ISSN 2712-0562
**Sustainable Engineering and Innovation**
*Original Research*
Vol. 6, No. 2, October 2024, pp.199-212
https://doi.org/10.37868/sei.v6i2.id309

# Design and implementation of voice-command controller for fixed-wing unmanned aerial vehicles using automatic speech recognition and natural language processing techniques

**Cherry Mae G. Villame[1*], Sherwin A. Guirnaldo[2]**
[1*]Department of Computer Engineering and Mechatronics (DCEM), MSUIIT, Philippines
[2]Department of Mechanical Engineering and Technology (DMET), MSUIIT, Philippines

[*]cherrymae.galangque@g.msuiit.edu.ph, [2]sherwin.guirnaldo@g.msuiit.edu.ph

Received Jul. 1, 2024
Revised Oct. 3, 2024
Accepted Oct. 10, 2024

**Abstract**

This paper explores the development of a voice command controller leveraging the capabilities of an automatic speech recognition (ASR) system and natural language processing (NLP) technique to manage a fixed-wing unmanned aerial vehicle (UAV). The controller is designed to interpret voice commands for controlling fixed-wing UAVs. The implementation of the system involved two key stages: (1) implementation of a voice command controller using integrated ASR and NLP techniques deployed in a simulated plane in the SITL simulator followed by (2) deployment of the controller to an actual Sky Surfer plane fixed-wing aircraft. The results indicate that the algorithm achieved an average confidence rate of 91.86 % in transcribing voice commands to words, with a Word Error Rate (WER) of approximately 0.021. The developed system demonstrated the ability to interpret both low-level and high-level commands for UAV control interfaces. Such an interface offers greater intuitiveness compared to traditional RC controls, potentially requiring less training to operate effectively. Moreover, it reduces human workload, as once commands are issued, the system can execute them without the need for continuous supervision.



*Keywords*: Speech recognition system, Natural language processing, Fixed-wing UAV, Dronekit, Mavproxy

## 1. Introduction

Unmanned aerial vehicles (UAVs) are increasingly valuable in military, commercial, and scientific realms [1]. Fixed-wing UAVs play a pivotal role in modern military operations, enhancing situational awareness, conducting intelligence, surveillance, and reconnaissance (ISR), and supporting various missions on the battlefield. Their adaptability, endurance, and ability to operate in hostile environments render them indispensable assets for militaries globally. Compared to drones, fixed-wing UAVs typically boast longer flight times and higher maximum speeds, making them ideal for covering extensive areas and executing missions requiring endurance and speed, such as reconnaissance and surveillance.

The control surfaces of fixed-wing UAVs are usually maneuvered by servo motors, controlled by an onboard flight controller. Pilots utilize a transmitter (remote control) equipped with a joystick or similar input device to relay commands to the flight controller, which then adjusts the position of the control surfaces to achieve the desired flight trajectory. However, for pilots, the conventional method of using a joystick and manually inputting waypoints into a guidance and control system during flight planning for fixed-wing UAVs is time-consuming [2]. Each waypoint necessitates scrolling through the alphabet using a knob, demanding over a minute of the pilot's focused attention. This process can potentially divert the pilot's attention from critical tasks like monitoring gauges, posing safety risks. Additionally, novice pilots encounter navigational challenges with fixed-wing UAVs, as controlling them via RC devices lacks a natural, intuitive interface, often leading to difficulties in mastering them [3]. Novice pilots commonly struggle with managing trims, over-controlling the aircraft, misjudging altitudes, and making hasty decisions. In military applications, drone/plane pilots face difficulties in controlling UAVs due to the burdensome nature of handling stick controls. In emergency situations, pilots cannot afford to be overwhelmed with multiple control tasks, necessitating a more streamlined approach. Furthermore, the use of fully manual control planes during ambushes or attacks can compromise safety, as both military personnel and pilots must prioritize concealment. Recognizing the potential for guiding aircraft using verbal commands, automating UAV control through speech commands allows pilots to execute tasks independently, enhancing safety. Incorporating speech recognition into fixed-wing UAVs [4] could significantly improve safety by enabling pilots to verbally input waypoints into the guidance and control system instead of manually adjusting knobs or selecting letters [5]. However, two primary challenges must be addressed: developing a robust speech recognition system and designing it to interface seamlessly with the aircraft's control system. Thus, this paper introduces a speech recognition engine integrated with natural language processing techniques for controlling fixed-wing unmanned aerial vehicles. This system serves as a controller capable of interpreting voice commands for fixed-wing UAV controls.

## 2. Research method

### 2.1. Flying platform

The aircraft chosen for this study is a Sky Surfer fixed-wing UAV as shown in Figure 1, renowned for its simplified structure and aerodynamic efficiency, enabling extended flight durations and higher speeds. Fixed-wing aircraft offer distinct advantages in terms of endurance and speed compared to other types of UAVs. Table 1 presents the specifications of the Sky Surfer fixed-wing UAV used in the study.

