the present and future states of the robot, LTL supports the inherent non-Markovian nature of unconstrained natural language commands, such as *"Move to the medicine store without going over the red bridge."* We use LTL to determine if a discrete trajectory satisfies the goals and constraints of the natural language command. LTL has the following grammatical syntax:

$$\phi := p \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \mathcal{G}\phi \mid \mathcal{F}\phi \mid \phi\mathcal{U}\psi \mid \mathcal{N}\phi$$

where $p \in \mathcal{P}$ is an atomic proposition, ϕ and ψ are LTL formulae, $\neg$, $\wedge$, and $\vee$ denote logical "not," "and," and "or," $\mathcal{G}$ denotes "globally," $\mathcal{F}$ denotes "finally," $\mathcal{U}$ denotes "until," and $\mathcal{N}$ denotes "next." Semantic interpretations of these operations are included in Manna and Pnueli [22]. For example, a command such as *"Go to the big blue bear but avoid the main green"* would have an LTL expression of $\texttt{F}\,(\textit{big blue bear} \wedge \neg\textit{main green})$.

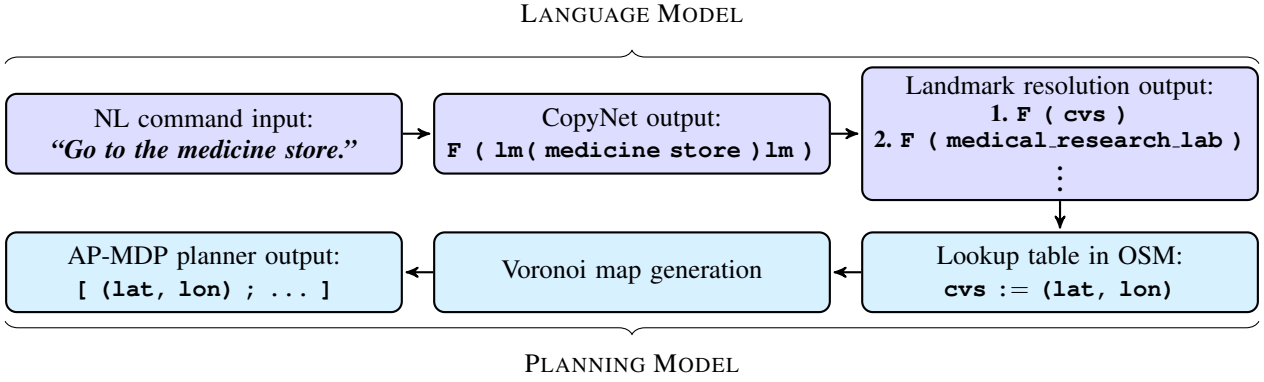

Fig. 2: **End-to-End System Pipeline.** Natural language is given to the language model, which returns a grounded LTL formula. The planning model then creates a motion plan which satisfies the LTL formula.

B. CopyNet

To translate natural language commands into logical forms, current approaches use a Seq2Seq model [4, 10, 11]. Seq2Seq models learn how to translate input sequences into output sequences. However, existing Seq2Seq models learn a mapping from a fixed input language to a fixed output language, and require all symbols in the output language to have appeared at training time. In contrast, our language model generalizes to any region, and thus needs the ability to understand words and commands the language model has not been trained on. In particular, it is essential that we extract unseen landmark referring expressions from the natural language command. For example, given the command *"Go to the medicine store"* our model needs to correctly identify that *"medicine store"* is the referring expression and the corresponding LTL formula would be `F (`*medicine store*`)`.

We approach this challenge with CopyNet [20], which is developed for cases when the output contains many subsequences from the input. CopyNet introduces a copy-attention mechanism atop the traditional Seq2Seq framework [23]. This copy mechanism is fundamental to our language model, allowing for a more domain-general model even with a small training set.

When comparing CopyNet to a purely generative recurrent neural network with the LCSTS dataset [24], Gu et al. demonstrates that CopyNet improves production of readable output for out-of-vocabulary (OOV) words. We selected CopyNet because it was accessible in multiple open-source implementations. We use Adam Klezcweski's implementation of CopyNet[2] with the addition of pre-trained GloVe embedding vectors [25]. We use mjc92's dataset[3] to validate Klezcweski's model.

