BROWN UNIVERSITY

UNDERGRADUATE HONORS THESIS

Generalizing Natural Language Instruction Following to Aerial Robots and Arbitrary Environments

---

Author:
Deniz BAYAZIT

Advisor and First Reader:
Dr. Stefanie TELLEX
Second Reader:
Dr. Ellie PAVLICK

A thesis submitted in fulfillment of the requirements for Honors in Computer Science

in the

Humans to Robots Laboratory
Department of Computer Science

May 22, 2020

BROWN UNIVERSITY

Abstract

Generalizing Natural Language Instruction Following to Aerial Robots and Arbitrary Environments

by Deniz BAYAZIT

Humans convey instructions and observations to each other via natural language. They use their knowledge of the environment as well as the language rules to process such information. Reproducing a similar ability in a robot would be extremely useful for untrained users who do not have an in depth knowledge of robot programming, but who would like to interact and collaborate in similar ways with their autonomous systems.

However, the type of language given to robots depends on (1) their varying affordances and (2) the varying environments. For the first case, we notice that a vast majority of related work on processing natural language instructions are for mobile ground robots. For example, unmanned aerial vehicles (UAVs) have more degrees of freedom that the instructions can refer to. For aerial robots, combining other modalities like eye-gaze and hand gestures with natural language is promising to generalize landmark specification in fully observable indoor environments. On the other hand, for larger outdoor spaces such MR specification is not enough as landmarks change from map to map. Most models require pre-training a language model on each map to understand commands referring to landmarks in the map.

For the first variance, we present an interface that uses natural language grounding using an MR interface to solve high-level task and navigational instructions given to an autonomous drone. To generalize to new environments outside of the training set in outdoor environments, we present a framework that parses references to landmarks, then assesses semantic similarities between the referring expression and landmarks. Ultimately, our system translates natural language commands involving arbitrary landmarks to trajectory plans for a drone. To evaluate these approaches, in each chapter we use corpus based evaluations, robot experiments, as well as video demonstrations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

To ensure accessibility to users who may not know robot programming, we need to deploy language models that can account for instruction variances. Human to robot communication shows several factors that can ensure re-usability of trained language models. We hypothesize that these factors include (1) the robot's affordances and (2) the environment in which it plans. Joining distinct modalities such as eye-gaze and hand gesture with natural language promise improved communication with robots that have different configuration spaces. On the other hand, for larger outdoor spaces such gestural specifications are not enough and landmarks change from map to map. Most models require training a language model on a specific map before it can understand commands referring to those landmarks. In this thesis, we focus on approaches accounting for these two variances.

In the second chapter, we present a system that uses natural language grounding with an MR interface to solve high-level task and navigational instructions given to an autonomous drone. Given a map, our interface first grounds natural language commands to reward specifications within a Markov Decision Process (MDP) framework. Then, it passes the reward specification to an MDP solver. Finally, the drone performs the desired operations in the real world while planning and localizing itself. Our approach uses MR to interact with a set of known virtual landmarks, enabling the drone to understand commands referring to objects without being equipped with object detectors for multiple novel objects or a predefined environment model. We find that users are able to command drones more quickly via both MR interfaces (with and without language) when compared to the web interface, with roughly equal system usability scores across all three interfaces.

In the third chapter, to generalize to new environments outside of the training set, we present a framework that parses references to landmarks, 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 that our approach enables untrained users to control a robot in large unseen outdoor environments with unconstrained natural language.

In the final chapter, we discuss future work that can alleviate re-training language models across different robots and environments.

Chapter 2

Using Natural Language and Mixed Reality to Control a Drone

FIGURE 2.1: An example of a task-oriented navigational command given to a drone through the MR interface with language.

This work was published in ICRA 2019 as "Flight, Camera, Action! Using Natural Language and Mixed Reality to Control a Drone" and is a joint work with Baichuan Huang, Daniel Ullman, Nakul Gopalan and Stefanie Tellex.

2.1 Introduction

As robots become increasingly autonomous, it is imperative that designers create intuitive and flexible ways for untrained users to interact with these systems. A natural language interface is immediately accessible to non-technical users and does not require the user to use a touchscreen or radio control (RC). A natural language interface can flexibly interpret the user's desires without requiring that a novice user become proficient in a specialized system interface. After a user specifies their goal using language, the robot can understand these instructions and engage in autonomous planning to follow the instructions while avoiding obstacles using off-the-shelf planners.

Command line and programming APIs are traditional interfaces used to control a robot, but they require the human user to have expertise in using a complex system interface. The current state of the art in commercial drone interfaces is a tablet or smartphone interface, or an RC controller (Skydio, 2018; DJI, 2018). Current natural language interfaces require a predefined model of the environment including landmarks (Tellex et al., 2011b; Karamcheti et al., 2017), which is difficult for a drone to obtain. For example, given the instruction "Fly around the wall to the chair and take a picture," the drone must already have a model of the wall and the chair to infer a policy. More recent approaches tackle this problem using Mixed Reality (MR) technology, powered by products like the HoloLens (Microsoft, 2018) to control a drone in hidden areas with gaze and gesture (Erat et al., 2018). However, to the best of our knowledge, MR has not been used to give high-level language commands to a drone.

We address these interaction problems by using natural language within Mixed Reality Head-mounted Displays (MR-HMDs) to provide an intuitive high-level interface for controlling a drone using goal-based planning. By using MR, a user can annotate landmarks with natural language in the drone's frame of reference. The process starts with the MR interface displaying the virtual environment of a room that can be adjusted to overlay 3D meshes on physical reality from the perspective of the user and be interacted with using gestures. Afterward, the user can annotate landmarks with colored boxes using the MR interface. Then, when the user sends a command, we use the I-DRAGGN framework (Karamcheti et al., 2017; Arumugam et al., 2018) to translate this natural language text to a reward function in the Markov Decision Process (MDP) domain. Finally, the drone built with a Raspberry Pi, called PiDrone Brand et al., 2018, generates a trajectory using planning, while also localizing itself to determine its current position. The systems we use during this process are the HoloLens for the MR interface, a base station for language mapping and MDP solving, and Robot Operating System (ROS) (Quigley et al., 2009) for communication with the drone.

To train and evaluate the language understanding system, we collected data using Amazon Mechanical Turk (MTurk). After recording simulation videos with AirSim Shah et al., 2017, we asked the workers to give a possible natural language command that would result in the execution of the observed behavior. We collected two datasets, one for action-oriented tasks which require the workers to give primitive instructions. The other is for goal-oriented tasks, which requires the workers to give higher-level descriptions. Our trained model obtains high accuracies for both action-oriented (94%) and goal-oriented commands (95%).

Further, we conducted an exploratory user study to compare the low-level 2D interface with two MR interfaces. Overall, we found that our MR system offers a user-friendly approach to control a drone with both low-level and high-level instructions. The language interface does not require direct, continuous commands from the user. Instead, users give an initial language command to the drone as shown in Fig. 2.1, and then it executes that command autonomously. Overall, the MR system does not require the user to spend as much time controlling the drone as with the 2D interface.