Figure 1. Sky Surfer fixed-wing unmanned aerial vehicle

Table 1. Skysurfer fixed-wing UAV specifications

| Command | Component |
|---|---|
| Wingspan | 1400 mm |
| Length | 925 mm |
| Flying weight | 625 g |
| Drive system | 2620 Brushless Outrunner motor |
| Servo motors | 4X 9g high speed micro servos |
| ESC | 20A Brushless speed controller |
| Battery | 11.1V ,3S 1300mAh 20C Li-polymer |

### 2.2. Overall system structure

Figure 2 shows the overall system structure. The main uses of the MAVProxy ground control station are (1) to create a MAVLINK communication bridge between the Dronekit script into the plane via UDP and telemetry, (2) to display the behavior of the plane using Mission planner via UDP and (3) assist with the interactive command of the pilot to the plane to travel to a target location. MAVProxy serves as the fully functional cross-platform portable ground control station software of the study. It was used to complement Mission Planner, which is another GCS that provides a graphical user interface to the pilot in monitoring the flights of the plane. A microphone from the laptop was used to listen to the pilot's command.

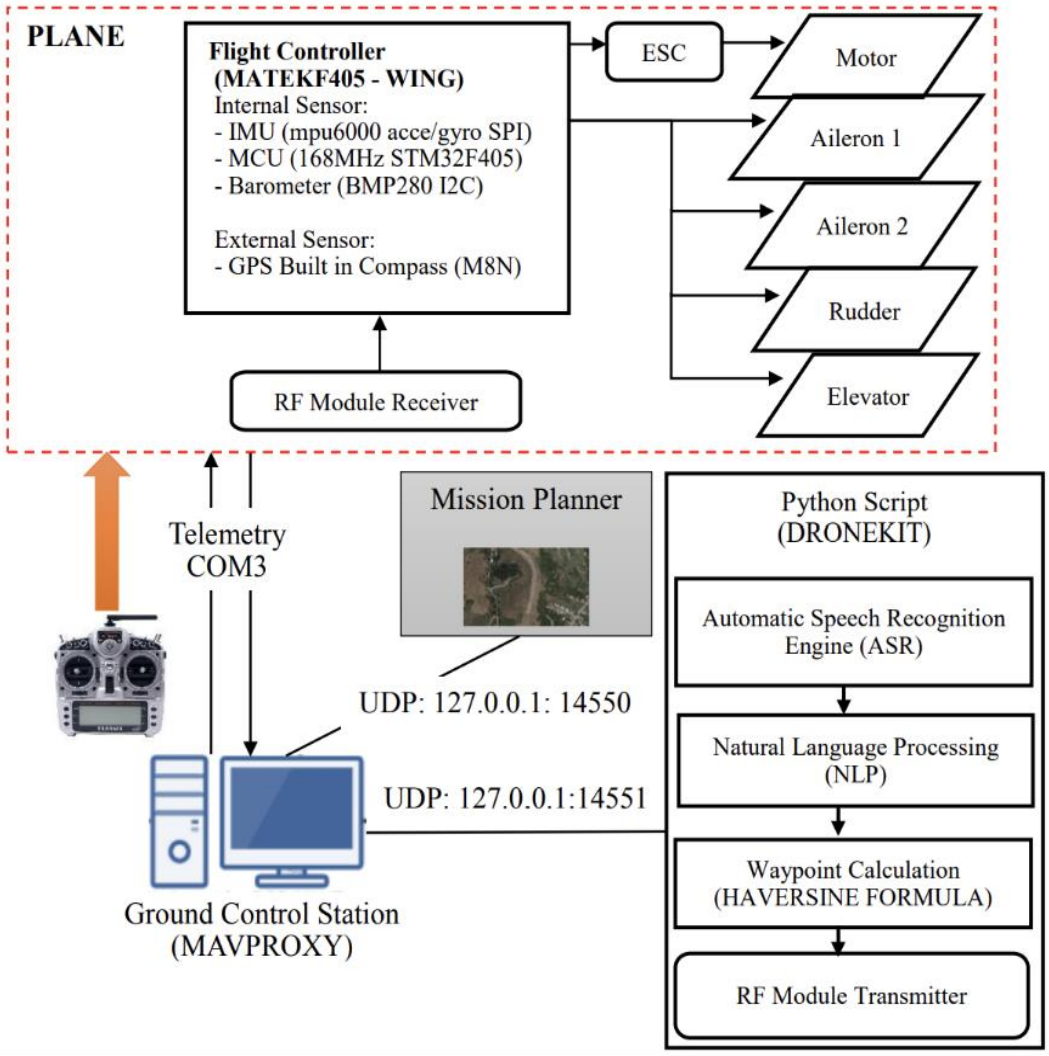


Figure 2. Overall system structure

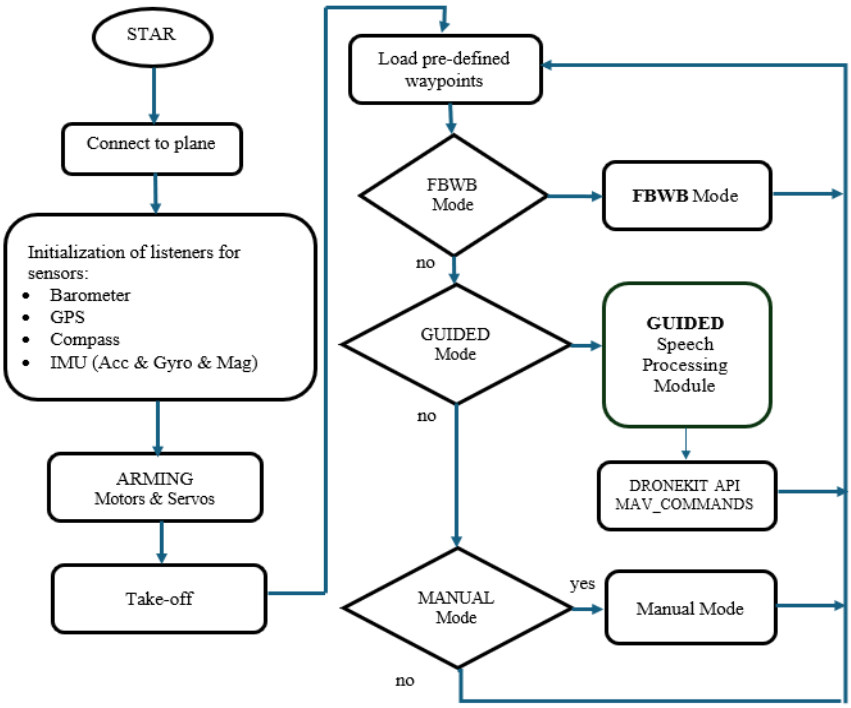


Figure 3. Ground station app flow diagram

When a command is heard, these voice commands are processed in the ASR and NLP module, translating these voice commands into waypoints using the haversine formula into the flight controller via a transmitter-receiver module for UAV navigational control. Once the Dronekit script is connected to the plane it will switch from manual mode to Guided mode. In guided mode, the vehicle can be able to respond in real-time to new tasks. A guided mission allows the UAV to respond to new waypoints that can arrive one after the other and will execute as long as it arrives before the previous one completes. This means that waypoints can be sent from a remote location in this case the ground control station. Waypoints in latitude and longitude which is the result of the Speech Processing Module will be sent together with the Dronekit API commands to the flight controller. The code was written in Python using the Dronekit API which transcodes instructions into MAVLink messages that are then forwarded through MAVProxy to send to the flight controller via telemetry to intelligently instruct the UAV. Figure 3 shows the process flow of the ground station application.

