Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

OUTDOOR OPERATIONS OF MULTIPLE QUADROTORS IN WINDY ENVIRONMENT

by
Deepan Lobo

Coordinated multiple small unmanned aerial vehicles (sUAVs) offer several advantages over a single sUAV platform. These advantages include improved task efficiency, reduced task completion time, improved fault tolerance, and higher task flexibility. However, their deployment in an outdoor environment is challenging due to the presence of wind gusts. The coordinated motion of a multi-sUAV system in the presence of wind disturbances is a challenging problem when considering collision avoidance (safety), scalability, and communication connectivity. Performing wind-agnostic motion planning for sUAVs may produce a sizeable cross-track error if the wind on the planned route leads to actuator saturation. In a multi-sUAV system, each sUAV has to locally counter the wind disturbance while maintaining the safety of the system. Such continuous manipulation of the control effort for multiple sUAVs under uncertain environmental conditions is computationally taxing and can lead to reduced efficiency and safety concerns. Additionally, modern day sUAV systems are susceptible to cyberattacks due to their use of commercial wireless communication infrastructure.

This dissertation aims to address these multi-faceted challenges related to the operation of outdoor rotor-based multi-sUAV systems. A comprehensive review of four representative techniques to measure and estimate wind speed and direction using rotor-based sUAVs is discussed. After developing a clear understanding of the role wind gusts play in quadrotor motion, two decentralized motion planners for a multi-quadrotor system are implemented and experimentally evaluated in the presence of wind disturbances. The first planner is rooted in the reinforcement

learning (RL) technique of state-action-reward-state-action (SARSA) to provide generalized path plans in the presence of wind disturbances. While this planner provides feasible trajectories for the quadrotors, it does not provide guarantees of collision avoidance. The second planner implements a receding horizon (RH) mixed-integer nonlinear programming (MINLP) model that is integrated with control barrier functions (CBFs) to guarantee collision-free transit of the multiple quadrotors in the presence of wind disturbances. Finally, a novel communication protocol using Ethereum blockchain-based smart contracts is presented to address the challenge of secure wireless communication.

The U.S. sUAV market is expected to be worth $92 Billion by 2030. The Association for Unmanned Vehicle Systems International (AUVSI) noted in its seminal economic report that UAVs would be responsible for creating 100,000 jobs by 2025 in the U.S. The rapid proliferation of drone technology in various applications has led to an increasing need for professionals skilled in sUAV piloting, designing, fabricating, repairing, and programming. Engineering educators have recognized this demand for certified sUAV professionals.

This dissertation aims to address this growing sUAV-market need by evaluating two active learning-based instructional approaches designed for undergraduate sUAV education. The two approaches leverages the interactive-constructive-active-passive (ICAP) framework of engagement and explores the use of Competition based Learning (CBL) and Project based Learning (PBL). The CBL approach is implemented through a drone building and piloting competition that featured 97 students from undergraduate and graduate programs at NJIT. The competition focused on 1) drone assembly, testing, and validation using commercial off-the-shelf (COTS) parts, 2) simulation of drone flight missions, and 3) manual and semi-autonomous drone piloting were implemented. The effective student learning experience from this competition served as the basis of a new undergraduate course on drone science

fundamentals at NJIT. This undergraduate course focused on the three foundational pillars of drone careers: 1) drone programming using Python, 2) designing and fabricating drones using Computer-Aided Design (CAD) and rapid prototyping, and 3) the US Federal Aviation Administration (FAA) Part 107 Commercial small Unmanned Aerial Vehicles (sUAVs) pilot test. Multiple assessment methods are applied to examine the students' gains in sUAV skills and knowledge and student attitudes towards an active learning-based approach for sUAV education. The use of active learning techniques to address these challenges lead to meaningful student engagement and positive gains in the learning outcomes as indicated by quantitative and qualitative assessments.

OUTDOOR OPERATIONS OF MULTIPLE QUADROTORS IN WINDY ENVIRONMENT

by
Deepan Lobo

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology – Newark
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering

Helen and John C. Hartmann Department of
Electrical and Computer Engineering

May 2022

APPROVAL PAGE

OUTDOOR OPERATIONS OF MULTIPLE QUADROTORS IN WINDY ENVIRONMENT

Deepan Lobo

Dr. Pramod Abichandani, Dissertation Advisor — Date
Assistant Professor, School of Applied Engineering and Technology
Department of Electrical and Computer Engineering, NJIT

Dr. Moshe Kam, Committee Member — Date
Dean of Newark College of Engineering,
Professor, Department of Electrical and Computer Engineering, NJIT

Dr. Mengchu Zhou, Committee Member — Date
Distinguished Professor, Department of Electrical and Computer Engineering,
NJIT

Dr. Cong Wang, Committee Member — Date
Associate Professor, Department of Electrical and Computer Engineering,
NJIT

Dr. Prateek Shekhar, Committee Member — Date
Assistant Professor, School of Applied Engineering and Technology, NJIT

BIOGRAPHICAL SKETCH

Author: Deepan Lobo

Degree: Doctor of Philosophy

Date: May 2022

Undergraduate and Graduate Education:

- Doctor of Philosophy in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2022
- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2018
- Bachelor of Technology in Electronics and Communication,
  Manipal Institute of Technology, Karnataka, India, 2016

Major: Electrical Engineering

Presentations and Publications:

Craig Iaboni, Himanshu Patel, Deepan Lobo, JiWon Choi and Pramod Abichandani, "Event Camera Based Real-Time Detection and Tracking of Indoor Ground Robots," *IEEE Access*, vol. 9, pp. 166588-166602, 2021