2.2 Related Work

Natural language is the primary mode of communication for humans, with the additional communicative help of gesture and gaze. This makes natural language an

obvious approach to controlling a drone. The question of how to effectively translate between natural language instructions and robot behavior has been widely studied in previous work Zettlemoyer and Collins, 2005; MacMahon, Stankiewicz, and Kuipers, 2006a; Kress-Gazit, Fainekos, and Pappas, 2008; Chung et al., 2015; MacGlashan et al., 2015; Mei, Bansal, and Walter, 2016; Arumugam et al., 2017. Some early work on converting from natural language instructions to robot behavior was conducted by mapping natural language to a formal logical goal description and action language Dzifcak et al., 2009a. Some methods provide models (such as MARCO MacMahon, Stankiewicz, and Kuipers, 2006a and DCG Howard, Tellex, and Roy, 2014) which connect natural language phrases to physical objects, actions, and environments. Huang et al. (2010) presented a natural language interface to a drone, but required a complete semantic map of the environment (including landmarks) in advance. To fit the robot into the stochastic environment, converting natural language to reward function in MDP has been proposed by MacGlashan et al. MacGlashan et al., 2015. Additionally, Karamcheti et al. Karamcheti et al., 2017; Arumugam et al., 2018 introduced I-DRAGGN framework, which is used in this paper to convert human language to drone behavior. All of these previous studies have required an a priori model of landmarks in the environment. By contrast, our approach with MR does not require an a priori model of landmarks, but instead users can specify virtual landmarks whose groundings are known by the language model. This interface enables the landmark objects to be specified in the drone's global frame such that it can interpret commands without a complete model of the environment.

Most previous user interfaces require users to control drones via RC controllers. This type of control typically requires sufficient skill and experience to proficiently operate a drone, which is a notable barrier to use for novice, untrained users (Peschel and Murphy, 2013). These specialized interfaces are not necessarily intuitive for inexperienced users, as they are low-level forms of control. Commercial drones like Skydio (Skydio, 2018) and DJI (DJI, 2018) use phone apps to control the drone. However, as a drone has 6 degrees of freedom, a 2D interface is not the most intuitive way to command it.

Collaboration between humans and robots using MR is a promising alternative to direct control via RC or phone apps. MR provides a more intuitive, user-friendly visualization than a 2D visualization tool such as Rviz Kam et al., 2015. Using MR to control an arm or a drone is facilitated by the use of gesture and gazes Erat et al., 2018; Rosen et al., 2017. Rosen et al. (2017) presented an MR interface to inform the user of a robot's intent. Herrmann and Schmidt (2018) also proposed a gesture-based interface and speech-based interface for teleoperating a drone via MR.

Sibirtseva et al. (2018) used a combination of natural language in an MR environment for reference resolution in a human-robot pair task. However, this system uses a Wizard-of-Oz approach to interpret the language, with a human-in-the-loop to provide groundings between natural language and object attribute tokens. By contrast, our approach allows the drone to process and execute a user's commands fully autonomously.

2.3 Approach

We consider the problem of drone navigation in an environment with objects and obstacles. Our system allows a human operator to give low-level instructions like "move forward three squares," or goal-oriented commands such as "go to the chair

and take a picture." The drone then interprets these instructions and follows the commands in the environment. Our system combines existing modules, such that the cognitive demands placed on human users are relatively small. Our contribution is the design of the overall interface which combines language grounding, planning, MR, and robotics, together with an evaluation of the system's performance.

2.3.1 PiDrone

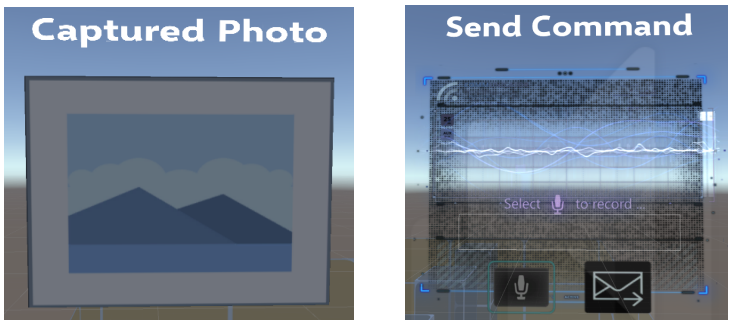
We required an autonomous drone that can localize itself in an environment. We chose the PiDrone as our robotic platform because it is a open-source system that is fully customizable Brand et al., 2018. The drone is equipped with one downward-facing camera. We also implemented localization with a particle filter (Monte Carlo localization) (Thrun, Burgard, and Fox, 2005). The drone flies over a highly-textured planar surface, and we use the OpenCV library Bradski, 2000 to extract and detect Oriented FAST and Rotated BRIEF (ORB) features. Each frame from the camera, along with the altitude value from the infrared sensor, are used to compute the bearing and the distance from the last frame and to update the weights of particles Thrun, Burgard, and Fox, 2005. The general goal is to keep track of the drone's position constantly, and to match features from the current frame with features from the current estimated position from the map to update the location of the drone based on the matched features. The drone will keep sending the current position to the Mixed Reality system and the base station via ROS. Although feature-based localization might not be as precise as the OptiTrack motion tracking system, it simulates the uncertainty of the real environment, and can be changed to ORB-SLAM Mur-Artal, Montiel, and Tardós, 2015 with a high-performance drone in a complex environment. For faster planning and language grounding to specific discrete areas of the environment, we chose to discretize our environment by creating a grid of cells where each cell is $50 \times 50 \times 30$ centimeters (width, length, height).

2.3.2 Mixed Reality Interface

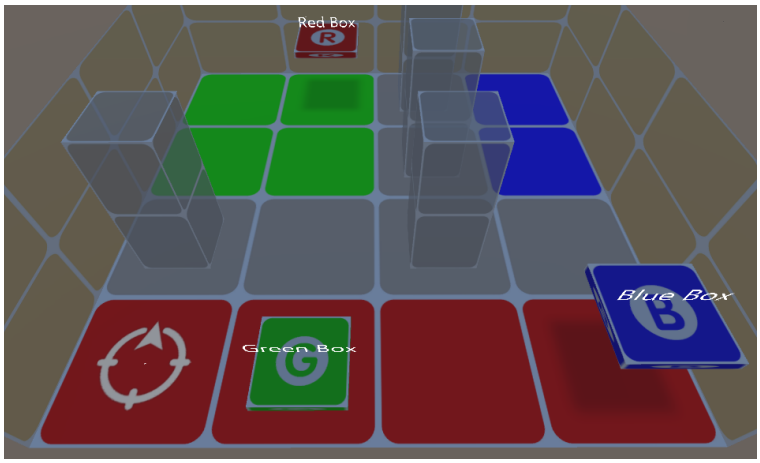
We use natural language, gaze, and gesture to control a drone through Mixed Reality. The position of the drone is provided by a localization method, and passed to the base station through ROS. The physical world coordinates are then translated into the grid-based coordinates of the MR world. This allows us build a virtual grid model in Unity 3D and deploy it on the HoloLens.