### 2.3. Automatic speech recognition engine (ASR) and natural language processing (NLP) pipeline

Google Cloud Speech API as shown in Figure 4 was used for the implementation of the ASR engine of the study. Google Cloud Speech API enables the developers to turn audio into text by applying neural network models easily using the API. The API can recognize more than 110 languages and variants, to support a global user base since it was implemented with multiple machine-learning models for increased accuracy. After listening to the voice command, the app sends the audio data to the speech-to-text API and initiates a long-running operation. With this operation, the app can periodically poll for recognition results.

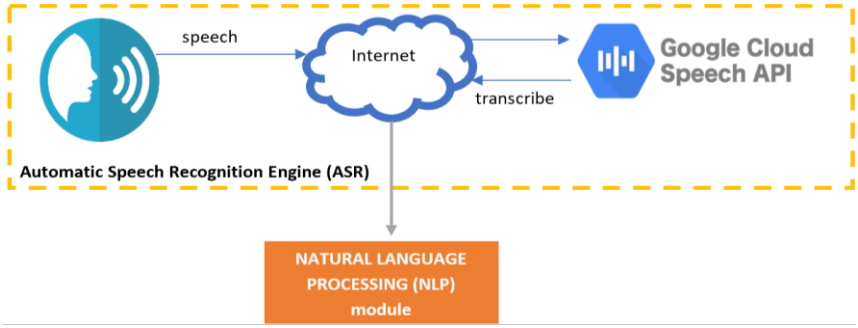


Figure 4. Automatic speech recognition engine module

The development of the NLP system is complicated; thus, the fundamental NLP pipeline was broken down into subtasks. The pipeline starts with sentence segmentation wherein the text output from the developed speech recognition engine is broken down into separate sentences or words. This was followed by word tokenization wherein sentences are split apart whenever there's a space between them and also treat punctuation marks as separate tokens followed by POS tagging. POS tagging is a process of identifying what part of speech a token belongs to, whether it is a noun, a verb, an adjective, and so on. After determining a token's tag, it will be followed by the identification of stop words. Stop words are common words that are unnecessary in the commands. Removing these words will not change the meaning of the corpus and it will lead to better results because the remaining words will be the most important to determining the meaning of the corpus. After eliminating the stop words, the remaining words were then evaluated for their meaning.

UAV control commands were assigned to every possible meaning of the words present in the commands. Figure 5 shows the NLP flow diagram of the system. The text output from the ASR system is passed to the NLP function for conversion into meaningful UAV control commands. The output then will be evaluated and compared on the dictionary created for checking. The following accepted commands and their equivalent actions are presented in Table 2 and Figure 6 respectively.