Pramod Abichandani, Christian Speck, Donald Bucci, William Mcintyre and Deepan Lobo, "Implementation of Decentralized Reinforcement Learning-Based Multi-Quadrotor Flocking," *IEEE Access*, vol. 9, pp. 132491-132507, 2021

Pramod Abichandani, Deepan Lobo, Christian Berry, Vaishali Parikh, William Fligor and Willaim McIntyre, "MATLABArduino. org: An Open-Source Website and YouTube channel for Embedded Systems Education," *IEEE Frontiers in Education Conference (FIE)* pp. 1-8, December. 21, 2021

Deepan Lobo, Drashti Patel, Jorim Morainvile, Prateek Shekhar and Pramod Abichandani, "Preparing Students for Drone Careers Using Active Learning Instruction," *IEEE Access*, vol. 9, pp. 126216-126230, 2021

Pramod Abichandani, Deepan Lobo, Smit Kabrawala and William McIntyre, "Secure Communication for Multi-quadrotor Networks Using Ethereum Blockchain," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1783-1796, Feb. 1, 2021

Pramod Abichandani, Deepan Lobo, Gabriel Ford, Donald Bucci, & Moshe Kam (2020). "Wind Measurement and Simulation Techniques in Multi-rotor Small Unmanned Aerial Vehicles," *IEEE Access*, 8, 54910-54927, 2020

Pramod Abichandani, William Mcintyre, William Fligor and Deepan Lobo, "Solar Energy Education Through a Cloud-Based Desktop Virtual Reality System," *IEEE Access*, vol. 7, pp. 147081-147093, 2019

To my parents for their love and support, my Nana for her constant prayers and blessings, my uncle Rolwin and uncle Royston for their words of encouragement, and finally to all the friends who helped shape me on this incredible journey.

ACKNOWLEDGMENT

I would like to thank my dissertation advisor, Dr. Pramod Abichandani, with whose help this dream has become a reality. For being patient with me and never giving up on me.

To my dissertation committee members, Dr. Moshe Kam, Dr. Mengchu Zhao, Dr. Cong Wang, and Dr. Prateek Shekhar, for their valuable feedback during each crucial stage of my Ph.D. journey. It was my great honor to have the committee's guidance and help in completing this dissertation.

I would like to extend my gratitude to the Department of Electrical and Computer Engineering faculty and staff members for their continued support throughout my doctoral studies.

I want to thank the numerous colleagues, graduate researchers, and undergraduate students who came through the Robotics and Data Laboratory (RADLab), especially my friend Craig Iaboni for sticking around and being a source of encouragement. I also would like to thank my friends especially Priya Bannerjee for being my biggest cheerleader, Riddish, Debojyoti, Uday and members of the FTW-J for being my anchors and sources of motivation.

Finally, I would like to thank my parents and sister for their prayers and for providing me with support every step of the way.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF TABLES
(Continued)

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF SYMBOLS

n	Number of quadrotors
d^{ij}	Euclidean distance between two quadrotors i and j
o^i	Mission origin point
e^i	Mission end point
O	Set of all origin points
E	Set of all origin points
d_{safe}	Minimum safe distance to avoid collision
U^i	Total arc length for quadrotor i
s^i	Linear speed of i^{th} quadrotor
ω^i	Angular speed of i^{th} quadrotor
$\mathbf{p}^i(t)$	Location of quadrotor i on its path at time t
$\mathbb{SP}^i(u)$	3-dimensional 7^{th} order spline path for quadrotor i
$\kappa^i(u)$	Curvature along the spline curve i
T_{hor}	Length of the receding horizon
$\mathcal{B}^{ij}$	Barrier function safe set for quadrotors i and j
$h^{ij}(q^i, q^j)$	Pairwise super ellipsoid for quadrotors i and j
q^i	full state of i^{th} quadrotor
$\mathcal{O}(t)$	Objective function of optimization problem
u^i	Arc length for spline i
a^i	Acceleration for quadrotor i
C^{ij}	Communication connectivity distance between quadrotors i and j
t	Time step
T_{mission}	Transit time between the start of the multiple quadrotor formation and their return to their start points averaged over multiple runs
n_{CBF}	Number of CBF activations averaged over multiple runs and rounded to the nearest integer
T_{comp}	Solver computation time averaged over all quadrotors, further averaged over multiple runs.
T_{saving}	Percentage of T_{comp} time saved
$D_{\text{obj-gap}}$	Percentage gap in the objective function values
e^i	Flight mission end point for quadrotor i
η_d	Communication distance between two quadrotors
$L_{along-wind}$	Scale length of wind turbulence in the along-wind direction of motion
$L_{cross-wind}$	Scale length of wind turbulence in the cross-wind direction of motion
$L_{vertical-wind}$	Scale length of wind turbulence in the vertical-wind direction of motion