We use the spatial mapping from the HoloLens to map the virtual grid model to the actual textured map. A manual calibration process is used to align the drone's coordinate frame with the HoloLens's frame. As studied in Hoenig et al. (2015), adding a connection between virtual and physical world is helpful for the user to perceive the environment. The user can also place or remove a virtual landmark at the location they are looking at by voice command or by a tap gesture in the air. It can also be dragged from one place to the other with gesture. The landmark facilitates communication, as the user is now able to instruct the drone to navigate to a specific position by saying "go to the landmark" rather than giving an explicit instruction.

As shown in Fig. 2.3b, we created a push-to-talk language input user interface (UI) in Unity 3D, which is adapted from the HoloLens library. We used push-to-talk because the drone makes significant noise in flight, which mistakenly triggers voice activity detection in the built-in speech recognition system. The user records their command with the UI and sends the natural language command to the drone. We use Google's



(A)



(B)

FIGURE 2.2: The UI in Unity 3D

(A) Left UI shows the photo captured by the drone. Right UI shows the voice input (seen by the user as in Fig. 2.1).

(B) The environment in MR.

Speech API Google, 2018 to convert speech to text. Additionally, we have a feedback UI which shows that the drone has completed a photo-taking goal-oriented task by displaying it when it is taken. The HoloLens will keep publishing the position of the landmark and the natural language through ROS. We use ROS-Sharp Siemens, 2017 to connect Unity 3D to ROS.

2.3.3 Markov Decision Process

We specify an MDP model to represent the drone's environment and actions, which helps in planning the robot's behavior. The MDP is a five-tuple of $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$. The variable $\mathcal{S}$ is the state spaces of environment, $\mathcal{A}$ is the action spaces of the drone, $\mathcal{T}$ denotes the state transition probabilities, $\mathcal{R}$ defines the reward function for the drone to enter in a specific state, and γ is the limit of the horizon of the planner Bellman, 1957. We use the simple-rl Abel, 2017 as the MDP solver to produce a policy which maps states to actions. The goal is to maximize the cumulative expected discounted reward. As the policy depends on the reward function and the initial state that is passed into the MDP solver, we can change the location of the virtual landmarks and still follow the command. We use the simple-rl library Abel, 2017 to create and solve an MDP domain.

The domain we use is shown in Fig. 2.3b. This model is adapted from the Cleanup Domain which is introduced by MacGlashan et al. (2015). The environment contains a "box", an "obstacle", and a "room". The "box" represents the virtual landmark. We assigned the attributes "color" and "position" to objects, so that the user can send meaningful tasks according to the environment. We use a propositional space of reward functions to represent goal-oriented natural language commands. For example, a command such as "take a picture of the green box" translates to the propositional function **photoInDrone boxColor**.

2.3.4 Language Model

The I-DRAGGN framework (Karamcheti et al., 2017; Arumugam et al., 2018) that we chose to employ is a hybrid task-grounding language model that takes in natural language commands and returns the corresponding reward functions via recurrent neural network methods. We preferred I-DRAGGN to other frameworks as it covered both action-oriented and goal-oriented tasks and because it showed better accuracy compared to other models listed in Karamcheti et al. (Karamcheti et al., 2017). We used PyTorch 0.4.0 Paszke et al., 2017 to complete this deep learning task. The reward functions are broken down into a callable unit and an argument as described in Karamcheti et al. Karamcheti et al., 2017. A callable unit is akin to a function that has arguments. This leads to improved generalization in the generation of reward functions, as the agent is capable of generating unseen function argument pairs (the combination of callable unit and argument does not appear in training data). For example, for an action-oriented command such as "Go backwards 5 spaces," a callable unit would be `back` and the argument would be `5`. Similarly, for a goal-oriented command such as "Take a photo of the blue box and move to the green room," a callable unit would be `photoInDrone_agentInRoom` and the argument would be `blue_green`.

Before collecting data, we picked out callable units and argument possibilities. In total we identified seven suitable action callable units (six directions of movement and take a photo) and three suitable goal callable units (take a photo of box, go to a colored room, and the combination of them). With the dimensions of the initial

	Goal	Action	Unseen Action
Raw Data	65.6±4.0%	84.6±1.2%	67.3±5.5%
Pruned Data	95.0±0.47%	94.0±0.8%	94.0±0.8%

TABLE 2.1: Accuracy results of the I-DRAGGN language model for Goal-oriented, Action-oriented, and Unseen Action-oriented commands. The factorized structure of the I-DRAGGN framework allows for generalization to unseen commands.

environment $7 \times 7 \times 4$, this made 31 action callable units to argument combination and 15 goal callable units to argument combination.

In order to collect the natural language to reward function data we used Amazon Mechanical Turk. We created 53 videos via AirSim Shah et al., 2017. We asked the workers to provide a natural language command that they thought would cause the drone to carry out the behavior observed in the video. Since goals have previously shown lower accuracy Karamcheti et al., 2017, we aimed for a higher ratio of goals to actions data. In total, our corpus has 2480 action and 1200 goal sentences.

Certain modifications to the Deep Learning model were made in order to improve accuracy, with two layers of GRU Cho et al., 2014. Specific parameters also had to be adjusted, such as 15 epochs, a learning rate of 0.01, an embedding size of 25, a dropout probability of 0.1, and a batch size of 32.

2.3.5 Grounding Module

Once the reward function is received by the base station, for action-oriented or goal-oriented tasks, the base station sends primitive actions to the drone after planning. For a goal-oriented task, a sequence of actions are sent and for action-oriented only a single command is sent to the drone.

2.4 Evaluation

Our evaluation aimed to assess the effectiveness of our approach at enabling natural language interaction, and to compare our approach with conventional interfaces using objective metrics (i.e., speed and quality of task completion) and subjective metrics (i.e., usability).

2.4.1 Corpus-based Evaluation

First, we assessed the effectiveness of the language understanding system at interpreting commands from 290 Amazon Mechanical Turk workers on a test-set. We collected a corpus with 2480 action commands and 1200 goal commands, and we refer to this as the "raw data." We cleaned this raw dataset as some annotators specified extraneous environmental objects unrelated to the task itself. We removed these extraneous words or tokens from the dataset to create a pruned corpus. We present results on both the raw and the pruned corpus in Table 2.1. We made a 90-10 partition for the goal-oriented and vanilla action-oriented training and testing datasets. For unseen action-oriented commands, we set 2240 data points for training and the rest for testing as only unseen combinations could be in the testing dataset.

Differences in accuracy can be seen in Table 2.1. We found that the learner has an easier time predicting commands seen previously and is capable of generalizing to unseen action commands given the factorization of the I-DRAGGN architecture. We also noticed an improvement in goal-oriented commands' accuracy with a higher goal to action data point ratio.

