Human-Swarm Interaction through Natural Language Commands

Roman Ibrahimov and Yifei Hu

***Abstract*—In the recent years, there has been a rapid growth in the field of Technology, which made it possible to produce high-performance robots in reduced cost and size. As a result, nowadays, it is possible to deploy a large amount of robots that collaborate with each other autonomously in a wide variety of applications, starting from search and rescue to theatrical performance. Researchers in the domains of bio-inspired Robotics and Control Theory are coming up with new approaches to the give the robotic swarms more autonomy. Although there might be a high degree of autonomy in the swarm system, full or partial presence of human operator, who interact with the system, is always more beneficial to avoid the shortcomings of autonomy. To achieve tangible interaction, a number of approaches such as tactile wearables, hand gesture recognition, etc. have been proposed. However, all these approaches have very narrow channel of information exchange and the most natural and productive way to interact with the swarm is yet to be discovered. In this work, we are applying Natural Language Processing (NLP) to build a rich and intuitive interaction approach in which a human gives voice commands to the robots for task fulfillment.In this work, we have implemented a simulation of four E-puck robots that receives voice commands from the user and they together manipulate an object based on the command. Furthermore, a single nano-quadcopter that grabs a physical object to deliver to the user was tested.**

***Index Terms*—Consensus Algorithm; NLP; Rendezvous Algorithm; Multi-Agent Systems; Swarm.**

I. Introduction

A. Background

Recent technological advancements can make it possible to deploy a swarm of robots in real-life scenarios such as mining [1], bridge inspection [2], search and rescue [3], space applications [4], etc. In such systems, agents share a common coordination that is based on distributed algorithms and information processing among all agents. The global behaviour of the swarms is not directly stated and it emerges throughout local interactions. Thus, individual agents cannot always fulfil the desired tasks by their own.

B. Motivation

Although there exist a number of work that provides robust swarm autonomy, according to [5], human presence in the system is still needed to 1) recognize and mitigate shortcomings of the autonomy; 2) have available "out-of-band" information not accessible to the autonomy and that can be utilized to increase performance; and 3) convey changes in intent as mission goals change.

C. Problem Statement

We have reviewed a number of works in Literature Review section. The reviewed projects seem to be very innovative and user-friendly. However, the HSI is not tangible and the operators only experience interaction throughout another device. The channel of information exchange is very narrow. The systems do not provide any feedback when there are unforeseen cases. Moreover, a novice user needs to have some time in order to learn how to use the tablet-based applications.

D. Objectives

Having all these in mind, we are proposing to use natural language as a sole way to interact with the swarm because it is one of the most common ways for humans to communicate. If a human wants to ask robots to move an object to a certain location, the human can simply say "bring me a cup of coffee". For human, such commands based on natural language have very smooth learning curve. Instead of using natural language, the human can also use an explicit command to achieve the same outcome:

- MoveObject(Coffee_X, 1, 5, 8, 1.5)

In the example above, "Coffee_X" represented the object the human required. The second parameter "1" specified the quantify of the object. The last three parameters referred to the human user's coordinate which could be understood by the robots.

Clearly, learning and using the explicit commands takes more efforts. From a sentence with 6 words to a series of explicit commands, natural language processing technology could be the bridge where sophisticated technology meets smooth user experience.

E. Motivation

F. Objectives

G. Problem Statement

II. Literature Review

A. Available Technologies in Human-Swarm Interaction

Since Human-Swarm Interaction (HSI) is relatively a new research domain, to our knowledge, there is not much work based on real-life agents in this field. Authors in [6] propose

Roman Ibrahimov is with the Smart Machine and Assistive Robotics Technology (SMART) Laboratory, Department of Computer and Information Technology, Purdue University, West Lafayette, IN 47907, USA (e-mail: ibrahir@purdue.edu).