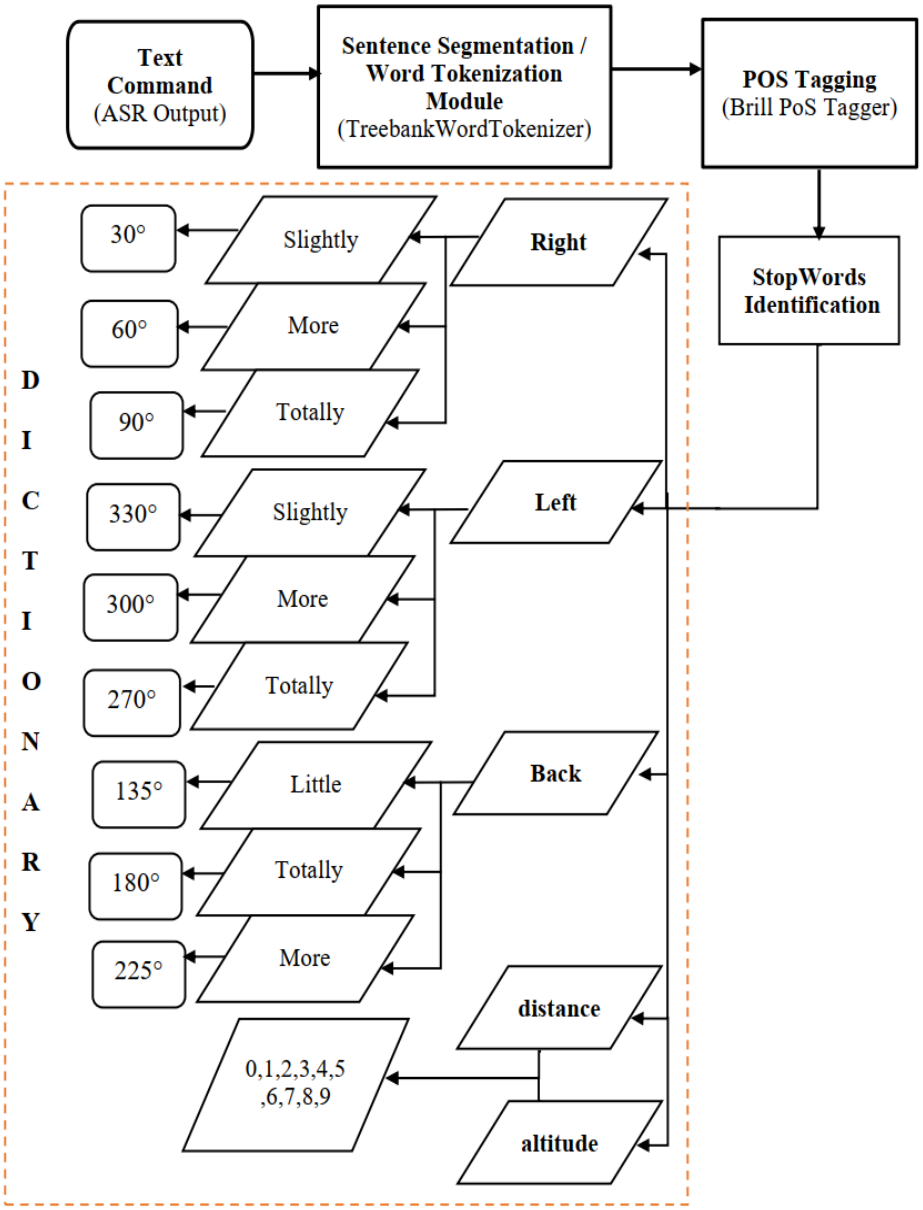


Figure 5. Natural language processing module flow diagram

Table 2. UAV command representation

| Command | Component |
|---|---|
| Slightly right | Plane will head to direction with bearing equal to 30° |
| More right | Plane will head to direction with bearing equal to 60° |
| Totally right | Plane will head to direction with bearing equal to 90° |
| Slightly left | Plane will head to direction with bearing equal to 330° |
| More left | Plane will head to direction with bearing equal to 300° |
| Totally left | Plane will head to direction with bearing equal to 270° |
| Little back | Plane will head to direction with bearing equal to 135° |
| More back | Plane will head to direction with bearing equal to 225° |
| Totally back | Plane will head to direction with bearing equal to 180° |
| Distance | Indicates the distance between previous waypoint to the next waypoint |
| Altitude | Indicates the altitude the plane should maintain upon arrival of the target destination |
| 0,1,2,3,4,5,6,7,8,9 | Numerical value provided after issuance of 'distance' and 'altitude' voice commands |

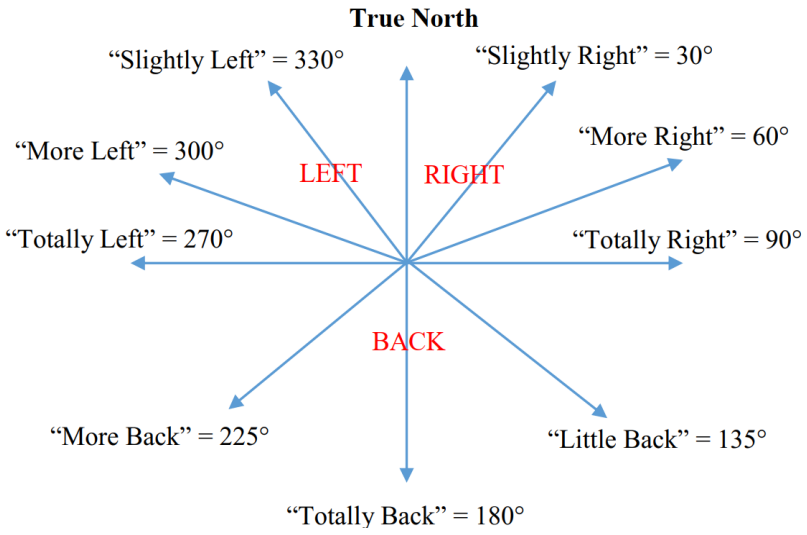


Figure 6. Command's equivalent graphical action

The results from the Natural Language Processing (NLP) module, specifically bearing, distance, and altitude, serve as inputs for determining the subsequent waypoint. The proponent utilizes the haversine formula, a fundamental equation in navigation, to compute the great-circle distances between two points on a sphere based on their respective longitudes and latitudes. Equations (1) and (2) depict the application of the haversine formula for computing the latitude and longitude coordinates of the next waypoint.