2.4.2 Demonstration

To demonstrate our interactive system, we set the workspace of the drone to be a 2×2 m surface. We set 60 cm to be the maximum distance from the ground, which is limited by the infrared sensor that the PiDrone is using. We created a grid-based environment. As the width and length of PiDrone is about 30 cm, we set each cell in the grid to be a 50×50 cm cube. If there is no command, the drone hovers at the center of the cell. Due to the lack of stability of the PiDrone and the cell's small size, the drone cannot stay stable in the target cell all the time, but the localization module enables the drone to correct its location. We have a $4 \times 4 \times 2$ grid, as we adapted the environment from Cleanup Domain MacGlashan et al., 2015, and we have three rooms. We have color attributes for each room (red, green, blue). The red room connects the green room and the blue room, as can be seen in Fig. 2.3b.

To simplify the learning for the user, who does not have experience with the drone, we avoided jargon like "roll," "pitch," and "yaw"; instead, we use direction commands like "forward" and "turn right". For example, when the user says "move forward three squares," a ROS message is sent to the drone and works with its localization module, so that the y coordinate of the target position of the drone is increased by 1.5 m. Then, the drone flies to the target position.

When the drone performs the task, it moves cell by cell. If *take photo* is required, the drone will fly lower or higher based on the altitude of the box. This mimics a *take photo* task in the real world, where the user might want to have an image closer to the scene or have a wider view of the scene. A video demonstration of the end-to-end system is available online[1].

2.4.3 User Study

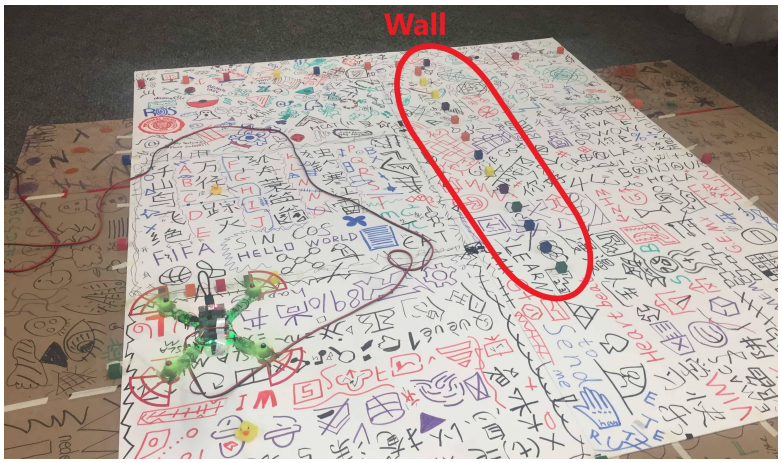
To understand how the MR and language interface works, we conducted an exploratory user study to compare our MR system, with and without natural language, to a baseline web control interface. Nine adults recruited from Brown University participated in the study. Participants received a $20 Amazon gift card as compensation for an expected 60 minutes of participation time.

Each participant used all three system interfaces within-subject (web interface, MR interface without language, MR interface with language) and completed the same three goal-oriented tasks using each interface. The three interfaces were presented to participants in random order, however participants always completed all three tasks for each interface in the same order.

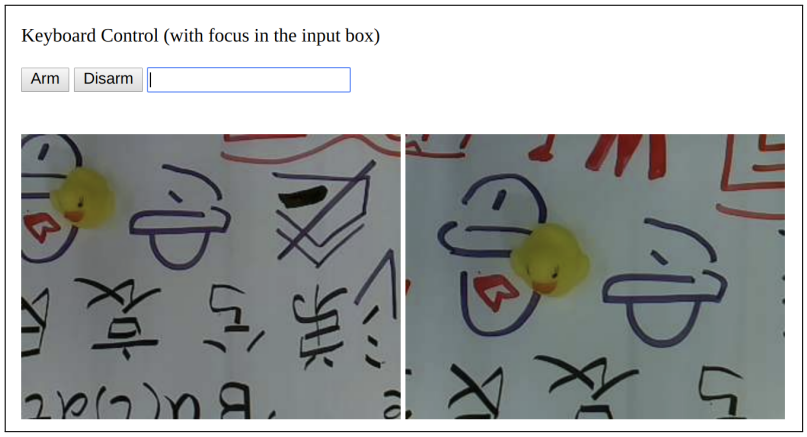
Procedure

After obtaining participant consent, we introduced the first interface. We allowed the participant to explore the interface for two minutes to become comfortable with

[1] https://youtu.be/T70b7Y7LW7Q



(A)

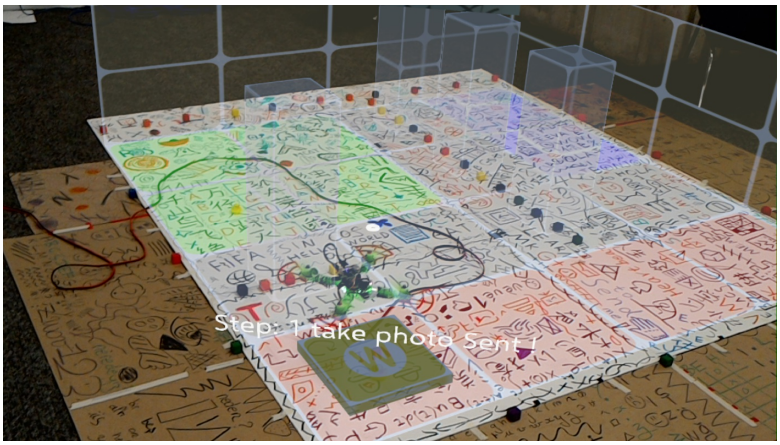


(B)

FIGURE 2.4: Web Interface.

(A) The user's view of the web interface.

(B) The web interface to control the drone. The user can input the keyword command through the text box. The left window is the live stream which captures images below the drone. The right window is the captured image once the user gives the "take photo" command.



(A)

(B)

FIGURE 2.6: MR without language model
After the user says "create box," "take photo," and "send task," the drone moves to the box and takes a photo.

(A) The text in the image reads: "Step: 1 take photo Sent !" The text appears one word at a time. The text is clearly readable when displayed in the HoloLens.

(B) A sample picture that would be taken by a participant in Tasks 1 and 3.

the system. Next, participants completed each of the three tasks in order. For each task, we recorded the command time (the time it took participants to instruct the drone for the task) and the execution time (the time it took the drone to execute the task). After completing all three tasks for a given interface, the participant completed the System Usability Scale (SUS) Brooke, 1996 for the interface. Participants completed the same sequence for the two remaining interfaces.

Task

We created three goal-oriented tasks that participants completed across all three interfaces. In Task 1, participants were instructed to command the drone to take a photo of a rubber ducky in the environment. In Task 2, participants were instructed to command the drone to move to a room with a specific color. In Task 3, participants were instructed to command the drone to take a photo of a rubber ducky and then move to a room with a specific color.

Interface

We describe the three interfaces next.