To train our model, we use a corpus of 668 natural language navigation instructions collected by Oh et al. [4] Each command has a corresponding LTL formula, making this dataset well-suited for training a Seq2Seq model like CopyNet. We augment the data by replacing goal locations with Brown campus landmark names scraped from OSM. We then divide these landmarks into unique datasets containing landmarks from north campus and south campus. In addition, we wrap references to landmarks with `lm(` and `)lm` as shown in step two of Fig. 2, simplifying extraction of landmark referring expressions for the landmark resolution model. Finally, we limit the dataset to the following three LTL structures:

$$\mathcal{F}(\phi) \mid \mathcal{F}(\phi \wedge \mathcal{F}(\psi)) \mid \mathcal{F}(\phi \wedge \neg\psi)$$

C. Landmark Resolution Model

1) Mapping Database: A key focus of our framework is the language model that grounds language to landmarks, as humans find landmarks important for navigation instructions, particularly for unfamiliar environments [26]. Landmarks are geographic objects important to human spatial cognition [27]. Following previous work [28, 29] we use OSM as our landmark database.

OSM is a global open-source map where any user can add landmarks and information about the landmarks. Critically, this information can be semantic in nature, such as the type of cuisine for a restaurant or the function of a building. We leverage OSM's extensive semantic database as the foundation of our language model, enabling groundings of semantic referring expressions to landmarks.

Two building blocks of the OSM database are NODES and WAYS. NODES are points with a latitude, longitude, and unique numerical ID. NODES commonly represent landmarks such as statues, benches, and trees. WAYS are lists of NODES, commonly representing larger landmarks like buildings, roads, and greens. Closed WAYS have a polygon geometry. Both NODES and WAYS can be tagged with key-value pairs about their appearances, functions, or other semantic information.

2) Landmark Resolution Model: Given all the possible landmark candidates in the map, the model needs to resolve the user's referring expression to the correct landmark. The landmark resolution model finds the maximally probable candidate by calculating the similarities between the referring expression and each landmark's semantic information.

[2]https://github.com/adamklec/copynet
[3]https://github.com/mjc92/CopyNet

The landmark resolution model receives the CopyNet output of an LTL formula with the user's referring expression. While any arbitrary model could resolve this expression given textual descriptions, images, or robot sensor data, we present a model that uses word embeddings to resolve the user's referring expression to the landmark name.

The model uses the database's semantic information about each landmark to find the intended landmark. However, the user's referring expression may not lexically align with the landmark database. For example, we would expect "store" and "shop" to have similar meaning, even if OSM's data model only supports `key:shop`. To resolve these lexical conflicts, we use word embeddings, which represent words or phrases as vectors in a high-dimensional vector space [25, 30, 31, 32, 33]. High-dimensionality allows us to use cosine similarity (the cosine of the angle between vectors) to compare semantic referring expressions.

A referring expression may fall into one or more of three possible categories: name, address, and general description. An example of a command using more than one category would be *"Fly to CVS pharmacy,"* which includes name and description.

name: Our model exclusively uses the OSM key `name`.

address: Our model exclusively uses `addr:housenumber` and `addr:street`.

descriptions: Our model uses keys we observed to be semantically significant in natural language commands, such as `amenity`, `shop`, and `leisure`.

For each category we gather the key values into lists. Then, to handle multiple categories of values, we create all possible combinations of these lists. For each combination, we compute the average of their word vectors. We then calculate the cosine distance between each of these averaged vectors and the phrase vector for the referring expression. Finally, we use the minimum cosine distance to identify the referred landmark. We evaluate this approach against other models in Section IV-B.2. The cosine distance between two vectors is defined as the difference between 1 and their cosine similarity.

D. Voronoi Maps and Planning

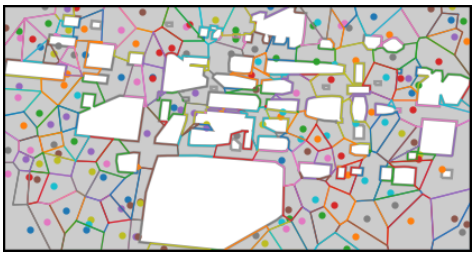

Fig. 3: **Map partitioned into Voronoi cells.** White holes represent regions containing landmarks.

We use the AP-MDP planner to convert grounded LTL formulae to high-level motion plans, and leave lower-level motion planning to the drone's autonomy system. Oh et al. [4] partitions a hard-coded map into a grid of flyable zones and target landmarks. However, since other real-world geometries can be large and complex, a more flexible approach to map partitioning is required.

