Flight, Camera, Action!
Using Natural Language and Mixed Reality to Control a Drone

Baichuan Huang[1], Deniz Bayazit[1], Daniel Ullman[1], Nakul Gopalan[1] and Stefanie Tellex[1]

***Abstract*— With increasing autonomy, robots like drones are more accessible to untrained users. Most users control the drone using a low-level interface, such as a radio-controlled (RC) controller. For a wider adoption of these technologies by the public, a much higher-level interface, such as natural language or mixed reality (MR), could be well-suited as it allows the automation of the control of the agent in a goal-oriented setting. In this paper we present an interface that uses natural language grounding within an MR environment to solve high-level task and navigational instructions given to an autonomous drone. To the best of our knowledge, this is the first work to perform fully autonomous language grounding in a MR setting for a robot. Given a map, our interface first grounds natural language commands to goal states within a Markov Decision Process (MDP) framework. Then, it passes the goal state 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 provide 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 conducted an exploratory user study to assess users' experience of our MR interface with and without natural language, as compared to a web interface. We found that users were able to command the drone more quickly via both MR interfaces as compared to the web interface, with roughly equal system usability scores across all three interfaces.**

I. INTRODUCTION

As drones 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.

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 [34, 10]. Current natural language interfaces require a predefined model of the environment including landmarks [35, 20] 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 [25] to control a drone in hidden areas with gaze and gesture [12]. However, to the best of our knowledge, MR has not been used to give high-level language commands to a drone.

Fig. 1: An example of a task-oriented navigational command given to a drone through the Mixed Reality and language interface. The scene is clear when displayed via the HoloLens.

We address these interaction problems by using natural language within MR to provide an intuitive high-level interface for controlling a drone using goal-based planning. By using MR, a user can annotate landmarks in the drone's frame of reference, enabling them to provide high-level natural language commands referring to the landmarks. The process starts with the MR interface displaying the virtual environment of a room that can be adjusted to overlay on physical reality with 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 [20, 3] 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 [6] 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 environment, a base station for language mapping and MDP solving, and Robot Operating System (ROS) 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 [31], we asked the workers to give a possible natural language command that would result in the execution of the observed

[1]Brown University, Providence, RI, 02912, USA. Correspondence: {baichuan_huang, deniz_bayazit, daniel_ullman, nakul_gopalan, stefanie_tellex}@brown.edu

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, 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.

II. 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 [38, 23, 21, 9, 22, 24, 2]. 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 [11]. Some methods provide models (such as MARCO [23] and DCG [16]) which connect natural language phrases to physical objects, actions, and environment. Huang et al. [17] 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. [22]. Additionally, Karamcheti et al. [20, 3] 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 [28]. These specialized interfaces are not necessarily intuitive for inexperienced users, as they are low-level forms of control. Commercial drones like Skydio [34] and DJI [10] 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 [19]. Using MR to control an arm or a drone is facilitated by the use of gesture and gazes [12, 30]. Rosen et al. [30] presented a MR interface to inform the user of a robot's intent. Sibirtseva et al. [32] use a combination of natural language in 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.

III. APPROACH

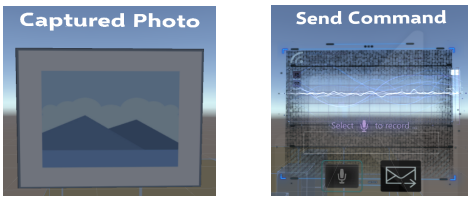
We consider the problem of drone navigation in an environment with objects and obstacles. Our system allows a human operator to give low-level instructions like "move forward three squares," or goal-oriented commands such as "go to the chair and take a picture." The drone then interprets these instructions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

A. PiDrone

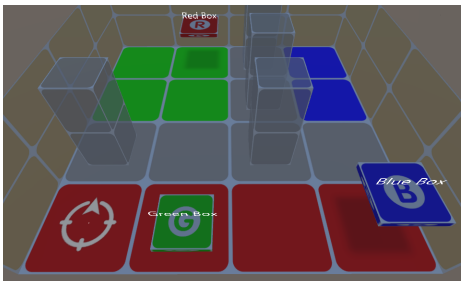
We require an autonomous drone that can localize itself in an environment. We chose the PiDrone as our robotic platform because it is a low-cost system that is fully customizable [6]. The drone is equipped with one downward-facing camera. We have also implemented localization with a particle filter (Monte Carlo localization) [36]. The drone flies over a highly textured planar surface, and we use the OpenCV library [5] to extract and detect 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 [36]. 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 [29]. 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 [26] 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).