- **Web Interface:** The web interface allows the user to control the drone freely without the localization module. They can use the keyboard to command the drone to fly forward, back, left, right, up, down, and take a photo. The advantage of this interface is that use of the keyboard is familiar to novice users, especially for users who have ever played a computer game (e.g., a computer game where the user controls a car). In this interface the participant does not have any visual aids, but instead only sees physical cubes on the map which represent the obstacles. As this interface does not have planning and only allows the user to use low-level actions, the participants are instructed that the drone is not allowed to cross the wall as shown in Fig. 2.5b.
- **MR without Language Model:** This interface is similar to the one we proposed and described in Section 2.3, except that it does not include our language model. Like the MR system with language model, it still uses a verbal keyword recognition system from HoloLens to receive predefined phrases (which can be replaced by a button) and the MDP solver to navigate. However, it cannot take natural language sentences to execute high-level commands and instead requires the user to put individual landmarks for the drone to plan to, while avoiding obstacles. In addition, a "take photo" label can be added to the box, to instruct the drone to fly to the box and take a photo of it. Fig. 2.7b shows how a user can command the drone to complete the task "take a photo of a cell." Such an interface is similar in spirit to the idea of robot end user programming (Forbes et al., 2015; Huang, Lau, and Cakmak, 2016), albeit within an MR environment.
- **MR with Language Model:** This interface combines goal-based MDP planning with natural language input, as described in Section 2.3.

Results

We evaluated the three interfaces based on the time it took participants to command the drone in each task (reflecting the amount of attention required when using each interface), as well as on the time it took the drone to complete the task. In addition, we assessed users' experience of each interface, and measured system usability via the System Usability Scale (SUS).

Across all three interfaces, participants were fastest to command the drone via MR without language, followed by slightly longer times to command via MR with language, and then by the longest times to command the drone via the web interface (Fig. 2.8). This is true across all three tasks:

- In Task 1, command via MR without language (M = 8.79, SD = 2.72) was faster than via MR with language (M = 18.28, SD = 2.93), which were both faster than via the web interface (M = 27.09, SD = 5.40).
- In Task 2, command via MR without language (M = 5.98, SD = 2.28) was faster than via MR with language (M = 10.64, SD = 2.95), which were both faster than via the web interface (M = 17.24, SD = 4.42).

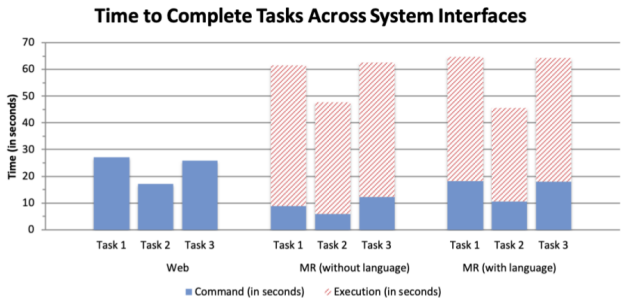


FIGURE 2.8: Average time (in seconds) required to complete each of the three tasks across all three interfaces
Each bar consists of the time to command the drone and the time for the drone to execute. However, for the web interface, the command times and execution times are one and the same (displayed as the solid blue bars), as the web interface involves continuous direct control of the drone.

- In Task 3, command via MR without language (M = 12.19, SD = 7.49) was faster than via MR with language (M = 17.92, SD = 4.29), which were both faster than via the web interface (M = 25.76, SD = 8.27).

Overall, the command times indicate that participants were able to command the drone more quickly via the MR interfaces, with control via the web interface taking longer to command the drone. These are promising initial findings in support of the relative ease of use of the MR interface over the web interface; the reduced amount of time that a user needs to spend controlling the drone frees human users to attend to other important tasks (such as providing oversight to the drone). Since the MR interface uses a goal-based system, with language as input, participants are able to control the drone intelligently and naturally. The system is in part limited by the state of the art of Google's Speech API, which cannot capture every sentence from participants accurately such that they had to repeat commands sometimes due to the loud sounds from the motor and propellers of the drone. This may partially account for why MR with natural language was not quicker to command the drone than the MR without natural language.

As can be seen in Fig. 2.8, the execution times for the MR interfaces were substantially longer than for the web interface (where the execution time and command time are one and the same). That is, after the drone received the commands via the MR interfaces, it took considerable time to actually execute the command and navigate through the environment. However, this is not a critique of the interface itself, but rather a limitation of the robotic platform we used in this study. The PiDrone is a low-cost drone with limited computing power and battery, which limits its localization capacity; with a more expensive, advanced drone it would be possible to cut the execution time drastically.

We also assessed users' experience of system usability with the SUS. All three systems fared well on system usability, with nearly-equal, high average system usability according to the SUS scores across the board: MR with language (M = 85.83, SD = 11.04), MR without language (M = 83.61, SD = 18.25), web interface (M = 84.44, SD = 12.30). Although we note the sizable standard deviations of the SUS scores, overall the SUS scores suggest that the MR system was no more burdensome than the web interface.

2.5 Conclusion

In this paper, we offer a mixed reality interface for controlling a drone with natural language. We demonstrated this system on a real robot and conducted a corpus-based evaluation, as well as an exploratory user study, to assess the system's effectiveness. The mixed reality interface allows people to provide landmarks that they can then refer to by using the natural language interface, which enables people to command drones with higher flexibility. Also, when using the MR interface people can simultaneously observe the drone and the environment while planning the task and giving commands, as compared to being forced to do these sequentially via other 2D and 3D interfaces. In our exploratory user study, we found that users were able to command the drone more quickly via both MR interfaces (with and without language) as compared to the web interface, with roughly equal system usability scores across all three interfaces.

Future work includes implementing the system on additional drones with larger workspaces and more complex flight patterns. We also seek to expand the scope of the tasks by letting users create more types of landmarks (including obstacles and rooms), so the user can also build models for specific environments and tasks. To improve the accuracy of task understanding, we hope to combine the gesture and language model Whitney et al., 2017 to allow the user to adjust landmarks to correct the actions of the drone and provide more intuitive communication between human and robot. Moreover, more visual cues could be added to the system to support the interactive execution of a plan Ganesan, 2017; Chakraborti et al., 2018. Our mixed reality interface is a promising step towards increasingly intuitive communication via natural language with robotic systems.

Chapter 3

Grounding Language to Landmarks in Arbitrary Outdoor Environments

FIGURE 3.1: 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.

This work was published in ICRA 2020 and is a joint work with Matthew Berg, Rebecca Mathew, Ariel Rotter-Aboyoun, Ellie Pavlick and Stefanie Tellex.

3.1 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 Tellex et al., 2011a; Artzi and Zettlemoyer, 2013; Paul et al., 2018; Oh et al., 2019. 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."*

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 capable of interpreting natural language commands, including references to nearby landmarks, with no training data for the environment. Our 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) OpenStreetMap contributors, 2017. 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 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%.

3.2 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 Kollar et al., 2014; Tellex et al., 2011c; Huang et al., 2010; Matuszek, Fox, and Koscher, 2010. To handle complex instructions, Tellex et al. (2011a) 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 Oh et al., 2019; Gopalan et al., 2018; Dong and Lapata, 2016. 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.