Our approach uses Voronoi cells [21]. We query OSM for landmarks in a 300 meter radius square around a center point, creating holes for each WAY polygon and five meter radius square holes around each NODE. Then, we randomly generate points inside the solid region, which are used to partition the map into Voronoi cells as shown in Fig. 3. We have observed the Voronoi cells can enable faster planning over large distances. When comparing our results in the predefined map by Oh et al. [4], Voronoi-based planning between two landmarks 48.28 meters apart ran in 37.08 ± 6.43 seconds, whereas the grid-based approach ran in 90.49 ± 0.27 seconds (over three runs).

Further, the AP-MDP planner understands landmarks as a single latitude and longitude coordinate, not a polygon. As such, we represent WAYS in the planner by choosing one corner NODE as its representative point.

To align with limitations of both natural language and our framework, we filter certain landmarks. Landmarks need to be named for the purposes of natural language commands, so they must have a `key:name`. We exclude any landmark containing the key `highway`, `railway`, `place`, `boundary`, or `waterway`, because it is difficult to use a singular representative point for very large landmarks.

IV. EVALUATION

We test that our system accurately grounds natural language commands with references to landmarks, without being trained on those landmarks. We conduct an end-to-end user evaluation where participants give natural language commands to the drone and observe the robot's actions in simulation. In addition, we perform a corpus-based evaluation on a diverse set of maps to test the limits of our framework. Finally, we demonstrate the system acting in a real outdoor domain[4].

A. User Evaluation

To test end-to-end performance on a map unseen to the language model during training, we ran an in-person user evaluation with 14 voluntary student participants. Each student gave three spoken natural language commands to our system and evaluated the resulting behavior of a Skydio R1 drone in a simulated outdoor map of Tulsa, Oklahoma.

The simulator is built in Unity [34], using outdoor environments generated with the Mapbox SDK [35] (Fig. 1). Using ROS and ROS# [36, 37], the simulator and planner communicate about the drone's flight status and flight trajectories. The simulator allows the participant to view the trajectory the drone takes given the participant's natural language command.

As shown in Table I, our model accurately grounds natural language commands to LTL and formed correct motion plans

[4] `https://youtu.be/a-JGems7fzs`

for 76.19% of user commands. In this table, we also break down failure cases. We observe challenges with two forms of natural language commands: commands that include spatial language, such as *"Go to l_1 near l_2"*; or commands with verbs or unexpectedly long phrases that CopyNet has not been sufficiently trained on. Spatial language phrases cause CopyNet to not copy enough words, resulting in improper groundings or improper LTL structures. We hypothesize that CopyNet failures are due to the limited use of spatial language in CopyNet's training dataset, and that a more representative training dataset would address these problems. Also, planner errors were due to an indexing bug that we resolved post-evaluation.

After using our system, users answered the NASA-TLX questionnaire to measure workload on a scale of 0 to 20 (least to most) [38]. On average, users reported high performance and low workload (Table II). Additionally, we use the Systems Usability Scale (SUS) [39] to understand system ease of use. We report a mean SUS score of 76.25 with standard deviation of 18.39, which is above the average SUS score of 68 [40].

	Percentage (%)
Speech-to-text errors	4.76
Incorrect grounding (Landmark Resolution)	2.38
Planner errors	4.76
Improper LTL (CopyNet)	11.90
Succeeded	**76.19**

TABLE I: System performance accuracy for in-person user evaluation

	Raw NASA-TLX (pts)
Performance	14.85 ± 05.38
Mental demand	03.50 ± 02.42
Physical demand	02.83 ± 04.49
Temporal demand	01.50 ± 01.50
Effort	03.40 ± 03.17
Frustration	05.50 ± 05.08

TABLE II: Raw NASA-TLX scores on a 20 point scale

B. Component Evaluation

We analyze the performance of individual components of our language pipeline to understand failure modes and potential improvements to our end-to-end system.

1) CopyNet Evaluation: We trained two models to evaluate CopyNet. The first is trained on natural language commands with a single landmark, the second on natural language commands with two landmarks. We trained with a learning rate of 0.001 over 8 epochs for the single landmark model and 15 epochs for the two landmark model. The models were then evaluated against phrases with seen and unseen landmarks as shown in Table III. For two landmark commands, we observe on average that CopyNet grounds 69.18% of commands containing one unseen landmark and 53.49% of commands containing two unseen landmarks to the correct LTL structure (Table III). CopyNet errors are principally attributed to not copying enough words from input to output.