B. Mixed Reality Interface

We use natural language, gaze, and gesture to control a drone through Mixed Reality. The position of the drone is



(a)



(b)

Fig. 2: The UI in Unity 3D. (a) Left UI shows the photo captured by the drone. Right UI shows the voice input (b) The environment in MR.

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 HoloLens' frame. As studied in Hoenig et al. [15], 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, we created a push-to-talk language input 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 mistakingly triggers voice activity detection in the speech recognition system. The user records their command with the UI and sends the natural language command to the drone. We use Google's Speech API [14] 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 [33] to connect Unity 3D to ROS.

C. 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, and $\mathcal{A}$ 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 limit the horizon of the planner [4]. We use the simple-rl [1] as the MDP solver to produce a policy which maps states to actions. The goal is to maximize the total 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 [1] to create and solve an MDP domain.

The domain we use is shown in Fig. 2. This model is adapted from the Cleanup Domain which is introduced by MacGlashan et al. [22]. 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**.

D. Language Model

The I-DRAGGN framework [20, 3] 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. [20]. We used PyTorch 0.4.0 [27] 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. [20]. 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. 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 and three suitable goal callable units. With the dimensions of the initial 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 simulation [31]. We asked the workers

	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 I: Accuracy results of the I-DRAGGN language model out of 3 experiments for Goal-oriented, Action-oriented and Unseen Action-oriented commands. The factorized structure of the I-DRAGGN framework allows for generalization to unseen commands.

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 [20], 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 [8]. 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.

E. 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.

IV. EVALUATION

Our evaluation aimed to assess the effectiveness of our approach at enabling natural language interaction, and to compare our approach to conventional interfaces in terms of objective metrics (such as speed and quality of task completion). We also assess subjective metrics such as convenience and workload.

A. Corpus-based Evaluation

First, we assessed the effectiveness of the language understanding system at interpreting commands from 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 I. 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.

The difference in the accuracies can be seen in Table I where they have been calculated out of three experiments. We notice that the learner has an easier time predicting commands it has seen previously and is capable of generalizing to unseen action commands given the factorization of the I-DRAGGN architecture. We also notice an improvement in goal-oriented commands' accuracy with a higher goal to action data point ratio.

B. 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 considered a grid-based environment. As the width and length of PiDrone is around 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 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 [22], and we have three rooms. To endow the meaning of the task and also provide the human perspective to the user, 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.

To simplify the learning for the user who does not have experience with the drone, we did not use jargon like "roll," "pitch," and "yaw"; instead, we use direction commands for the drone The drone's primitive actions are *forward*, *back*, *turn left*, *turn right*, *up*, *down*, and *take photo*. 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.

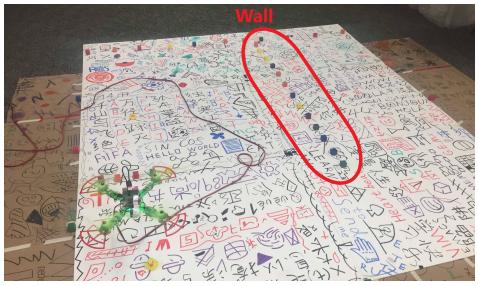
When the drone carries the task, it moves cell by cell. If *take photo* is required, the drone will fly lower or higher based on the altitude of the box. This mimics the *take photo* task in the real world, the user might want to have an image closer to the scene or have a wider view of the scene. A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

C. User Study

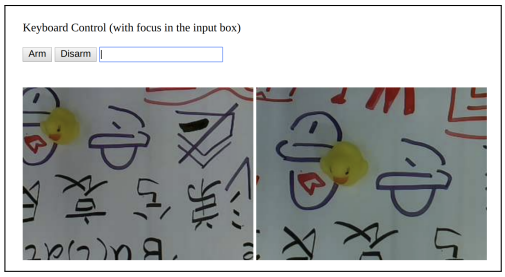
To understand how the MR and language interface works, we conducted an exploratory user study to compare our MR system, with and without natural language, to a baseline web control interface. Nine adults recruited from Brown University participated in the study. Participants received a $20 Amazon gift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