$u_{along-wind}$	Along-wind gust speed
$u_{crosswind}$	Cross-wind gust speed
$u_{vertical-wind}$	Vertical-wind gust speed
$\sigma_{along-wind}$	RMS gust speeds in the along-wind
$\sigma_{cross-wind}$	RMS gust speeds in the cross-wind
$\sigma_{vertical-wind}$	RMS gust speeds in the vertical-wind
h	Altitude from sea level (in feet)
$\lvert u_{wind-20} \rvert$	Wind speed at 20 feet
r_{COH}	Cohesion objective seeks to minimize the distance between a quadrotor sUAV and its one-hop neighbors while maintaining an inter-sUAV separation distance
k_{sep}	Inter-sUAV separation distance threshold
r_{ALN}	Velocity alignment objective seeks to minimize the velocity heading between the local quadrotor sUAV and the average heading of its one-hop neighbors
r_{COL}	Collision avoidance distance objective between the flock members
k_{col}	Collision avoidance distance threshold
r_{TGT}	Target seek objective seeks to minimize the distance between a local quadrotor sUAV and the current waypoint of the flock
r_{OBS}	Objective was formulated to ensure safe operations in an environment with obstacles
d^{io}	Quadrotor i distance from an obstacle
k_{obs}	Obstacle avoidance threshold
$\overline{s}$	State space representation of the motion planner for estimates of relative position (i.e., range and bearing) and velocity heading of neighboring sUAV
A	Discretized action space
$Q_m(s_o^{(m)}, a)$	The OF-GM policy uses the learned Q-tables per module
$\tau_{mission}$	Total time to complete the mission from takeoff to landing
ζ_{loss}^{total}	Fraction of total transmitted radio packets that were received across all quadrotors during the flight test
ζ_{loss}^{avg}	Total packet loss per quadrotor pair averaged across a total number of quadrotor pairs during the flight test
τ_{image}	Total time taken to transfer images over blockchain network
$\mathbb{E}$	Inertial Reference Frame
$\mathbb{B}$	Body fixed frame

${}^{\mathbb{E}}u_{quad-ground}$	Velocity of the quadrotor in the IRF
${}^{\mathbb{E}}u_{quad-air}$	Velocity of the quadrotor relative to the surrounding air
${}^{\mathbb{E}}u_{wind}$	Velocity of wind in the IRF
$\Psi_{quad-air}$	True heading of the aircraft
$\Psi_{quad-ground}$	Actual track angle of the quadrotor
Ψ_{wind}	Wind direction
P_d	Dynamic pressure
P_t	Total pressure of the fluid
P_s	Static pressure of the fluid surrounding the tube
ρ	Density of air
u_{pitot}	Airspeed of pitot tube
α	Angle-of-attack
β	Angle-of-sideslip
L	Distance between transducers
ϕ	Roll angle
θ	Pitch angle
${}^{\mathbb{B}}F_{drag}$	Drag force
C^D	Drag coefficient
A_{proj}	Area of the quadrotor frame exposed to the wind
λ	Tilt angle
γ	Angle between the heading direction of the quadrotor and the projection of $b_y \times b_x$ on the ${}^{\mathbb{B}}e_x - {}^{\mathbb{B}}e_y$ plane
u_{a-w}, u_{c-w}, u_{v-w}	Along-wind, cross-wind, and vertical wind speed respectively
σ_{a-w}, σ_{c-w}, σ_{v-w}	RMS gust speeds (also called turbulence intensities) in the along-wind, crosswind, and vertical directions respectively

LIST OF DEFINITIONS

UAV	Unmanned aerial vehicle
RH	Receding horizon
MINLP	Mixed integer nonlinear programming
OF-GM	Object focused greatest mass
SARSA	Sate action reward state action
PSD	Power spectral density
CFD	Computational fluid dynamics
HWIL	Hardware in the loop
SWIL	Software in the loop
RL	Reinforcement learning
CBF	Control barrier function
CBL	Competition-based learning
MVMP	Multi vehicle motion planning
ICAP	Interactive constructive active passive
FAA	Federal Aviation Administration
PBL	Project-based learning
VTOL	Vertical take-off landing
WTR	Wind triangle relationship

CHAPTER 1

INTRODUCTION

Small unmanned aerial vehicles (sUAVs) are a growing class of vehicles that can perform complex tasks, especially in hard-to-reach areas. Rotor-based small unmanned aerial vehicles (sUAV) or drones have witnessed widespread adoption across agriculture, search and rescue operations, surveying, reconnaissance, and photography [6, 61, 246, 276, 280, 305]. There is a distinct advantage of using a multi-sUAV system over a single sUAV platform as they provide increased capabilities for tasks such as surveying, search-and-rescue operations, and mapping [48, 232, 308]. However multi-sUAV systems are not straightforward to deploy autonomously.

Several challenges exist to successfully deploy a multi-sUAV system in an outdoor environment. We have identified five critical challenges which include:

1. Wind: The small size and weight of sUAVs makes them sensitive to wind disturbances.

2. Motion planning: is computationally intensive as the number of sUAVs increases and requires high control effort in presence of wind disturbances.

3. Safety and collision avoidance: sUAVs are susceptible to collisions among each other and surrounding infrastructure due to wind disturbances.

4. Communications: The wireless links between sUAVs can be exploited to disrupt their operation.

5. Education: that addresses sUAV programming, designing, and Federal Aviation Administration (FAA) legalities is yet to catch up with the explosive growth of the US sUAV market.

This dissertation implements and evaluates effective approaches to address each of these key challenges. The main contributions of this dissertation are articulated in the following.

1.1 Contributions

1. A comprehensive first in literature, review of wind measurement and simulation techniques for multi-rotor sUAVs.

2. Implementation and experimental evaluation of a novel decentralized multi-quadrotor flocking motion planning in the presence of wind gusts using reinforcement learning (RL).

3. First-in-literature implementation and experimental evaluation of a decentralized Mixed Integer Nonlinear Programming (MINLP) based motion planner for outdoor multi-quadrotors using safety barrier certificates.

4. Implementation and experimental validation for an Ethereum blockchain based multi-quadrotor communication system for secure wireless data transfer

5. Implementation and multi-methods evaluation of a Competition-based learning (CBL) learning instructional approach for teaching design and fabrication of sUAVs, flight simulations, and commercial drone piloting.

6. Classroom implementation and multi-methods assessment of an active learning-based instructional approaches for teaching sUAV programming, design and fabrication of sUAVs, and commercial sUAV pilot credentialing.