Number of Landmarks	Seen (%)	1 Seen, 1 Unseen (%)	Unseen (%)
One	100.00 ± 0.00	N/A	74.50 ± 2.88
Two	99.48 ± 0.20	69.18 ± 2.52	53.49 ± 2.95

TABLE III: CopyNet accuracy

2) Landmark Resolution Evaluation: We compare our landmark resolution model to other models, as shown in Table IV. We create the following baselines to evaluate the effectiveness of our landmark resolution model. The Name model represents a landmark by just its name phrase vector, an average of word embedding vectors for every word in its name. The Uniform model represents a landmark by assigning equal weight to every OSM semantic feature (including the name of the landmark) and averages their phrase vectors. The term frequency-inverse document frequency (tf-idf) [41] model weighs each semantic feature's phrase vector with its tf-idf score, a metric to downweigh frequent or uninformative words by document, where each map is a document. All models use minimum cosine distance to identify the referred landmark.

Landmark names often contain proper nouns, which may be OOV. We evaluate if using morphological information (e.g. prefixes, suffixes, roots, etc.) helps the model process OOV words by comparing fastText [31, 33], which uses such information, to larger word embedding models like Word2Vec and GloVe [25, 30].

We evaluate on 129 references collected from seven researchers in the Brown University Humans to Robots Lab. We showed each person OSM landmark information from a single map and asked for different landmark referring expressions by type(s): name, address, and description.

We define the grounding accuracy to be the percentage of landmarks returned by our language model that matches the intended reference. We calculate grounding accuracy and mean reciprocal rank (MRR) of every landmark resolution model and word embedding combination. MRR is defined as the average of the reciprocal rank scores across multiple queries. The reciprocal rank score of a query (a user's semantic reference) is the multiplicative inverse of the correct landmark's ranking. For example, if the landmark resolution model ranks the true landmark corresponding to a user's semantic reference as third-most likely, the reciprocal rank would be $1/3$ (assuming the list is three landmarks long).

Table IV shows that our landmark resolution model performs best with GloVe, which we attribute to its large vocabulary.

C. Corpus-Based Evaluation

We test our language model's ability to both identify the appropriate LTL structure and properly extract landmarks

		Name	Uniform	tf-idf	Our Model
Accuracy (%)	fastText	41.86	43.41	51.94	**58.14**
	Word2Vec	42.64	45.74	54.26	**58.91**
	Glove840B	44.96	48.06	55.81	**68.99**
MRR (%)	fastText	47.72	61.73	49.15	**66.84**
	Word2Vec	51.22	63.51	54.38	**68.97**
	Glove840B	51.39	65.22	54.38	**76.35**

TABLE IV: Landmark grounding accuracy and MRR results for different landmark resolution and word embedding models

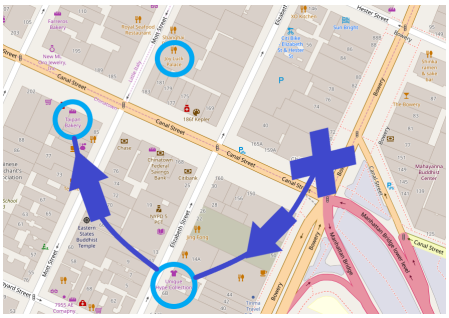

Fig. 4: **AMT trajectory example.** An OSM region with trajectory that corresponds to `F(lm(`l_1`)lm & F(lm(`l_2`)lm))`

City Name	Number of Landmarks	Accuracy (%)
Jacksonville #2	16	17.14
Boston	39	20.00
New York #1	71	30.00
Chicago #2	26	35.71
Charlotte #1	24	35.71
Seattle	119	37.14
Denver #1	27	40.00
Philadelphia #1	21	44.29
Indianapolis	10	45.71
Denver #2	21	45.71
Jacksonville #1	19	47.14
Los Angeles #1	60	48.57
Los Angeles #2	62	52.86
Columbus #2	26	52.86
Chicago #1	22	54.29
Houston	32	54.29
New York #2	73	54.29
Philadelphia #2	90	55.71
San Diego #1	41	55.71
San Diego #2	31	55.71
Charlotte #2	15	57.14
Columbus #1	10	70.00
Average	**38.86**	**45.91 $\pm$ 12.70**

TABLE V: Corpus-based language pipeline accuracy

from unseen commands by collecting a test set of challenging natural language commands from AMT. We collected commands for 22 urban American regions. (Table V).