$$\varphi_2 = A \sin (\sin \varphi_1 * \cos \beta + \cos \varphi_1 * \sin \beta * \cos \theta) \quad (1)$$

$$\lambda_2 = \lambda_1 + A \tan^2 (\sin \theta * \sin \beta * \cos \varphi_1, \cos \beta - \sin \varphi_1 * \sin \varphi_2) \quad (2)$$

Where $\varphi_1$ is the latitude of the previous waypoint, $\lambda_1$ is the longitude of the previous waypoint, $\theta$ is the bearing (clockwise from north), $\beta$ is the angular distance $d/R$ and $d$ is the distance traveled by the plane, $R$ = 6378.1 the earth's radius

## 3. Results and discussion

This section shows the success of utilizing automated speech recognition engine (ASR) and natural language processing (NLP) techniques to translate text commands into fixed-wing UAV controls. The fixed-wing UAV chosen for this study is the Sky Surfer. This aircraft is constructed from solid lightweight Expanded Polyolefin (EPO) foam, providing a sturdy framework capable of supporting both the aircraft itself and its payload. Additionally, EPO foam is chosen for its safety features, as it minimizes the risk of damage in the event of unintentional hard landings. The Sky Surfer is equipped with a 3S Li-Po battery rated at 11.1V and 3600mAh, boasting a 25C discharge rating, ensuring ample power for its operations. With a full four-channel configuration, it allows independent control of aileron, elevator, rudder, and throttle. The aircraft is 95 percent assembled as ready-to-fly (RTF), requiring minimal setup, including the installation of the flight controller, servos, and the assembly of the wing and push-rod components for control surfaces. Figure 7 shows the Sky Surfer with the assembled flight controller, telemetry, vtx, and runcam.

Figure 7. The Sky Surfer with onboard peripherals

The MATEKF405-WING flight controller was placed inside the plane. The motor was powered through the ESC while servos were powered through the onboard battery eliminator circuit (BEC) which is an electronic circuit designed to deliver electrical power to other circuitry without the need for multiple batteries. The figure also shows how the external GPS with compass, camera, 3DR telemetry, and EACHINE video transmitter were mounted. Figure 8 shows how a sample voice command from the pilot was translated into the ASR Engine from

Google Cloud Speech API and fed it into the NLP module for processing, translating into an equivalent waypoint to be sent to the flight controller in the plane.

Two performance metrics were employed to assess the effectiveness of the algorithm that integrates automatic speech recognition (ASR) and natural language processing (NLP) techniques for voice recognition and translation. The initial approach utilizes confidence scores, a feature derived from the ASR decoder, to gauge the accuracy of ASR responses. The second method involves calculating the Word Error Rate (WER), which quantifies the total number of words deleted, inserted, or substituted relative to the expected total number of words. Equation 3 illustrates the computation process for determining the WER of the algorithm.

$$WER = (S + D + I) / H + S + D \quad (3)$$

Where I is the total number of insertions, D is the total number of deletions, S is the total number of substitutions, and H is the total number of hits.

Using Equation 3, the WER of the algorithm is $0.0207253886 \approx 0.021$. Along with confidence scores from testing multiple voice inputs, Table 2 shows that the algorithm had a total number of insertions I of 0, a total number of deletions D is 2, a total number of substitutions S is 2, and a total number of hits H is 189. Table 3 shows the confidence scores of the ASR /NLP Algorithm.