Yifei Hu is with the Applied Knowledge Representation and Natural Language Understanding (AKRaNLU) Laboratory, Department of Computer and Information Technology, Purdue University, West Lafayette, IN 47907, USA (e-mail: hu381@purdue.edu).

a mixed-granularity interface for multi-human-multirobot interaction. The interface is based on a networked augmented reality application that allows the operators to visualize and modify the global state of the system collaboratively on common tablets and smartphones. In this case, the only channel of interaction with the swarm is a table or smartphone. In a similar work [7], based on time-varying density functions, the operator touches tablet screen to design densities to manipulate the swarm as a whole. The user user sees color-changing visual feedback on the agents and on the screen.

B. Natural Language Processing in Human-Robot Interaction

Natural language is the default communication method between humans. Through natural language, we can convey either simple or extremely complicated information without special training since language has been the most common skill for everyone.

Matuszek et al. [8] proposed a statictical machine learning based method which translated English into robot control language (RCL). They trained their models on two maps with routes described in English and RCL and tested the models on two new maps. The models achieved 71.8% F1 score on the parsing task.

Matson et al. [9] used ontological semantic technology (OST) [10] to build a human-robot communication model. The explicit commands were extracted through natural language inputs using Text Meaning Representations (TMRs). The model was adaptive and required minimal human interference.

The literature showed various methods for human-robot interaction with natural language technology. However, to our best knowledge, there are not many research related to natural-language based human-swarm interaction system.

III. Proposed Approach

A. System Architecture

The overall system (Fig.1) is based on Robot Operating System 2 (ROS2). As the project contains simulation of multiple robots, ROS2 is an ideal tool that supports multiple robots per ROS network and it has ”Master”-less system with nodes capable of self-discovery.

Various voice commands are received from a Python script which runs NLP algorithm. Given the voice commands, the system triggers velocity commands based on consensus algorithm. The input to the algorithm is the coordinates of each individual robot. Then, the commands are sent to the controllers of the robots and the robots move to a certain point in the plane.

B. Receiving User's Input

This project used Python language for the back-end algorithms. However, Python did not have libraries that can directly communicate with user's microphone devices (PyAudio has not longer been available for the newer Python versions). We had to choose different programming languages to build a user interface and pass the voice command to the back-end Python algorithms. The users might want to communicate

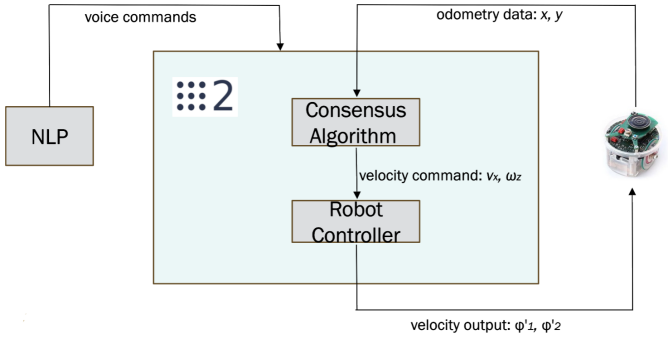

Fig. 1. An architecture depicting the workflow of the system.

with the swarm on various devices (smartphone, tablet, or PC) running different operating system (Mac OS, Windows, Android, or iOS). Therefore, the compatibility of the front-end (user interface) has become our top concern. To build a user interface with great compatibility within a relatively short time period (about 1-3 weeks), we finally decided to choose HTML and Javascript because any devices or systems should be able to use web applications.

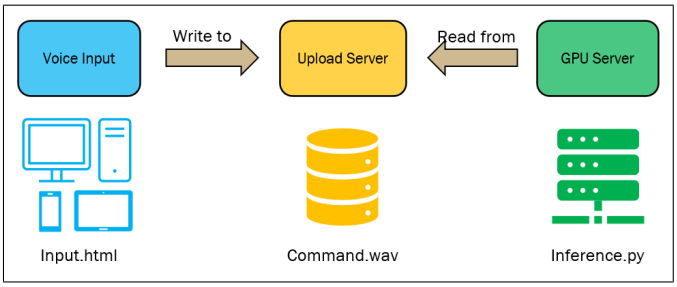

Fig. 2. Communication between front-end and back-end

Figure above showed the brief communication structure between the front-end and the back-end. The majority of the browsers restricted Cross Domain Access so that we were not allowed to directly send the recorded audio file to the remote GPU server (to run inference algorithms). To tackle this issue, we first stored the audio file to a directory (Upload Server) where both the input device and the GPU server had access to. Then we created a loop on the GPU server which kept scanning the directory and detecting any new WAV files (user's input commands). If a new WAV file was detected, the GPU server would run subsequent algorithms on the WAV file and finally convert the input voice command into certain swarm actions.