AMT workers viewed a screenshot of a region in OSM with an overlaid trajectory (Fig. 4). Trajectories allow us to ask AMT workers for natural language commands without extensive language prompting. At the start of each task, AMT workers saw an example map and related example commands. We further provided a detailed task description to ensure AMT workers responded with high-level commands, not low-level, action-oriented instructions. Every AMT worker was given semantic information about each landmark to allow for flexibility in landmark referring expressions. We provided Google search cards without the landmark's address as to not bias the AMT workers with OSM semantic data. We have published 1540 collected commands, each formed by a unique AMT worker. Compensation was $0.50 per task.

We achieve a 45.91% mean accuracy of grounding natural language to correct fully-formed LTL. Some inaccuracies in the corpus-based evaluation may be due to unclear AMT instructions, which would lead to incorrect AMT worker annotations.

V. CONCLUSIONS

We present a framework for grounding complex, unseen natural language commands to motion plans for a robot operating in outdoor environments. For a 14-participant user evaluation, our system showed a 76.19% end-to-end accuracy and a mean NASA-Task Load Index (TLX) performance score of 14.85 out of 20 points. In addition, we demonstrate a mean accuracy of 45.91% for resolving a challenging corpus of natural language referring expressions to previously-unseen landmarks. We further present an improved planning model for Linear Temproal Logic (LTL) expressions over large and complex geometries. Last, we provide the collected corpus of 1540 natural language to LTL trajectory commands.

Future work can focus either on improving components of our framework, such as improving the copying mechanism or adding a vision module to our landmark resolution model. We can also direct work towards expanding the model's reach beyond navigation with OpenStreetMap. Search and rescue operations require responders to refer to dynamic entities, like people or cars, which are not listed in most maps. Incorporating a probabilistic spatial distribution could account for referring to these dynamic landmarks (e.g. *"Find the car behind the building"*). Finally, the user's location could be used to resolve ambiguity between multiple suitable landmark candidates.

ACKNOWLEDGMENTS

This work is supported by the National Aeronautics and Space Administration under grant number NNX16AR61G, the US Army under grant number W911NF1920145, and with support from the Sloan Foundation. The authors would like to thank Nakul Gopalan and Thao Nguyen for their feedback and support.

REFERENCES

[1] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy, "Approaching the symbol grounding problem with probabilistic graphical models," *AI Magazine*, vol. 32, no. 4, p. 64, 2011.

[2] Y. Artzi and L. Zettlemoyer, "Weakly supervised learning of semantic parsers for mapping instructions

to actions," *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 49–62, 2013. [Online]. Available: https://www.aclweb.org/anthology/Q13-1005

[3] R. Paul, A. Barbu, S. Felshin, B. Katz, and N. Roy, "Temporal grounding graphs for language understanding with accrued visual-language context," *ArXiv*, vol. abs/1811.06966, 2018.

[4] Y. Oh, R. Patel, T. Nguyen, B. Huang, E. Pavlick, and S. Tellex, "Planning with state abstractions for non-markovian task specifications," in *Robotics: Science and Systems*, 2019.

[5] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[6] T. Kollar, S. Tellex, D. Roy, and N. Roy, "Grounding verbs of motion in natural language commands to robots," *Experimental Robotics Springer Tracts in Advanced Robotics*, p. 31–47, 2014.

[7] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy, "Understanding natural langugage commands for robotic navigation and mobile manipulation," in *National Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence, 2011.

[8] A. S. Huang, S. Tellex, A. Bachrach, T. Kollar, D. Roy, and N. Roy, "Natural language command of an autonomous micro-air vehicle," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2663–2669.

[9] C. Matuszek, D. Fox, and K. Koscher, "Following directions using statistical machine translation," in *Proceedings of the 5th ACM/IEEE International Conference on Human-robot Interaction*, ser. HRI '10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 251–258. [Online]. Available: http://dl.acm.org/citation.cfm?id=1734454.1734552

[10] N. Gopalan, D. Arumugam, L. L. S. Wong, and S. Tellex, "Sequence-to-sequence language grounding of non-markovian task specifications," in *Robotics: Science and Systems*, 2018.