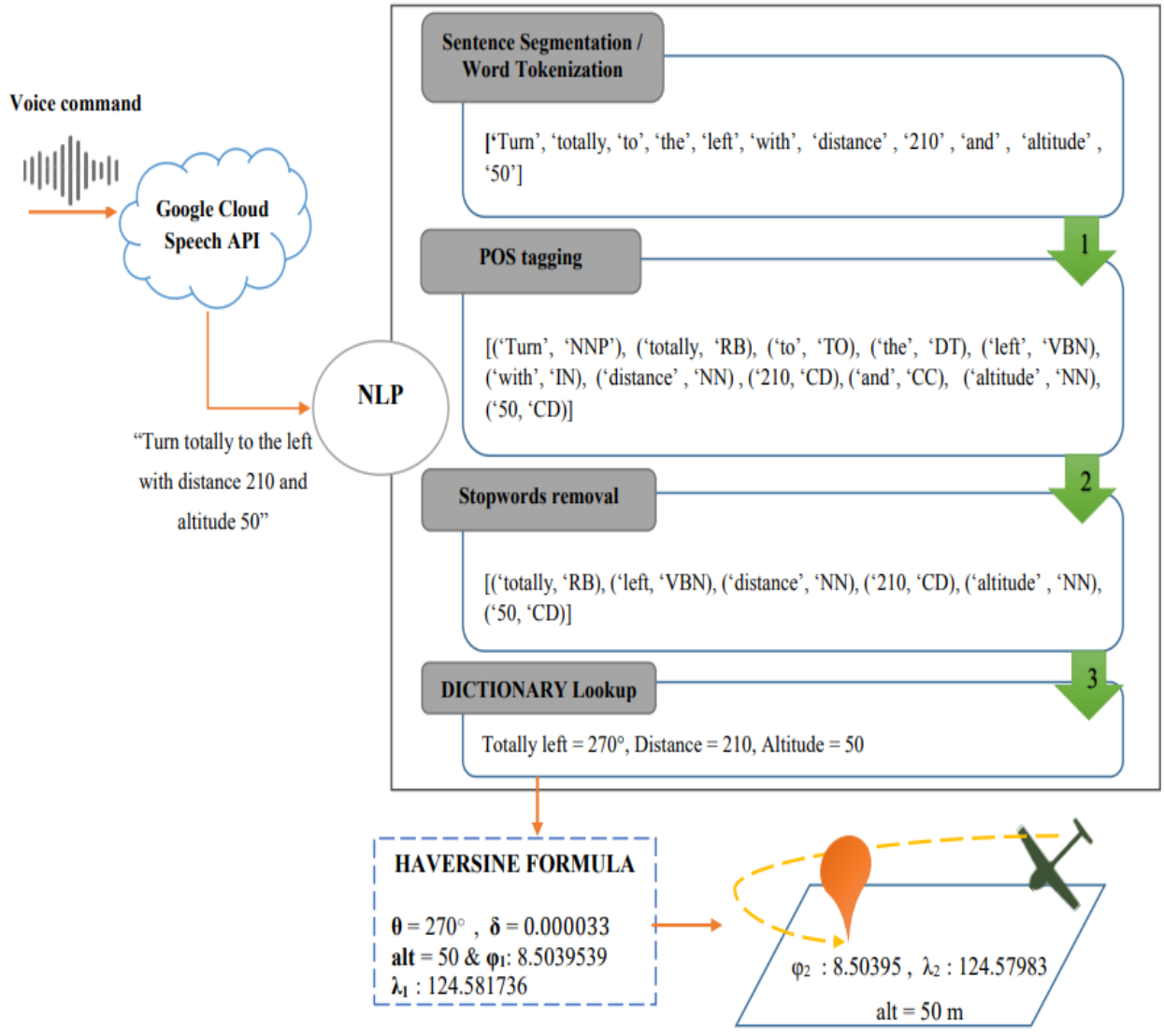


Figure 8. Sample waypoint calculation from voice command

Table 3. Confidence scores of the ASR /NLP algorithm

| Voice input | Output | Confidence rate |
|---|---|---|
| turn totally to the left with distance 150 and altitude 70 | turn totally to the left with distance 150 and altitude 70 | 92.90 % |
| turned slightly to the right with distance 310 and altitude 50 | turn slightly to the right with distance 310 and altitude 50 | 94.60 % |
| turn a little back with distance 350 and altitude 50 | turn a little back with distance 350 and altitude 50 | 78.80 % |
| turn totally back with distance 150 and altitude 30 | turn totally back with distance 150 and altitude 30 | 87.40 % |
| turn slightly to the left with distance 470 and altitude 30 | turn slightly to the left with distance 470 and altitude 30' | 94.80 % |
| turn more to the left with distance 250 and altitude 60 | turn more to the left with distance 250 and altitude 60 | 94.50 % |
| turn totally to the right with distance 90 and altitude 30 | turn totally to the right with distance 90 and altitude 30 | 92.50 % |
| turn totally back with distance 150 and altitude 30 | turn totally back with distance 150 and altitude 30 | 89.50 % |
| turn totally to the left with distance 90 and altitude 40 | turn totally to the left with distance 90 and altitude 40 | 94.80 % |
| turn slightly to the left with distance 100 and altitude 70 | turn slightly to the left with distance 180 270 | 84.20 % |
| turn more to the right with distance 170 and altitude 40 | turn more to the right with distance 170 and altitude 40 | 94.80 % |
| turn totally to the right with distance 90 and altitude 50 | turn totally to the right with distance 90 and altitude 50 | 94.80 % |
| turn little back with distance 350 and altitude 50 | turn little back with distance 350 and altitude 50 | 87.60 % |
| turn slightly to the left with distance 110 and altitude 70 | turn slightly to the left with distance 110 and altitude 70 | 94.70 % |
| turn more to the left with distance 120 and altitude 50 | turn more to the left with distance 120 and altitude 50 | 94.80 % |
| turn slightly to the right with distance 310 and altitude 50 | turn slightly to the right with distance 310 and altitude 50 | 92.60 % |
| turn totally back with distance 150 and altitude 30 | turn totally back with distance 150 and altitude 30 | 94.80 % |
| turn more to back with distance 470 and altitude 20 | turn more to back with distance 470 and altitude 20 | 92.70 % |
| Go Home | go home | 94.80 % |
| Average | | 91.90 % |

The flight controller assumes the responsibility of managing the UAV's control surfaces to guide it toward the designated target location. In testing the controller, the proponent first tests it in the SITL simulator before

deploying it in a real Sky Surfer fixed-wing UAV. The commands are executed to verify the efficacy of the implemented voice command controller within the Ground Control Station application in directing the flight plans of the UAV through the Software-In-The-Loop (SITL) simulator before executing it in a real plane.