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

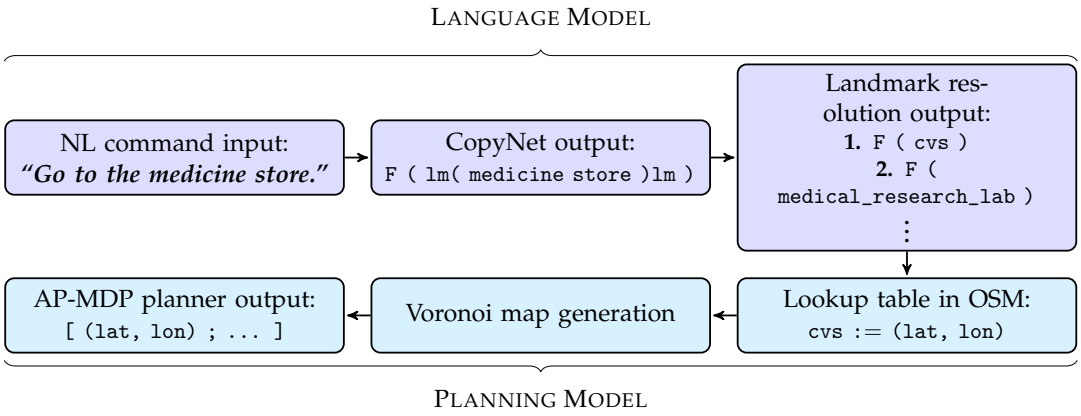


FIGURE 3.2: End-to-End System Pipeline
Natural language is given to the language model, which returns a grounded LTL formula. The planning model then creates a motion plan which satisfies the LTL formula.

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 Artzi and Zettlemoyer, 2013; Cheng et al., 2017; Misra et al., 2015; Damonte, Goel, and Chung, 2019; Yin et al., 2018. To create more domain-independent groundings, Cheng et al. (2017) demonstrates a neural semantic parser that uses an intermediate form containing natural language. Misra et al. (2015) 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 MacMahon, Stankiewicz, and Kuipers, 2006b; Kollar et al., 2010; Dzifcak et al., 2009b; Andreas and Klein, 2015. Dzifcak et al. (2009b) presents a framework for grounding natural language commands into a logical form representing goals and actions, while Kollar et al. (2010) 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.

3.3 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 3.2.

The language model grounds natural language commands to LTL formulae. The LTL structure is created by CopyNet Gu et al., 2016, 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. `F(`*CVS* $\wedge$ `F(`*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 Voronoi, 1908. The partitioned map along with the LTL formula are supplied to the AP-MDP planner Oh et al., 2019. This planner extracts goals and constraints from the LTL formula to create a motion plan as a series of latitude and longitude points.

3.3.1 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 (1992). For example, a command such as *"Go to the big blue bear but avoid the main green"* would have an LTL expression of `F(`*big blue bear* $\wedge$ $\neg$*main green*`)`.

3.3.2 CopyNet

To translate natural language commands into logical forms, current approaches use a Seq2Seq model Oh et al., 2019; Gopalan et al., 2018; Dong and Lapata, 2016. 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 Gu et al., 2016, 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 Bahdanau, Cho, and Bengio, 2014. 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 Hu, Chen, and Zhu, 2015, 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 pretrained GloVe embedding vectors Pennington, Socher, and Manning, 2014. 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. (2019) 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. 3.2, simplifying extraction of landmark referring expressions for the landmark resolution model. Finally, we limit the dataset to the following three LTL structures:

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

3.3.3 Landmark Resolution Model

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 Lovelace, Hegarty, and Montello, 2014. Landmarks are geographic objects important to human spatial cognition Richter and Winter, 2014. Following previous work Rousell et al., 2015; Drager and Koller, 2012 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] https://github.com/adamklec/copynet
[3] https://github.com/mjc92/CopyNet

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.

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 Pennington, Socher, and Manning, 2014; Mikolov et al., 2013; Bojanowski et al., 2016; Joulin et al., 2017; Grave et al., 2018. 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:house number` 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 3.4.2. The cosine distance between two vectors is defined as the difference between 1 and their cosine similarity.

3.3.4 Voronoi Maps and Planning

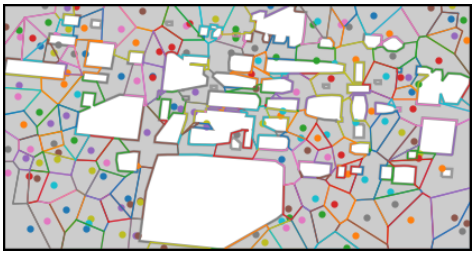


FIGURE 3.3: Map partitioned into Voronoi cells
White holes represent regions containing landmarks.

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. (2019) 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 (Voronoi, 1908). 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. 3.3. 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. (2019), 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.

3.4 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].

3.4.1 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 *Unity*, using outdoor environments generated with the Mapbox SDK *Mapbox Unity SDK* (Fig. 3.1). Using ROS and ROS# Quigley et al., 2009; Siemens, 2017, 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 3.1, our model accurately grounds natural language commands to LTL and formed correct motion plans for 76.19% of user commands. In this table,

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

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) Hart and Staveland, 1988. On average, users reported high performance and low workload (Table 3.2). Additionally, we use the Systems Usability Scale (SUS) Brooke, 1996 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 Sauro, 2011.

	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 3.1: 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 3.2: Raw NASA-TLX scores on a 20 point scale

3.4.2 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.

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 3.3. 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 3.3). 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 3.3: CopyNet accuracy

Landmark Resolution Evaluation

We compare our landmark resolution model to other models, as shown in Table 3.4. 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) Sammut and Webb, 2010 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 Bojanowski et al., 2016; Grave et al., 2018, which uses such information, to larger word embedding models like Word2Vec and GloVe Pennington, Socher, and Manning, 2014; Mikolov et al., 2013.

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

3.4.3 Corpus-Based Evaluation

We test our language model's ability to both identify the appropriate LTL structure and properly extract landmarks from unseen commands by collecting a test set of

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

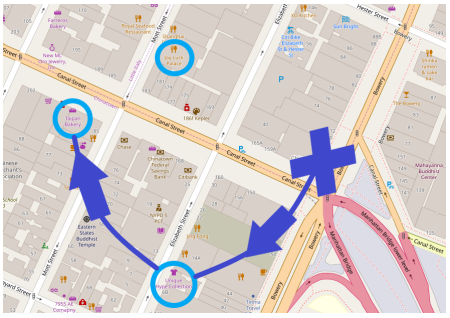


FIGURE 3.4: AMT trajectory example
An OSM region with trajectory that corresponds to `F(lm(`l_1`)lm & F(lm(`l_2`)lm))`

challenging natural language commands from AMT. We collected commands for 22 urban American regions. (Table 3.5).

AMT workers viewed a screenshot of a region in OSM with an overlaid trajectory (Fig. 3.4). 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.

3.5 Conclusion

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

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 3.5: Corpus-based language pipeline accuracy

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.

Chapter 4

Conclusion

In this thesis, we investigated how factors such as (1) the robot's affordances and (2) the environment in which it plans affects Natural Language (NL) instruction following. In the second chapter, we presented a Mixed Reality (MR) and NL interface to control a drone and demonstrate that the combined interface enables people to command drones with higher flexibility. In the third chapter, we presented a framework that grounds natural language commands containing unseen landmarks to motion plans for a robot operating in outdoor environments. This work allows users to not pre-train a language model for a specific environment and achieves generalization in multiple outdoor environments.