1.2 Organization

The remainder of the dissertation is organized as follows:

- In **Chapter 2**, we present a comprehensive review of techniques for measuring wind speed and airspeed for multi-rotor sUAVs. Three categories of sensing techniques are reviewed: flow sensors, anemometers, and tilt-angle based approaches. We also review techniques for generating wind disturbances in simulation. Wind simulation techniques that use power spectral density (PSD) functions, computational fluid dynamics (CFD), and probabilistic models are examined. Finally, we provide an open-source Python implementation of the Dryden wind turbulence model and embedded code to interface with an ultrasonic anemometer.

- In **Chapter 3**, we present a decentralized, multi-objective reinforcement learning (RL) path planner that achieves waypoint based flocking with quadrotors in the presence of wind gusts (i.e., collision avoidance, velocity alignment, and cohesion). This planner is learned using an object-focused, greatest mass,

state-action-reward-state-action (OF-GM-SARSA) approach. It is trained using a combination of point-mass kinematic models and simulators of quadrotor rigid body dynamics. The Dryden wind gust model is used to simulate wind gusts. The learned planner is then integrated with the DJI N1 flight controller stack of the DJI M100 quadrotor platform. Experimental results are provided in the form of hardware-in-the-loop (HWIL) tests as well as real-time, outdoor flight testing.

- In **Chapter 4**, we address the problem of collision-free, fast, decentralized motion planning for small multi-rotor unmanned aerial vehicles operating in a windy environment using receding mixed-integer nonlinear programming (RH-MINLP). Each quadrotor solves an RH-MINLP to generate its time optimal speed profile along a minimum snap spline path while satisfying constraints on kinematics, dynamics, communication connectivity, and collision avoidance. RH-MINLPs are solved using a novel framework that combines simulated annealing and interior-point methods to handle discrete variables and nonlinear problem components. The Dryden wind gust model is used to simulate realistic wind disturbances. Control Barrier Functions (CBFs) are used for guaranteeing collision avoidance in the face of wind disturbances while alleviating the need to constantly recalculate the motion plans. The framework is validated via Hardware-in-theloop (HITL) experiments using up to 6 DJI M100 quadrotors.

- In **Chapter 5**, we provide experimental validation of an Ethereum blockchain-based software and hardware architecture that enables secure communication for multiple small Unmanned Aerial Vehicles (sUAVs). The experiments involved 3 DJI M100 quadrotors that shared images captured during flight based on smart contracts created using Ethereum's Turing complete programming language. The effect of image size, difficulty level, and consensus algorithms on image transfer times during flight are presented. The effects of wireless network disruptions on the Ethereum network are also documented. The fully documented smart contract code is open-sourced to assist readers in quick prototyping.

- In **Chapter 6**, we present a competition-based active learning approach that prepares undergraduate students for careers in the drone industry. Multiple assessment methods were used to evaluate the students' progress and the perceived drone-related skills gained by the end of the workshops. Assessment methods included data from direct formative (weekly or biweekly curricular milestones), direct summative (drone piloting for multiple flight tests), indirect quantitative (Likert-style feedback survey about student perceptions), and indirect qualitative (descriptive feedback comments) tools.

- In **Chapter 7**, we present an active learning-based instructional approach that prepares students for careers in the drone industry. The approach leverages the ICAP framework and focuses on three foundational pillars of drone careers: 1) drone programming using Python, 2) designing and fabricating drones using Computer-Aided Design (CAD) and rapid prototyping, and 3) the US Federal Aviation Administration (FAA) Part 107 Commercial small Unmanned Aerial Vehicles (sUAVs) pilot test. A case study approach using multiple assessment methods was implemented to examine students' gains in skills

and knowledge and attitudes towards an active learning-based approach for drone education. Assessment methods included direct formative (FAA quizzes and programming assignments), direct summative (capstone project), indirect quantitative (survey of learning gains), and indirect qualitative (focus group interviews and capstone project process videos) tools.

- In **Chapter 8**, we summarize our findings and present future directions of this research.

CHAPTER 2

WIND MEASUREMENT AND SIMULATION TECHNIQUES IN MULTI-ROTOR SMALL UNMANNED AERIAL VEHICLES

2.1 Introduction

Paper: *P. Abichandani, D. Lobo, G. Ford, D. Bucci, & M. Kam (2020). "Wind Measurement and Simulation Techniques in Multi-rotor Small Unmanned Aerial Vehicles," in IEEE Access, 8, 54910-54927, 2020*

Unmanned Aerial Vehicles (UAVs) can be classified into two broad categories: fixed-wing UAVs and multi-rotor UAVs. Over the past two decades, multi-rotor UAVs have emerged as an aerial platform of choice in commercial, research, and defense markets due to some distinct advantages over their fixed-wing counterparts. These advantages include their ability to perform vertical take-off and landing (VTOL), hover at a spot, and yaw at a zero-turn radius. Several market studies provide insight into the 50B+ USD (and growing) global market size of UAVs. A majority of modern applications use small multi-rotor aerial vehicles, also classified as small unmanned aerial vehicles (sUAVs) [102, 126, 256, 284, 290]. This growth in market adoption of multi-rotor sUAVs is primarily due to their affordable cost and strict government regulations across most nations on the use of large UAVs for non-military applications.

Multi-rotor sUAVs are used for several tasks such as agricultural yield monitoring, land surveying, photography, air quality assessment, search and rescue operations, formation control, target tracking, payload transportation, and military operations [5, 20, 119, 218, 358]. Additional uses of these sUAVs are found in meteorological and atmospheric studies [76, 90, 255, 321, 379] and inflow mapping [214, 270, 376].