C. Translating User's Voice

The user's input was an audio (WAV) file. The swarm system only understood explicit functions which were predefined. The gap between the input and output was fulfilled using NLP technology. Figure below showed detailed steps in the NLP module.

When the GPU server detected a new WAV file in the Upload Server, it sent the WAV file to the Automatic Speech Recognition (ASR) layer. We deployed a speech recognition library (SpeechRecognition) in the ASR layer which would

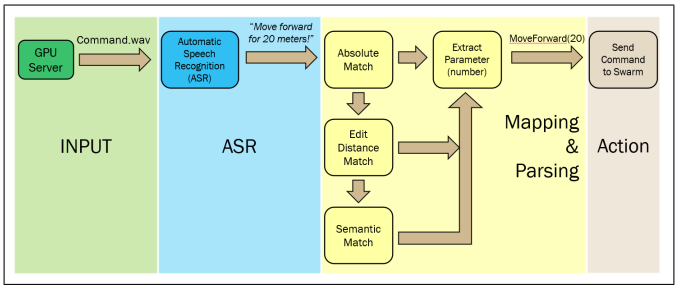

Fig. 3. Structure of the NLP module

convert the voice command into text format. The library used Google Cloud Speech API which (according to Google) provided the State-of-the-art accuracy and supported 125 languages.

The return value from the ASR layer would be a string. We needed to map the string to one of the existing functions and extract any parameters inside the string.

For each pre-defined function, we also provide at least one sample natural language form and specify whether the function required any parameters.

TABLE I
NUMBER OF SIMILAR WORDS WITH DIFFERENT EDIT DISTANCE (ED)

Function	Text Form	Param.
TurnLeft()	Turn left	Nullable
TurnLeft()	Make a left turn	Nullable
TurnLeft()	Turn left for N degrees	Not Null
TurnRight()	Turn right	Nullable
...	...	...
MoveForward()	Move forward for 20 meters	Not Null

Table above showed some sample functions and their text form(s). For each function, there could be multiple text forms to increase the success rate of the mapping process. The mapping process had 3 steps: absolute match, soft (edit distance) match, and semantic match.

Absolute match happened when the input text was exactly the same as one of the text forms. In this case, the input sentence would be directly passed to the next step without going through any sophisticated algorithms. If the users were well trained at the beginning, they would use mostly the commands already defined in as text forms. However, the advantage of using natural language to interact with swarm was that users did not need any special training. Therefore, the input commands would possibly fail to absolutely match any text forms and went to the next step: soft match.

Soft match used the edit distance algorithm [11] to mitigate the potential impact of cases, spellings, singular/plural, tense, and other minor issues. The edit distance algorithm calculated the character-level difference between 2 given strings and returned a number to quantitatively describe the difference. The soft match step could solve the following situation:

- Input sentence: make left turn.
- function text form: make a left turn.

Sometimes the ASR layer could make mistakes translating voice into text (for instance, missing a word). In such cases, if the input sentence and one of the text form had an edit distance smaller than a certain threshold, we could still consider it a match. The edit distance algorithm also did not required much computation power thus could be executed instantly.

If both the absolute and soft matches failed, the input sentence would go through the **semantic match** which was the most sophisticated algorithm in this module. The semantic match step used a pre-trained BERT language model [12] to compare the semantic similarity of two given sentences. The language model would return a percentage to describe the semantic similarity. We compared the input sentence to every single function's text form(s) and considered the one with the highest semantic similarity to be the correct match.