To ensure NL instruction generalization across types of robots, immediate future work includes implementing our MR & NL interface on additional robots with larger workspaces and letting users create more types of landmarks (including obstacles and rooms). Further future work can be on identifying contextual information such as the robot's abilities or skills and the environments' features and incorporating those into a robot-agnostic language model.

To generalize NL instruction understanding across types of environments, we can improve components of our framework, such as the copying mechanism or adding a vision module to our landmark resolution model. Moreover, 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.

While these are the first steps to make robots accessible to untrained users, this thesis shows that more work needs to be done jointly in different areas of computer science such as natural language processing, computer vision and decision making to achieve seamless online communication.

Acknowledgements

I would like to thank my research advisors Stefanie Tellex and Ellie Pavlick for guiding me within the fields of Natural Language Processing & Robotics and offering the most welcoming lab environments. I would like to thank all of my research partners and mentors including Rebecca Mathew, Matthew Berg, Ariel Rotter-Aboyoun, Thao Nguyen, Kaiyu Zheng, Baichuan Huang, Eric Rosen, Roma Patel, Daniel Ullman and Nakul Gopalan. I would like to thank my academic advisor Thomas Doeppner for his valuable input in my academic course decisions. Finally, a big thank you to my family for supporting me in my interests.

Bibliography

Abel, David (2017). simple-rl. https://github.com/david-abel/simple_rl, [Accessed: 2018].

Andreas, Jacob and Dan Klein (2015). "Alignment-based compositional semantics for instruction following". In: *CoRR* abs/1508.06491. arXiv: 1508.06491. URL: http://arxiv.org/abs/1508.06491.

Artzi, Yoav and Luke Zettlemoyer (2013). "Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions". In: *Transactions of the Association for Computational Linguistics* 1, pp. 49–62. DOI: 10.1162/tacl_a_00209. URL: https://www.aclweb.org/anthology/Q13-1005.

Arumugam, Dilip et al. (2017). "Accurately and Efficiently Interpreting Human-Robot Instructions of Varying Granularities". In: *Robotics: Science and Systems*.

Arumugam, Dilip et al. (2018). "Grounding natural language instructions to semantic goal representations for abstraction and generalization". In: *Autonomous Robots*.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural Machine Translation by Jointly Learning to Align and Translate". In: *CoRR* abs/1409.0473.

Bellman, R. (1957). "A Markovian decision process". In: *Journal of Mathematics and Mechanics* 6 (5), pp. 679–684.

Bojanowski, Piotr et al. (2016). "Enriching Word Vectors with Subword Information". In: *arXiv preprint arXiv:1607.04606*.

Bradski, G. (2000). "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools*.

Brand, Isaiah et al. (Oct. 2018). "An integrated introduction to robotics for the next generation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Brooke, John (1996). "SUS-A quick and dirty usability scale". In: *Usability Evaluation in Industry* 189.194, pp. 4–7.

Chakraborti, Tathagata et al. (2018). "Projection-Aware Task Planning and Execution for Human-in-the-Loop Operation of Robots in a Mixed-Reality Workspace". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 4476–4482.

Cheng, Jianpeng et al. (July 2017). "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, pp. 44–55. DOI: 10.18653/v1/P17-1005. URL: https://www.aclweb.org/anthology/P17-1005.

Cho, Kyunghyun et al. (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: http://www.aclweb.org/anthology/D14-1179.

Chung, I. et al. (2015). "On the performance of hierarchical distributed correspondence graphs for efficient symbol grounding of robot instructions". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Damonte, Marco, Rahul Goel, and Tagyoung Chung (2019). "Practical Semantic Parsing for Spoken Language Understanding". In: *CoRR* abs/1903.04521. arXiv: 1903.04521. URL: http://arxiv.org/abs/1903.04521.

DJI (2018). *The Future Of Possible*. URL: https://www.dji.com/.

Dong, Li and Mirella Lapata (2016). "Language to Logical Form with Neural Attention". In: *CoRR* abs/1601.01280. arXiv: 1601.01280. URL: http://arxiv.org/abs/1601.01280.

Drager, Markus and Alexander Koller (2012). "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*.

Dzifcak, Juraj et al. (2009a). "What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution". In: *IEEE International Conference on Robotics and Automation*.

Dzifcak, Juraj et al. (2009b). "What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution". In: *2009 IEEE International Conference on Robotics and Automation*, pp. 4163–4168.

Erat, Okan et al. (2018). "Drone-Augmented Human Vision: Exocentric Control for Drones Exploring Hidden Areas". In: *IEEE Transactions on Visualization and Computer Graphics* 24.4, pp. 1437–1446.

Forbes, Maxwell et al. (2015). "Robot programming by demonstration with situated spatial language understanding". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, pp. 2014–2020.

Ganesan, Ramsundar Kalpagam (2017). "Mediating Human-Robot Collaboration through Mixed Reality Cues". PhD thesis. Arizona State University.

Google (2018). *Google Speech API*. URL: https://cloud.google.com/speech-to-text/.

Gopalan, Nakul et al. (2018). "Sequence-to-Sequence Language Grounding of Non-Markovian Task Specifications". In: *Robotics: Science and Systems*.

Grave, Edouard et al. (2018). "Learning Word Vectors for 157 Languages". In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Gu, Jiatao et al. (2016). "Incorporating Copying Mechanism in Sequence-to-Sequence Learning". In: *ArXiv* abs/1603.06393.

Hart, Sandra G. and Lowell E. Staveland (1988). "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research". In: *Human Mental Workload*. Ed. by Peter A. Hancock and Najmedin Meshkati. Vol. 52. Advances in Psychology. North-Holland, pp. 139 –183. DOI: https://doi.org/10.1016/S0166-4115(08)62386-9. URL: http://www.sciencedirect.com/science/article/pii/S0166411508623869.

Herrmann, Roman and Ludger Schmidt (2018). "Design and Evaluation of a Natural User Interface for Piloting an Unmanned Aerial Vehicle". In: *i-com* 17.1, pp. 15–24.

Hoenig, Wolfgang et al. (2015). "Mixed reality for robotics". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, pp. 5382–5387.

Howard, T. M., S. Tellex, and N. Roy (2014). "A natural language planner interface for mobile manipulators". In: *IEEE International Conference on Robotics and Automation*.

Hu, Baotian, Qingcai Chen, and Fangze Zhu (2015). "LCSTS: A Large Scale Chinese Short Text Summarization Dataset". In: *EMNLP*.

Huang, Albert S et al. (2010). "Natural language command of an autonomous micro-air vehicle". In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, pp. 2663–2669.

Huang, Justin, Tessa Lau, and Maya Cakmak (2016). "Design and evaluation of a rapid programming system for service robots". In: *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*. IEEE Press, pp. 295–302.

Joulin, Armand et al. (Apr. 2017). "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, pp. 427–431.

Kam, Hyeong Ryeol et al. (2015). "Rviz: A toolkit for real domain data visualization". In: *Telecommunication Systems* 60.2, pp. 337–345.