Despite significant growth in their adoption and relatively friendlier regulatory environment, a key challenge in the use of multi-rotor sUAVs is their high sensitivity

to wind disturbances. Multi-rotor sUAVs are known to be susceptible to degradation of flight stability and performance due to wind gusts [18, 288, 351]. Estimating such wind disturbances and using them to inform flight controls can improve safety and overall flight-plan implementation [27,263,334,356]. As such, an increasing amount of research is dedicated to the study and mitigation of wind effects on multi-rotor sUAVs. Towards this end, this study presents a comprehensive review of wind measurement and simulations for multi-rotor small UAVs.

2.1.1 The Wind Triangle Relationship

The *wind triangle relationship* (WTR) is a foundational topic in aviation that encapsulates the relationship between an aircraft's motion and wind velocity [27, 47, 65, 178, 190, 201, 241, 287, 295, 324, 327]. Specifically, the WTR is a vector relationship between an aircraft's ground velocity, air velocity, and wind velocity. By determining two of the three velocity vectors, it is possible to estimate the third velocity vector.

Illustrated in Figure 2.1, a quadrotor sUAV, is heading towards its destination in open airspace. During transit, this quadrotor experiences windy conditions and

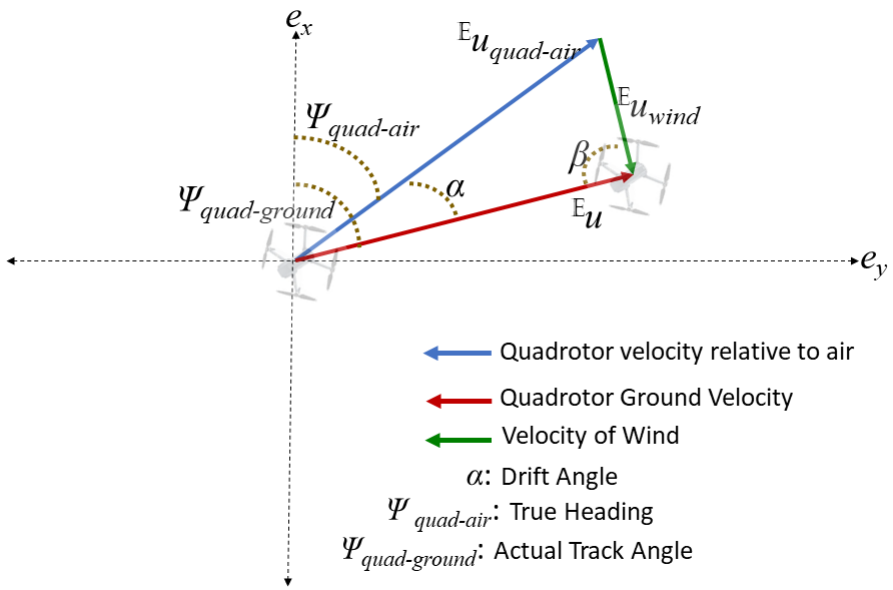


Figure 2.1 Wind causes a drift leading to a difference in ground velocity and air velocity.

drifts off-course by an angle of α from its desired trajectory. The motion of the quadrotor is described using three vector quantities in the Inertial Reference Frame (IRF) $\mathbb{E}$. These three vectors are:

a. Quadrotor ground velocity ${}^{\mathbb{E}}u_{quad-ground}$. The velocity of the quadrotor in the IRF. It is the horizontal velocity of the quadrotor relative to the ground. A GPS module is used to obtain the magnitude and direction of ${}^{\mathbb{E}}u_{quad-ground}$. In aviation theory, the actual path on which a quadrotor travels over the ground is called *track*. $\Psi_{quad-ground}$ represents the actual track angle of the quadrotor after it has drifted off course due to wind.

b. Quadrotor air velocity ${}^{\mathbb{E}}u_{quad-air}$. The velocity of the quadrotor relative to the surrounding air. It is described by true airspeed and true heading of the aircraft $\Psi_{quad-air}$. ${}^{\mathbb{E}}u_{quad-air}$ estimates are obtained via onboard sensor data or flow sensors using techniques described in Section 2.2.

c. Wind velocity ${}^{\mathbb{E}}u_{wind}$. The velocity of wind in the IRF. ${}^{\mathbb{E}}u_{wind}$ can be estimated using a dedicated wind sensor as described in Section 2.2. Alternatively, once ${}^{\mathbb{E}}u_{quad-ground}$ and ${}^{\mathbb{E}}u_{quad-air}$ estimates are available, ${}^{\mathbb{E}}u_{wind}$ is obtained as

$$ {}^{\mathbb{E}}u_{wind} = {}^{\mathbb{E}}\, u_{quad-ground} - {}^{\mathbb{E}}\, u_{quad-air} \tag{2.1} $$

2.1.2 Wind Speed and Airspeed Estimation in Fixed-Wing sUAVs

A considerable amount of effort has been invested in studying wind and airspeed estimation in fixed-wing sUAVs [46, 65, 190, 265, 288, 289]. In general, these studies make use of measurements from flow sensors such as pitot tubes (discussed in Section 2.2.1) and the (non-linear) dynamic model of a fixed-wing sUAV to perform estimation of wind speed and sUAV airspeed.

In [65], the authors proposed an extended Kalman filter (EKF) based method using the WTR to estimate horizontal wind speed and direction simultaneously and used a scaling factor for pitot tube measurements to determine airspeed. In [190], the authors proposed a two-cascaded EKFs based architecture that fused measurements from the GPS, IMU, and the aircraft dynamic model to provide estimates of airspeed, angle of attack and sideslip angle without the need for a pitot tube pressure sensor.