[11] L. Dong and M. Lapata, "Language to logical form with neural attention," *CoRR*, vol. abs/1601.01280, 2016. [Online]. Available: http://arxiv.org/abs/1601.01280

[12] J. Cheng, S. Reddy, V. Saraswat, and M. Lapata, "Learning structured natural language representations for semantic parsing," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 44–55. [Online]. Available: https://www.aclweb.org/anthology/P17-1005

[13] D. K. Misra, K. Tao, P. Liang, and A. Saxena, "Environment-driven lexicon induction for high-level instructions," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 992–1002. [Online]. Available: https://www.aclweb.org/anthology/P15-1096

[14] M. Damonte, R. Goel, and T. Chung, "Practical semantic parsing for spoken language understanding," *CoRR*, vol. abs/1903.04521, 2019. [Online]. Available: http://arxiv.org/abs/1903.04521

[15] P. Yin, C. Zhou, J. He, and G. Neubig, "Structvae: Tree-structured latent variable models for semi-supervised semantic parsing," *CoRR*, vol. abs/1806.07832, 2018. [Online]. Available: http://arxiv.org/abs/1806.07832

[16] M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the talk: Connecting language, knowledge, and action in route instructions," *The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference*, vol. 2, no. 6, p. 4, 2006.

[17] T. Kollar, S. Tellex, D. Roy, and N. Roy, "Toward understanding natural language directions," in *HRI 2010*, 2010.

[18] J. Dzifcak, M. J. Scheutz, C. Baral, and P. W. Schermerhorn, "What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution," *2009 IEEE International Conference on Robotics and Automation*, pp. 4163–4168, 2009.

[19] J. Andreas and D. Klein, "Alignment-based compositional semantics for instruction following," *CoRR*, vol. abs/1508.06491, 2015. [Online]. Available: http://arxiv.org/abs/1508.06491

[20] J. Gu, Z. Lu, H. Li, and V. O. K. Li, "Incorporating copying mechanism in sequence-to-sequence learning," *ArXiv*, vol. abs/1603.06393, 2016.

[21] G. Voronoi, "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les parallélloèdres primitifs." *Journal für die reine und angewandte Mathematik*, vol. 134, pp. 198–287, 1908.

[22] Z. Manna and A. Pnueli, *The Temporal Logic of Reactive and Concurrent Systems*. Berlin, Heidelberg: Springer-Verlag, 1992.

[23] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014.

[24] B. Hu, Q. Chen, and F. Zhu, "Lcsts: A large scale chinese short text summarization dataset," in *EMNLP*, 2015.

[25] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://www.aclweb.org/anthology/D14-1162

[26] K. L. Lovelace, M. Hegarty, and D. R. Montello, "Elements of good route directions in familiar and

unfamiliar environments," in *International conference on spatial information theory*. Springer, 1999, pp. 65–82.

[27] K.-F. Richter and S. Winter, *Introduction: What Landmarks Are, and Why They Are Important*. Springer International Publishing, April 2014, pp. 1–25.

[28] A. Rousell, S. Hahmann, M. Bakillah, and A. Mobasheri, "Extraction of landmarks from openstreetmap for use in navigational instructions," in *Association of Geographic Information Laboratories in Europe*, 2015.

[29] M. Drager and A. Koller, "Generation of landmark-based navigation instructions from open-source data," in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 2012.

[30] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. USA: Curran Associates Inc., 2013, pp. 3111–3119. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999792.2999959

[31] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[32] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, April 2017, pp. 427–431.

[33] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning word vectors for 157 languages," in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[34] Unity Technologies, "Unity." [Online]. Available: https://unity.com/

[35] Mapbox, "Mapbox unity sdk." [Online]. Available: https://github.com/mapbox/mapbox-unity-sdk

[36] M. Quigley, J. Faust, T. Foote, and J. Leibs, "ROS: An open-source robot operating system," in *IEEE International Conference on Robotics and Automation Workshop on Open Source Software*, 2009.

[37] Siemens, "*ROS#*," 2017, https://github.com/siemens/ros-sharp, [Accessed: 2018].

[38] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Human Mental Workload*, ser. Advances in Psychology, P. A. Hancock and N. Meshkati, Eds. North-Holland, 1988, vol. 52, pp. 139 – 183. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166411508623869

[39] J. Brooke, "SUS-a quick and dirty usability scale," *Usability Evaluation in Industry*, vol. 189, no. 194, pp. 4–7, 1996.

[40] J. Sauro, "Sustisfied? little-known system usability scale facts user experience magazine," 2011. [Online]. Available: https://uxpamagazine.org/sustified/

[41] C. Sammut and G. I. Webb, Eds., *TF–IDF*. Boston, MA: Springer US, 2010, pp. 986–987. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_832