Karamcheti, Siddharth et al. (2017). "A Tale of Two DRAGGNs: A Hybrid Approach for Interpreting Action-Oriented and Goal-Oriented Instructions". In: *Annual Meeting of the Association for Computational Linguistics Workshop on Language Grounding for Robotics*.

Kollar, Thomas et al. (2010). "Toward understanding natural language directions". In: *HRI 2010*.

– (2014). "Grounding Verbs of Motion in Natural Language Commands to Robots". In: *Experimental Robotics Springer Tracts in Advanced Robotics*, 31–47. DOI: 10.1007/978-3-642-28572-1_3.

Kress-Gazit, Hadas, Georgios E. Fainekos, and George J. Pappas (2008). "Translating Structured English to Robot Controllers". In: *Advanced Robotics* 22.12, pp. 1343–1359.

Lovelace, Kristin L., Mary Hegarty, and Daniel R. Montello (2014). "Elements of Good Route Directions in Familiar and Unfamilar Environments". In:

MacGlashan, James et al. (2015). "Grounding English Commands to Reward Functions". In: *Robotics: Science and Systems*.

MacMahon, Matt, Brian Stankiewicz, and Benjamin Kuipers (2006a). "Walk the talk: Connecting language, knowledge, and action in route instructions". In: *National Conference on Artificial Intelligence*.

– (Jan. 2006b). "Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions." In:

Manna, Zohar and Amir Pnueli (1992). *The Temporal Logic of Reactive and Concurrent Systems*. Berlin, Heidelberg: Springer-Verlag. ISBN: 0-387-97664-7.

Mapbox. *Mapbox Unity SDK*. URL: https://github.com/mapbox/mapbox-unity-sdk.

Matuszek, Cynthia, Dieter Fox, and Karl Koscher (2010). "Following Directions Using Statistical Machine Translation". In: *Proceedings of the 5th ACM/IEEE International Conference on Human-robot Interaction*. HRI '10. Piscataway, NJ, USA: IEEE Press, pp. 251–258. ISBN: 978-1-4244-4893-7. URL: http://dl.acm.org/citation.cfm?id=1734454.1734552.

Mei, Hongyuan, Mohit Bansal, and Matthew R. Walter (2016). "Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences". In: *AAAI Conference on Artificial Intelligence*.

Microsoft (2018). *Microsoft HoloLens*. URL: https://www.microsoft.com/en-us/hololens/.

Mikolov, Tomas et al. (2013). "Distributed Representations of Words and Phrases and Their Compositionality". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'13. Lake Tahoe, Nevada:

Curran Associates Inc., pp. 3111–3119. URL: http://dl.acm.org/citation.cfm?id=2999792.2999959.

Misra, Dipendra Kumar et al. (July 2015). "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, pp. 992–1002. DOI: 10.3115/v1/P15-1096. URL: https://www.aclweb.org/anthology/P15-1096.

Mur-Artal, Raúl, J. M. M. Montiel, and Juan D. Tardós (2015). "ORB-SLAM: A Versatile and Accurate Monocular SLAM System". In: *IEEE Transactions on Robotics* 31.5, pp. 1147–1163. DOI: 10.1109/TRO.2015.2463671.

Oh, Yoonseon et al. (2019). "Planning with State Abstractions for Non-Markovian Task Specifications". In: *Robotics: Science and Systems*.

OpenStreetMap contributors (2017). *Planet dump retrieved from https://planet.osm.org*. https://www.openstreetmap.org.

Paszke, Adam et al. (2017). "Automatic differentiation in PyTorch". In: *Neural Information Processing Systems Workshop on The Future of Gradient-based Machine Learning Software & Techniques*.

Paul, Rohan et al. (2018). "Temporal Grounding Graphs for Language Understanding with Accrued Visual-Language Context". In: *ArXiv* abs/1811.06966.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). "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, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://www.aclweb.org/anthology/D14-1162.

Peschel, J. M. and R. R. Murphy (Jan. 2013). "On the Human–Machine Interaction of Unmanned Aerial System Mission Specialists". In: *IEEE Transactions on Human-Machine Systems* 43.1, pp. 53–62. ISSN: 2168-2291. DOI: 10.1109/TSMCC.2012.2220133.

Quigley, Morgan et al. (2009). "ROS: An open-source Robot Operating System". In: *IEEE International Conference on Robotics and Automation Workshop on Open Source Software*.

Richter, Kai-Florian and Stephan Winter (2014). In: *Landmarks: GIScience for Intelligent Services*. Chap. Introduction: What landmarks are, and why they are important.

Rosen, Eric et al. (2017). "Communicating robot arm motion intent through mixed reality head-mounted displays". In: *International Symposium on Robotics Research*.

Rousell, Adam et al. (2015). "Extraction of landmarks from OpenStreetMap for use in navigational instructions". In: *Association of Geographic Information Laboratories in Europe*.

"TF–IDF" (2010). In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, pp. 986–987. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_832. URL: https://doi.org/10.1007/978-0-387-30164-8_832.

Sauro, Jeff (2011). *SUStisfied? Little-Known System Usability Scale Facts User Experience Magazine*. URL: https://uxpamagazine.org/sustified/.

Shah, Shital et al. (2017). "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles". In: *Field and Service Robotics*. eprint: arXiv:1705.05065. URL: https://arxiv.org/abs/1705.05065.

Sibirtseva, Elena et al. (2018). "A Comparison of Visualisation Methods for Disambiguating Verbal Requests in Human-Robot Interaction". In: *arXiv preprint arXiv:1801.08760*.

Siemens (2017). ROS#. https://github.com/siemens/ros-sharp, [Accessed: 2018].

Skydio (2018). *The self-flying camera has arrived*. URL: https://www.skydio.com/.

Tellex, Stefanie et al. (2011a). "Approaching the Symbol Grounding Problem with Probabilistic Graphical Models". In: *AI Magazine* 32.4, p. 64. DOI: 10.1609/aimag.v32i4.2384.

– (2011b). "Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation". In: *AAAI Conference on Artificial Intelligence*.

– (2011c). "Understanding Natural Langugage Commands for Robotic Navigation and Mobile Manipulation". In: *National Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence.

Thrun, Sebastian, Wolfram Burgard, and Dieter Fox (2005). *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press. ISBN: 0262201623.

Unity Technologies. *Unity*. Version 2019.3.0a6. URL: https://unity.com/.

Voronoi, Georges (1908). "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs." In: *Journal für die reine und angewandte Mathematik* 134, pp. 198–287.

Whitney, David et al. (2017). "Reducing errors in object-fetching interactions through social feedback". In: *International Conference on Robotics and Automation*.

Yin, Pengcheng et al. (2018). "StructVAE: Tree-structured Latent Variable Models for Semi-supervised Semantic Parsing". In: *CoRR* abs/1806.07832. arXiv: 1806.07832. URL: http://arxiv.org/abs/1806.07832.

Zettlemoyer, Luke S. and Michael Collins (2005). "Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars". In: *Uncertainty in Artificial Intelligence*.