In [289], the authors proposed an attitude, heading, and wind estimation algorithm that incorporated measurements from an air data system (ADS) to predict attitude and airspeed of a fixed-wing UAV effectively. The body axis velocity components were incorporated in the WTR to estimate horizontal and vertical components of the wind velocity. The ADS was comprised of a pitot tube attached along the length of the fixed-wing UAV to measure airspeed and wind vanes to measure the angle of attack and sideslip angle. The estimation algorithm used ADS measurements in the state equations of an unscented Kalman filter (UKF) to produce smooth estimates of airspeed and attitude of the aircraft [367]. In a later work [288], the same authors compared their proposed method with other state-space formulations for wind estimation in fixed-wing sUAV, which used a Kalman filter algorithm. The UKF-based method of [289] showed the best match in simulations with the weather reference data collected experimentally. In [295], an optical flow sensor was used to estimate the angle of attack and sideslip angle to estimate the correct heading of the aircraft. In [42], a nonlinear wind observer provided estimates of wind speed and airspeed. The wind observer combined the model of the fixed-wing aircraft with measurements from a GPS, IMU, and a pitot tube. In [46], a nonlinear state estimator was used, based on multiplicative Kalman filtering [216], to provide real-time estimates of orientation, velocity, and position of the flying vehicle along with 3D wind velocity components. In addition to the fixed-wing sensor suite, the method also used measurements from an angle-of-attack and angle-of-sideslip sensors attached to the aircraft.

2.1.3 Key Differences between Fixed-wing and Multi-rotors

A fixed-wing sUAV has a fixed heading with respect to its motion due to its construction and a non-zero turning radius. The aircraft is always pointing in its direction of motion. This orientation is beneficial for measuring airspeed using

external sensors. Also, the physical frame of a fixed-wing sUAV allows for easy mounting of sensors without significant rotor turbulence. These sensors are commonly fixed in the nose or the wings of the aircraft.

For a multi-rotor sUAV, on the other hand, mounting an external airspeed wind speed sensor is challenging due to a physical frame design that does not provide enough mounting surface area. Secondly, airspeed and wind speed measurements are significantly affected by interference from rotor turbulence and other aerodynamic effects caused by rotors. Lastly, the sensor systems used to estimate airspeed and wind speed need to account for the fact that a multi-rotor sUAV has a non-zero turning radius and can rapidly move in any direction.

Section 2.2 discusses wind and airspeed measurement techniques. These techniques are classified based on the wind sensors they use. Flow-sensing based techniques are discussed in Section 2.2.1; ultrasonic anemometer-based sensing techniques in Section 2.2.2; the tilt-angle based approach which relies on measurements from the on-board IMU and requires flight testing the sUAV in a wind tunnel is described in Section 2.2.3. For each of these techniques, we provide a list of representative studies and detail the sensors used, primary applications, experimental results, and description of the multi-rotor platforms. For the benefit of readers and practitioners, open-source Arduino and Python code to interface with an FT-205 ultrasonic anemometer is provided.

Section 2.3 presents an exposition on the simulation models used to describe wind turbulence. These include Dryden and von Karman turbulence models described in Section 2.3.1, discrete gust models described in Section 2.3.2, and computational fluid dynamics (CFD) models in Section 2.3.3. For the benefit of readers and practitioners, open-source Python code to incorporate the Dryden wind turbulence model in simulations is provided.

2.2 Wind and Airspeed Measurement Techniques for Multi-Rotor sUAV

This section discusses the sensor hardware and associated estimation techniques used to estimate airspeed, wind speed, and wind direction for a multi-rotor sUAV platform. From existing literature, the measurement approaches used for wind speed and airspeed can be categorized based on the hardware used, as depicted in Table 2.1. These categories are:

1. Measurements using flow sensor: The difference between static air pressure and dynamic air pressure experienced by the sUAV is used to infer airspeed of the sUAV.
2. Measurements using ultrasonic anemometer: The speed of ultrasonic pulses in the air is used to infer wind speed and direction directly.
3. Measurements using IMU and global position sensor: used to measure acceleration and attitude measurements of the sUAV, which are then used to estimate airspeed and wind speed of the sUAV.

Table 2.1 Summary of Wind and Airspeed Measurement Techniques

Technique	**Estimation**	**Principle**	**Key Benefits**	**Operating Challenges**
Flow Sensor	Airspeed	Pressure difference	Light weight and low cost	Measures only wind speed. Dependent on direction of flow of air. Sensor placement is non-trivial.
Ultrasonic Anemometer	Wind speed	Time of flight theory	Light weight. Can measure wind speed and direction.	Sensitive to temperature and air pressure. Sensor placement is non-trivial.
Ultrasonic Anemometer	Wind speed	Acoustic resonance principle	Light weight. Can measure wind speed and direction. Independent of air pressure and temperature.	Sensor placement is non-trivial. Requires real-time calibration.
Tilt-Angle Method	Airspeed	Experimental relation between tilt-angle and wind speed	No extra hardware necessary. Measure wind speed and direction	Requires wind tunnel testing. Not reliable for low wind speeds.

2.2.1 Flow Sensors

Wind estimation using pressure flow sensors is a widely used approach in fixed-wing sUAVs [178,287,289,295]. The most common type of flow sensors is pitot-static tubes. These flow sensors are lightweight and relatively inexpensive. Pitot-static tubes are mounted on the nose of the aircraft to provide estimates of airspeed ${}^{\mathbb{E}}u_{quad-air}$. This airspeed estimate can then be used to derive the components of wind velocity, using the WTR. Figure 2.2 depicts various sUAV platforms fitted with flow sensors.