Table 4 displays the 1st voice commands with the real Sky Surfer, the calculation of the next waypoint from these commands, and the execution of the plane as shown in Mission Planner illustrated in Figure 9.

Table 4. 1st command "turn totally to the left with distance 150 and altitude 70" voice command

| Previous waypoint | Voice command (ASR) | NLP output | Calculated next waypoint |
|---|---|---|---|
| $\varphi_1$: 8.5045<br>$\lambda_1$: 124.5817 | turn totally to the left with distance 150 and altitude 70 | Turning to 270 degrees with a distance of 150 meters and altitude of 70 meters.<br>$\theta = 270^\circ$<br>$\delta = 0.15$<br>alt = 70 | $\varphi_2$: 8.5045<br>$\lambda_2$: 124.5803<br>alt = 70 m |

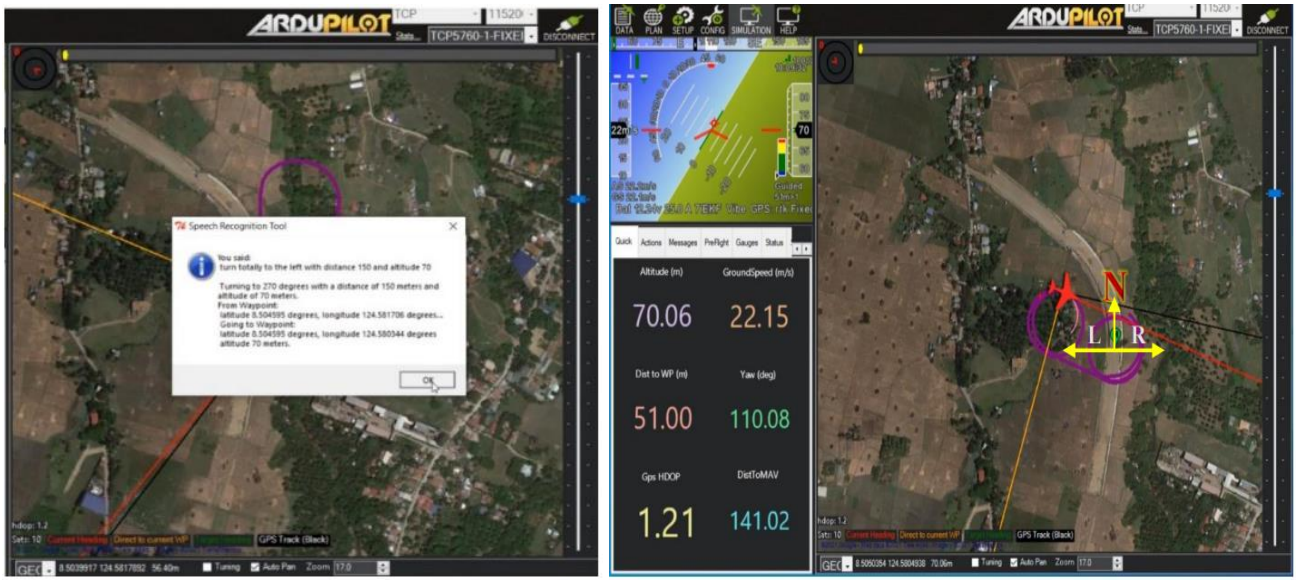


Figure 9. 1st command "Turning to 270 degrees with a distance of 150 meters and altitude of 70 meters." real-time message box monitoring and Sky Surfer plane execution

Table 5 shows sample 2nd voice commands with the real Sky Surfer, the calculation of next waypoint from these commands, and the execution of the plane in as shown in Mission Planner as illustrated in Figure 10

Table 5. 2nd command "turn slightly to the right with distance 310 and altitude 50" voice command

| Previous waypoint | Voice command (ASR) | NLP output | Calculated next waypoint |
|---|---|---|---|
| $\varphi_1$: 8.5045<br>$\lambda_1$: 124.5800 | Turn slightly to the right with distance 310 and altitude 50 | Turning to 30 degrees with a distance of 310 meters and altitude of 50 meters.<br>$\theta = 30^\circ$<br>$\delta = 0.31$<br>alt = 50 | $\varphi_2$: 8.5068906<br>$\lambda_2$: 124.58138<br>alt = 50 m |


Figure 10. 2nd command "Turning to 30 degrees with a distance of 310 meters and altitude of 50 meters" real-time message box monitoring and Sky Surfer plane execution

Table 6 shows sample 3rd voice commands with the real Sky Surfer, the calculation of the next waypoint from these commands, and the execution of the plane in as shown in Mission Planner as illustrated in Figure 11.

Table 6. 3rd command "turn little back with distance 350 and altitude 50" voice command

| Previous waypoint | Voice command (ASR) | NLP output | Calculated next waypoint |
|---|---|---|---|
| $\varphi_1$: 8.506<br>$\lambda_1$: 124.5820 | Turn little back with distance 350 and altitude 50 | Turning to 135 degrees with a distance of 350 meters and altitude of 50 meters.<br>$\theta = 135^\circ$<br>$\delta = 0.31$<br>alt = 50 | $\varphi_2$: 8.5042<br>$\lambda_2$: 124.5843<br>alt = 50 m |


Figure 11. 3rd command "Turning to 30 degrees with a distance of 310 meters and altitude of 50 meters" real-time message box monitoring and Sky Surfer plane execution

Table 7 shows sample 4th voice commands with the real Sky Surfer, the calculation of the next waypoint from these commands, and the execution of the plane as shown in Mission Planner in Figure 12.