Running the language model requires much more computation power than other steps, which was the reason we made it the last option for matching the input sentence to a function. Latency has been always the concern for human-robot communication and running the language model would add latency to the whole system.

After the input sentence being matched to one of the pre-defined functions, we also needed to extract the parameters from the input sentence because some of the functions required certain parameters. In this case, we used the Stanza [13] library to parse the sentence and extract any numbers in that sentence. Currently, we only assumed that the input sentence would contain at least one number and that number would be the parameter (in our expected unit).

Finally, we sent the function and parameters to the robotics controller and triggered certain actions. By the end of this project, we only defined five functions. However, new functions can always be added by typing in their text forms and specifying whether parameters are needed. The NLP module was designed to be extendable.

D. Consensus Algorithm

We are going to consider our swarm as a networked multi-agent system $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ which has a set of nodes $\mathcal{N} = \{1, 2, ..., n\}$ according to Figure **??**. $\mathcal{E}$ is a set of edges and an unordered par of distinct nodes is denoted as $\{i, j\} \in \mathcal{E}$, whereas $\mathcal{N}_i = \{j | \{i, j\} \in \mathcal{E}\}$ stands for the set of neigboring nodes i.

In the system, each agent has its initial scalar value $x_i(0) \in \mathcal{R}$ and initial values of the each node in the form of vector is denoted as $x(0) = (x_1(0), x_2(0)..., x_n(0))$.

Let x_i stand for the heading direction or the information state of vehicle i. In this case, the dynamics of the consensus is as follows:

$$x_i(t+1) = \sum_{i=1}^{N} w_{ij} x_j(t) \tag{1}$$

where N is the set of vehicle i's neighbors including itself. Later, the consensus problem states that we have to choose a control law w_{ij} at each time constant t. In this case, the information state of $x_i(t)$ of the all agents should converge to the same value:

$$x_1(t) = x_2(t) = ... = x_m(t) = x^* \tag{2}$$

where x^* is a consensus point. There is no communication between agent *i* and *j* if w_{ij} = *0*. If the value of w_{ij} is non-zero, it means that there is an arc between vertexes *j* and *i*. In this case, the communication model for the network is a graph that links the agents. Thus, the problem is to determine the right values of w_{ij} such that the system converges to a global consensus.

E. Randezvous Problem

The goal of randezvous algorithm is to allow a collection of robots to meet at at the centroid of their initial positions. The algorithm is actually formulated based on consensus algorithm.

According to the algorithm, we have to model a single-integrator dynamics $\dot{x}_i = u_i$ whereas the control input to the each agent i would be $u_i \in \mathbb{R}^2$. The control input should be designed in a way that:

$$\lim_{t \to \infty} (x_i - x_j) = 0, \forall i, j = 1, ..., N \tag{3}$$

u_i can be defined as $u_i = \sum_{j \in N_i}(x_j - x_i) \Rightarrow \dot{x}_i = \sum_{j \in N_i}(x_j - x_i)$. If we consider the above-mentioned undirected graph structure $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, network dynamics can be formulated as $x = [x_{1,1}x_{2,1}...x_{N,1}x_{1,2}x_{2,2}...x_{N,2}]^T$ from which $\dot{x} = -(I \otimes L)x$. L is a graph Laplacian for the graph $\mathcal{G}$ and $L = D - A$ where D and A are adjacency and degree matrices respectiveley. Based on the properties of L we can show that the robots final position would be at the average of the initial conditions.

IV. Results and Analysis

The proposed approach was applied in two scenarios: a) simulation and b) real robot.

A. Simulation in Webots

In the Webots simulation environment, we created an squared arena with four E-puck robots at the corners and one ball at the center (Fig. 5). The objective was to move the ball out of the arena.

This task was designed for a multi-agent system because one single E-puck robot could not push the ball following a direct line. When the ball tended to leave the designated route, other E-puck robots must push it back.