Working Principle of Pitot Tubes: A pitot-static tube, also called Prandtl tube, is used to measure the dynamic pressure P_d of a moving fluid (air) [166,239]. In Figure 2.3, a pitot-static tube consists of a probe that faces the direction of the oncoming wind. The tube has an opening along its central axis that allows the oncoming wind to pass through the main channel and into Chamber 1. The two openings on the outer surface of the tube (outer tube openings) are connected to a set of channels leading to Chamber 2. The channels leading to Chamber 2 (shown in blue) are kept separate from the main channel. The incoming air is brought to rest in Chamber 1 since there is no outlet, and holds the total pressure of the fluid P_t.

Static pressure P_s refers to the pressure of the fluid surrounding the tube. The openings connected to Chamber 2 are perpendicular to the direction of oncoming wind flow. As such, Chamber 2 holds air at static pressure. A transducer element placed between Chambers 1 and 2 measures the pressure difference between total pressure P_t

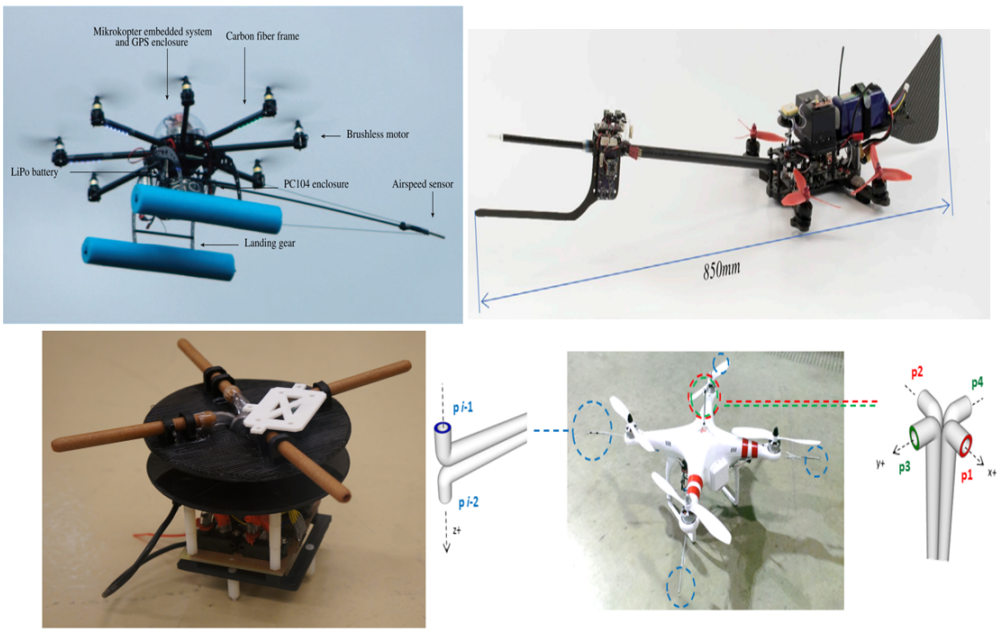


Figure 2.2 Top Left: Pitot tube attached to octorotor [27] Top Right: MHPP flow sensor attached to a quadrotor [270] Bottom Right: Differential pressure sensors attached to quadrotor [387] Bottom Left: Differential pressure sensors attached to quadrotor [355].

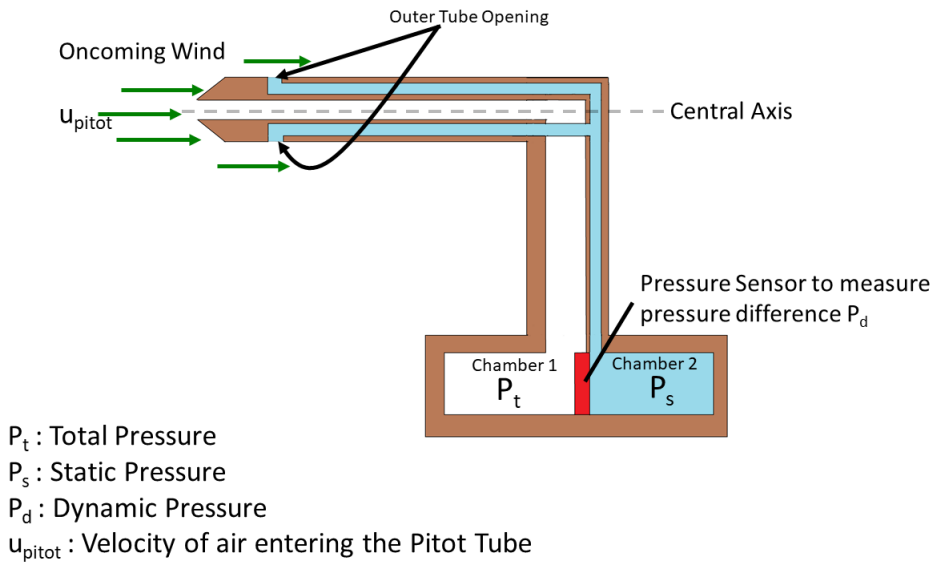


Figure 2.3 Working of a pitot tube.

and static pressure P_s. This pressure difference is the dynamic pressure of air, P_d. The dynamic pressure P_d measured is then used to estimate the airspeed of the pitot tube mounted on the aerial vehicle. This estimation is accomplished through Bernoulli's equation, which states that total pressure is the sum of dynamic pressure and static pressure. Bernoulli's principle also provides for the dynamic pressure experienced by the pitot tube as one half the density of air (ρ) times the square of the tube's speed u_{pitot}.