[1] https://youtu.be/G18Nn7-UOxk

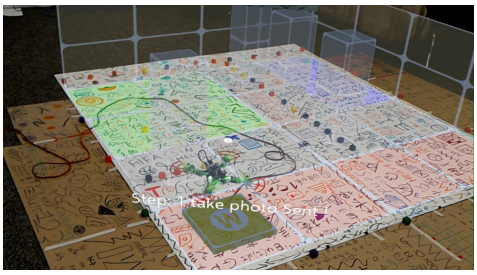


(a)



(b)

Fig. 3: 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)

Fig. 4: 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) Shows a sample pictures that would be taken by a participant in Tasks 1 and 3

1) 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 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) [7] for the interface. Participants completed the same sequence for the two remaining interfaces.

2) 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 to move to a room with specific color.

3) 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 users to use low-level actions, the participants are instructed that the drone is not allowed to cross the wall as shown in Fig. 3.
- **MR without Language Model:** This interface is similar to the one we proposed and described in the Approach section, except that this version 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 in between, 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. 4 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 [13, 18], albeit within an MR environment.
- **MR with Language Model:** This interface combines goal-based MDP planning with natural language input, as described in the Approach section.

4) 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. 5). 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).
- 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. This may partially account for why MR with natural language was not quicker to command the drone than the MR without natural language. Further, the loud sound from the motor and the propellers, disturbs the quality of voice recording done by the Google Speech API.

As can be seen in Fig. 5, 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.

Finally, 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 three interfaces:

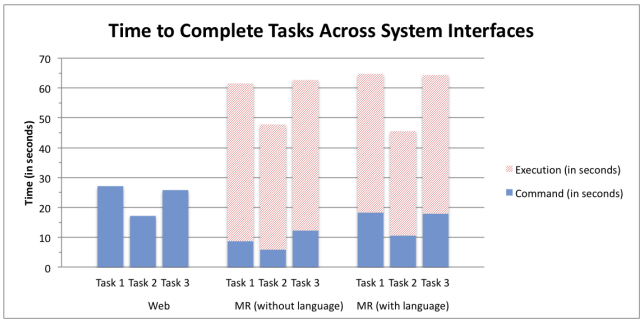


Fig. 5: The bars show the average time (in seconds) required to complete each of the three tasks across all three interfaces. Each bar consists of the time to command the drone and the time for the drone to execute. However, for the web interface, the command times and execution times are one and the same (displayed as the solid blue bars), as the web interface involves continuous direct control of the drone.

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.

V. 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 drone 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 as compared to the web interface, with roughly equal system usability scores across all three interfaces.

In the future, we intend to implement the system on additional drones with a larger workspace and more complex flight characteristics. 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 [37] to allow the user to adjust landmarks to correct the actions of the drone and provide more intuitive communication between human and robot. Our mixed reality interface is a promising step towards increasingly intuitive communication via natural language with robotic systems.

References

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

[2] Dilip Arumugam, Siddharth Karamcheti, Nakul Gopalan, Lawson L.S. Wong, and Stefanie Tellex. Accurately and efficiently interpreting human-robot instructions of varying granularities. *Robotics: Science and Systems*, 2017.

[3] Dilip Arumugam, Siddharth Karamcheti, Nakul Gopalan, Edward C Williams, Mina Rhee, Lawson LS Wong, and Stefanie Tellex. Grounding Natural Language Instructions to Semantic Goal Representations for Abstraction and Generalization. *Autonomous Robots*, 2018.

[4] R. Bellman. A Markovian decision process. *Journal of Mathematics and Mechanics*, 6:679–684, 1957.

[5] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[6] Isaiah Brand, Josh Roy, Aaron Ray, John Oberlin, and Stefanie Tellex. An Integrated Introduction to Robotics for the Next Generation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2018.

[7] John Brooke. Sus: A quick and dirty usability scale. 189, 11 1995.

[8] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 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)*, pages 1724–1734. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1179. URL http://www.aclweb.org/anthology/D14-1179.

[9] I. Chung, O. Propp, M. R. Walter, and T. M. Howard. 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*, 2015.

[10] DJI. The future of possible, 2018. URL https://www.dji.com/.

[11] Juraj Dzifcak, Matthias Scheutz, Chitta Baral, and Paul 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. In *IEEE International Conference on Robotics and Automation*, 2009.