Table 7. 4th command “go home” voice command

| Previous waypoint | Voice command (ASR) | NLP output | Calculated next waypoint |
|---|---|---|---|
| $\varphi_1$: 8.5042<br>$\lambda_1$: 124.5843 | Go home | going home<br>default alt = 20 | $\varphi_2$: 8.504164<br>$\lambda_2$: 124.58134<br>alt = 20 m |

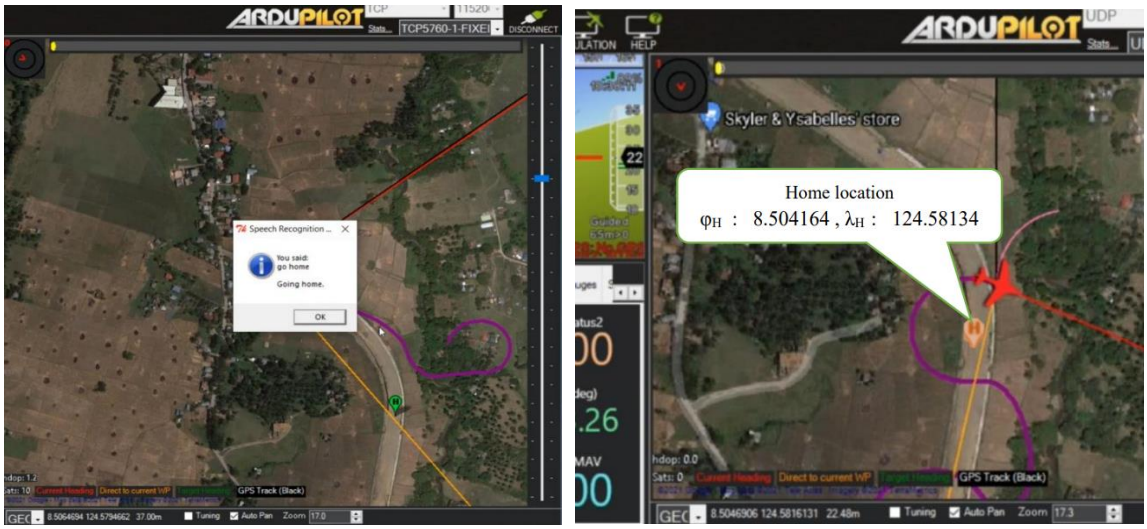


Figure 12. 4th command “go home” real-time message box monitoring and Sky Surfer plane execution

## 4. Conclusions

This study aimed to investigate the feasibility of employing voice commands to control the flight of a fixed-wing UAV. The system implementation had two stages: (1) development and testing of the voice command controller to control flight plans of a plane using the SITL simulator before (2) deploying it to control flight plans of a real fixed-wing plane. Based on the results the average confidence rate of the algorithm in transcribing voice to words was 91.86 % and the Word Error Rate (WER) is ≈ 0.021. Using the capabilities of the automatic speech recognition system (ASR) of Google Speech API and natural language processing (NLP), the implemented voice controller could translate voice commands into an equivalent waypoint and use this result together with DroneKit API to fly a fixed-wing UAV via MAVLink protocol. Although there are delays in the transmission of the commands because of the ASR and NLP processing, the results show that the system was capable of translating voice commands into controls for plane maneuvering and that the plane was able to fly to the desired location by the pilot.

### Declaration of competing interest

All authors declared no known financial or non-financial competing interests in any material discussed in this paper.

### Funding information

Funding was received from DOST-Engineering Research and Development for Technology (ERDT).

### Acknowledgments

Special acknowledgment is also extended to the individuals from the Center for Artificial Intelligence Research (CAIR), Engineering Research and Development for Technology (ERDT), and the Department of Computer Engineering and Mechatronics (DCEM) for their contributions to the project conceptualization and for providing the necessary premises and facilities essential for the study's development.

**Author contribution**

The contribution to the paper is as follows: Cherry Mae G. Villame: primary author; Sherwin A. Guirnaldo: research adviser. All authors approved the final version of the manuscript."

**References**

[1] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time delay neural networks," Technical report, ATR Interpreting Telephony Research Laboratory, Kyoto, Japan, 1987. [Online]. Available: https://doi.org/10.1121/1.2025362

[2] M. Quigley, M. A. Goodrich, and R. W. Beard, "Semiautonomous human-UAV interfaces for fixed-wing mini-UAVs," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3, pp. 2457-2462, 2004.

[3] S. S. Anand and R. Mathiyazaghan, "Design and fabrication of voice controlled unmanned aerial vehicle," *IAES International Journal of Robotics and Automation (IJRA)*, vol. 5, no. 3, 2016. [Online]. Available: https://doi.org/10.11591/ijra.v5i3.pp205-212

[4] B. A. Q. Al-Qatab and R. N. Ainon, "Arabic speech recognition using Hidden Markov Model Toolkit (HTK)," in *Proceedings of the IEEE*, 2010.

[5] P. Doherty, G. Granlund, K. Kuchcinski, E. Sandewall, K. Nordberg, E. Skarman, and J. Wiklund, "The WITAS unmanned aerial vehicle project," in *Proceedings ECAI*, 2000.

This page intentionally left blank.