In this simulation, we first gave a command by saying "Move the object." The voice command went through the NLP pipeline and was converted into a pre-defined function: Move(object_1, -1, -1). In this function, "object_1" was the ball and "-1, -1" represented the coordinate of the target location. The four E-puck robots first came to the center According to the Randezvous problem, the meeting point of the robots would be the center of arena in which the ball is located. When the above-mentioned voice command is received, the robots come to the center around the ball. The movement of the robots towards the center is stopped within a certain threshold to be able to manipulate the ball. The all the robots get the same heading direction and remove the ball from the center of the arena.

B. Experiment with a real drone

We also tested the proposed approach on a real hardware. Based on the voice commands, a nano-quadcopter delivers a remote object to a user in Fig. 4.

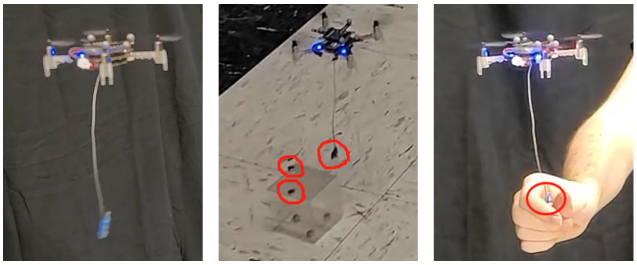

Fig. 4. A nano-quadcopter picks and delivers a remote object to a user.

V. Conclusion

This project demonstrated the possibility to build a human-swarm interaction system based on natural language commands. We tested the consensus algorithm and various NLP models in both the simulator and the real world. Within the limited schedule, we also proposed a unique method to build stable communication between Javascript and Python.

We are also fully aware of the limitations of this project. The communication between the front-end and the back-end could have noticeable latency which was introduced by the system structure. The swarm was only able to work together as one and the users were not allowed to give commands to any specific agents. Our algorithms worked flawlessly in the simulation. However, we didn't solve the hardware issues in the real world experiments and failed to test the swarm in the real world.

This project is only the beginning of exploring a young domain with great potentials. More works shall be conducted in the future.

VI. Future Works

A. Dealing with fuzzy expression and units

In the current NLP pipeline, we could not convert fuzzy expressions into explicit values. Fuzzy information is very common in natural language. Below is an example of fuzzy Vs. explicit expression:

- Move forward a little bit.
- Move forward for 2 meters.

Interpreting fuzzy expression (like "a little bit") is still a challenge in the NLP domain. However, the language would not be "natural" if there is no fuzzy information.

Another issue we encountered was to convert different input unit into the unit that the swarm understood. The swarm only used meter as the unit for distance, but the input could contain other units which could cause troubles. In the future, we might be able to create a rule-based unit-conversion system to tackle this issue.

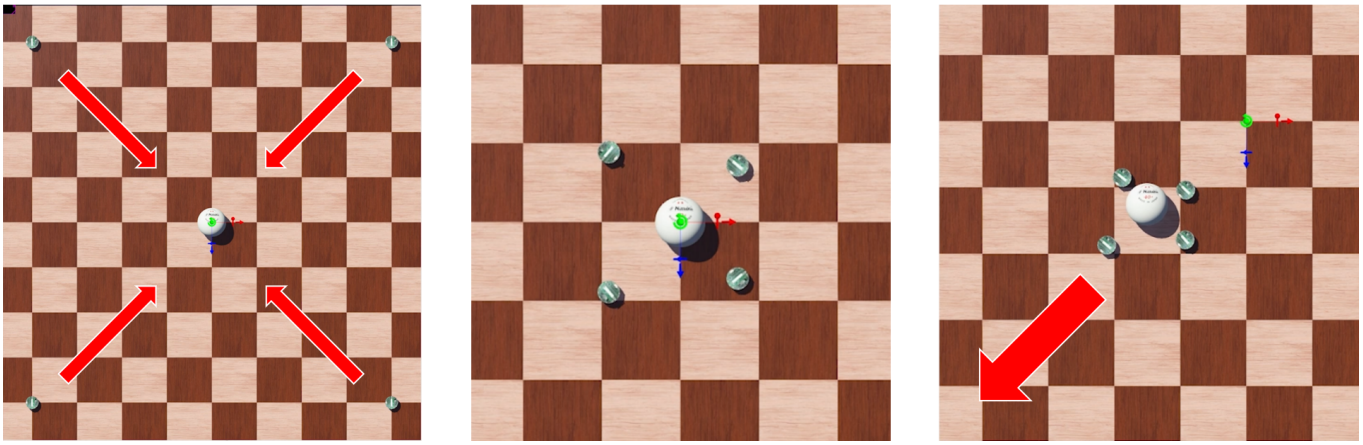

Fig. 5. Object manipulation in simulation: (left) the agents come to a meeting point in the center, (middle) agents stop around the object, and (right) agents move the object from the center