[12] Okan Erat, Werner Alexander Isop, Denis Kalkofen, and Dieter Schmalstieg. Drone-augmented human vision: Exocentric control for drones exploring hidden areas. *IEEE transactions on visualization and computer graphics*, 24(4):1437–1446, 2018.

[13] Maxwell Forbes, Rajesh PN Rao, Luke Zettlemoyer, and Maya Cakmak. Robot programming by demonstration with situated spatial language understanding. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2014–2020. IEEE, 2015.

[14] Google. Google speech api, 2018. URL https://cloud.google.com/speech-to-text/.

[15] Wolfgang Hoenig, Christina Milanes, Lisa Scaria, Thai Phan, Mark Bolas, and Nora Ayanian. Mixed reality for robotics. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 5382–5387. IEEE, 2015.

[16] T. M. Howard, S. Tellex, and N. Roy. A natural language planner interface for mobile manipulators. In *IEEE International Conference on Robotics and Automation*, 2014.

[17] Albert S Huang, Stefanie Tellex, Abraham Bachrach, Thomas Kollar, Deb Roy, and Nicholas Roy. Natural Language Command of an Autonomous Micro-air Vehicle. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2663–2669. IEEE, 2010.

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

[19] Hyeong Ryeol Kam, Sung-Ho Lee, Taejung Park, and Chang-Hun Kim. Rviz: a toolkit for real domain data visualization. *Telecommunication Systems*, 60(2):337–345, 2015.

[20] Siddharth Karamcheti, Edward C. Williams, Dilip Arumugam, Mina Rhee, Nakul Gopalan, Lawson L.S. Wong, and Stefanie Tellex. A tale of two DRAGGNs: A hybrid approach for interpreting action-oriented and goal-oriented instructions. *Annual Meeting of the Association for Computational Linguistics Workshop on Language Grounding for Robotics*, 2017.

[21] Hadas Kress-Gazit, Georgios E. Fainekos, and George J. Pappas. Translating structured english to robot controllers. *Advanced Robotics*, 22(12): 1343–1359, 2008.

[22] James MacGlashan, Monica Babes-Vroman, Marie desJardins, Michael L. Littman, Smaranda Muresan, Shawn Squire, Stefanie Tellex, Dilip Arumugam, and Lei Yang. Grounding english commands to reward functions. In *Robotics: Science and Systems*, 2015.

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

[24] Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *AAAI Conference on Artificial Intelligence*, 2016.

[25] Microsoft. Microsoft hololens, 2018. URL https://www.microsoft.com/en-us/hololens/.

[26] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*,

31(5):1147–1163, 2015. doi: 10.1109/TRO.2015.2463671.

[27] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *Neural Information Processing Systems Workshop on The Future of Gradient-based Machine Learning Software & Techniques*, 2017.

[28] J. M. Peschel and R. R. Murphy. On the humanmachine interaction of unmanned aerial system mission specialists. *IEEE Transactions on Human-Machine Systems*, 43(1):53–62, Jan 2013. ISSN 2168-2291. doi: 10.1109/TSMCC.2012.2220133.

[29] Morgan Quigley, Josh Faust, Tully Foote, and Jeremy Leibs. ROS: an open-source robot operating system. In *IEEE International Conference on Robotics and Automation Workshop on Open Source Software*, 2009.

[30] Eric Rosen, David Whitney, Elizabeth Phillips, Gary Chien, James Tompkin, George Konidaris, and Stefanie Tellex. Communicating Robot Arm Motion Intent Through Mixed Reality Head-mounted Displays. In *International Symposium on Robotics Research*, 2017.

[31] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. URL `https://arxiv.org/abs/1705.05065`.

[32] Elena Sibirtseva, Dimosthenis Kontogiorgos, Olov Nykvist, Hakan Karaoguz, Iolanda Leite, Joakim Gustafson, and Danica Kragic. A comparison of visualisation methods for disambiguating verbal requests in human-robot interaction. *arXiv preprint arXiv:1801.08760*, 2018.

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

[34] Skydio. The self-flying camera has arrived, 2018. URL `https://www.skydio.com/`.

[35] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R. Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *AAAI Conference on Artificial Intelligence*, 2011.

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

[37] David Whitney, Eric Rosen, James MacGlashan, Lawson LS Wong, and Stefanie Tellex. Reducing errors in object-fetching interactions through social feedback. In *International Conference on Robotics and Automation*, 2017.

[38] Luke S. Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Uncertainty in Artificial Intelligence*, 2005.