B. Bi-directional natural language interaction

Normally, communication came both ways. In our system, we could send commands to the swarm but could not receive any feed back from the swarm. The next step for us to explore is to display the sensor data (and other output data) from the swarm at the front-end user interface. It would be better if we can also translate the sensor data to natural language and make them more easy to understand by untrained users.

C. Modified drone hardware

The current Crazyflie 2.1 platform that is used in the project has some limitations such maximum weight-lift capacity of 10g. It, in turn, makes it difficult to manipulate bigger objects that are heavier than 10g. Having it in mind, we plan to build a bigger quadcopter based on the same platform using BigQuad deck, which enables lifting bigger objects. Furthermore, we plan to add a controllable magnetic grabber and in this case, it would be possible to pick and leave objects in a desirable positions.

VII. Individual Contributions

Table below showed the individual contribution for Yifei Hu and Roman Ibrahimov in this project.

TABLE II
INDIVIDUAL CONTRIBUTION

Tasks	Yifei	Roman
Project Planning	50%	50%
Literature Review	50%	50%
System Design	50%	50%
Swarm algorithm implementation	10%	90%
NLP algorithm implementation	90%	10%
Software integration	50%	50%
Hardware implementation	1%	99%
Blog Maintenance	80%	20%

References

[1] MA Subhan, AS Bhide, and COEB SSGB. Study of unmanned vehicle (robot) for coal mines. *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, 1(10):116–120, 2014.

[2] Daniel Carrillo-Zapata, Emma Milner, Julian Hird, Georgios Tzoumas, Paul J Vardanega, Mahesh Sooriyabandara, Manuel Giuliani, Alan FT Winfield, and Sabine Hauert. Mutual shaping in swarm robotics: user studies in fire and rescue, storage organization, and bridge inspection. *Frontiers in Robotics and AI*, 7, 2020.

[3] Jim Pugh and Alcherio Martinoli. Inspiring and modeling multi-robot search with particle swarm optimization. In *2007 IEEE Swarm Intelligence Symposium*, pages 332–339. IEEE, 2007.

[4] Dario Izzo and Lorenzo Pettazzi. Autonomous and distributed motion planning for satellite swarm. *Journal of Guidance, Control, and Dynamics*, 30(2):449–459, 2007.

[5] Andreas Kolling, Phillip Walker, Nilanjan Chakraborty, Katia Sycara, and Michael Lewis. Human interaction with robot swarms: A survey. *IEEE Transactions on Human-Machine Systems*, 46(1):9–26, 2015.

[6] Jayam Patel and Carlo Pinciroli. Improving human performance using mixed granularity of control in multi-human multi-robot interaction. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1135–1142. IEEE, 2019.

[7] Yancy Diaz-Mercado, Sung G Lee, and Magnus Egerstedt. Human–swarm interactions via coverage of time-varying densities. *Trends in Control and Decision-Making for Human–Robot Collaboration Systems*, pages 357–385, 2017.

[8] Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In *Experimental robotics*, pages 403–415. Springer, 2013.

[9] Eric T Matson, Julia Taylor, Victor Raskin, Byung-Cheol Min, and E Cho Wilson. A natural language exchange model for enabling human, agent, robot and machine interaction. In *The 5th International Conference on Automation, Robotics and Applications*, pages 340–345. IEEE, 2011.

[10] Sergei Nirenburg and Victor Raskin. *Ontological semantics*. Mit Press, 2004.

[11] Fred J Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*, 2020.