$$\begin{aligned} P_t &= P_s + P_d, \\ P_t &= P_s + \underbrace{\rho \frac{u_{pitot}^2}{2}}_{P_d}, \\ u_{pitot}^2 &= K \frac{2(P_t - P_s)}{\rho} \end{aligned} \tag{2.2}$$

Where K is a correction factor determined at the time of calibration of the tube to account for measurement sensitivity to air temperature and pressure. Pitot sensors

require calibration to account for their sensitivity to the atmospheric temperature, atmospheric pressure, moisture in the air, flow angle, angle of attack, and angle of sideslip. Procedures and data required for such calibration are described in [121, 180] and references therein. Several modern pitot tube sensors come fully calibrated and temperature-compensated from the factory [313]. The relationship between u_{pitot} and the airspeed of an sUAV is described in [65] as

$$^{\mathbb{E}}u^2_{quad-air} = \frac{|u_{pitot}|^2}{\cos\alpha\cos\beta} \tag{2.3}$$

Where α and β are angle-of-attack and angle-of-sideslip respectively of the aerial vehicle. In the case of a fixed-wing sUAVs, the angle-of-attack and angle-of-sideslip information are obtained by mounting dedicated sensors on the tail and nose of the vehicle. By contrast, due to their multi-rotor aerodynamics and physical construction geometry, multi-rotor sUAVs do not exhibit high angle-of-attack and angle-of-sideslip. To account for this, the authors in [27] use a small angle approximation for angle-of-attack and angle-of-sideslip ($\alpha \approx 0$ and $\beta \approx 0$). This approximation leads to the following relationship

$$^{\mathbb{E}}u^2_{quad-air} = \frac{P_t - P_s}{\big(\rho\cos\alpha\cos\beta\big)/(2K)} \equiv \frac{P_t - P_s}{sf} \tag{2.4}$$

Where sf is a scaling factor, this scaling factor is experimentally determined at the time of mounting the pitot tube to the multi-rotor sUAV [27].

Pitot tubes for Multi-Rotor sUAVs: Airspeed measurements using a pitot tube are for the direction in which the tube is pointing. Multi-directional airspeed detection in the case of multi-rotor sUAVs, accounting for multiple movement directions, would

Table 2.2 Representative Studies on Wind and Airspeed Estimation using Pressure Flow Sensors in Multi-rotor sUAVs

#	Author	Application	Flow Sensor Type	Multi-rotor Platform
1	*B. Arain and F. Kendoul [27]*	Improve flight performance	Pitot tube and pressure sensor	Helix8 octarotor
2	*Nguyen Khoi Tran [355]*	Improve flight performance	Sensirion differential pressure sensors	AsTec Pelican quadrotor
3	*Marino et al. [270]*	Wind sensing platform in urban environment	MHPP flow sensor	Custom quadrotor
4	*Sydney et al. [387]*	Improve flight performance	Honeywell differential pressure sensor	DJI Phantom quadrotor

require the use of several pitot tubes, thereby increasing the weight and complexity of the vehicle. Additionally, compared to fixed-wing sUAVs, the use of flow sensors in multi-rotor vehicles is limited by interference from the rotors (rotor wash) [27, 270]. An approach to address this interference is to find experimentally, or by dynamic flow modeling, the location of minimum rotor interference on the body of a multi-rotor vehicle.

Table 2.2 summarizes representative studies on wind and airspeed measuring techniques using pressure flow sensors mounted on multi-rotor sUAV. The table lists the sensor type and the sUAV platform used in the studies. The studies use a flow sensor to improve flight performance or to enable the multi-rotor sUAV as a wind sensing platform.

Remarks on Air Flow Sensors: Pressure flow sensors are the most common devices used to measure airspeed of large aircraft or small fixed-wing sUAVs. These sensors are lightweight and relatively inexpensive devices compared to other airspeed sensors (such as anemometers). As multi-rotor sUAVs are being increasingly used for

a wide variety of applications, pressure flow sensors serve as an inexpensive option to measure airspeed of a multi-rotor sUAV.

On the other hand, the working of a pressure flow sensor is dependent on the flow of oncoming wind into the opening of its channel. This can be a problem in the case of multi-rotor sUAVs since the rotors can disturb the oncoming airflow and result in inaccurate airspeed measurements. It is thus crucial to select a suitable location on the multi-rotor sUAV that receives minimum interference from the rotors. Another limitation of pressure flow sensors is that they cannot measure wind direction simultaneously with wind speed. Heading and attitude measurements of the multi-rotor vehicle need to be considered to estimate wind direction. Pressure flow sensors measure airspeed by measuring the pressure difference through their tube. A multi-rotor sUAV due to its 6-degree of freedom would require the use of multiple pressure sensors to measure wind speed in all directions. This adds complexity to the mounting of these sensors and limits the payload carrying capacity of a multi-rotor sUAV.

2.2.2 Ultrasonic Anemometers

An ultrasonic anemometer is a small, lightweight sensor that can provide estimates on wind speed and direction. The durability and longevity of the ultrasonic anemometer offered by the lack of any moving parts make it a popular sensor choice to be used with multi-rotor vehicles, especially in meteorological applications [90,255,321]. Figure 2.4 depicts various sUAVs fitted with ultrasonic anemometers.

Working Principle: An ultrasonic anemometer measures wind speed using the time period taken by an ultrasonic pulse to travel from a transmitter to a receiver. The workings of this sensor are based on the time of flight principle (TOF) [142,215], wherein the system emits an ultrasonic pulse using a transmitter or emitter, and is able to measure the distance to a solid object (also referred to as the reflector) based

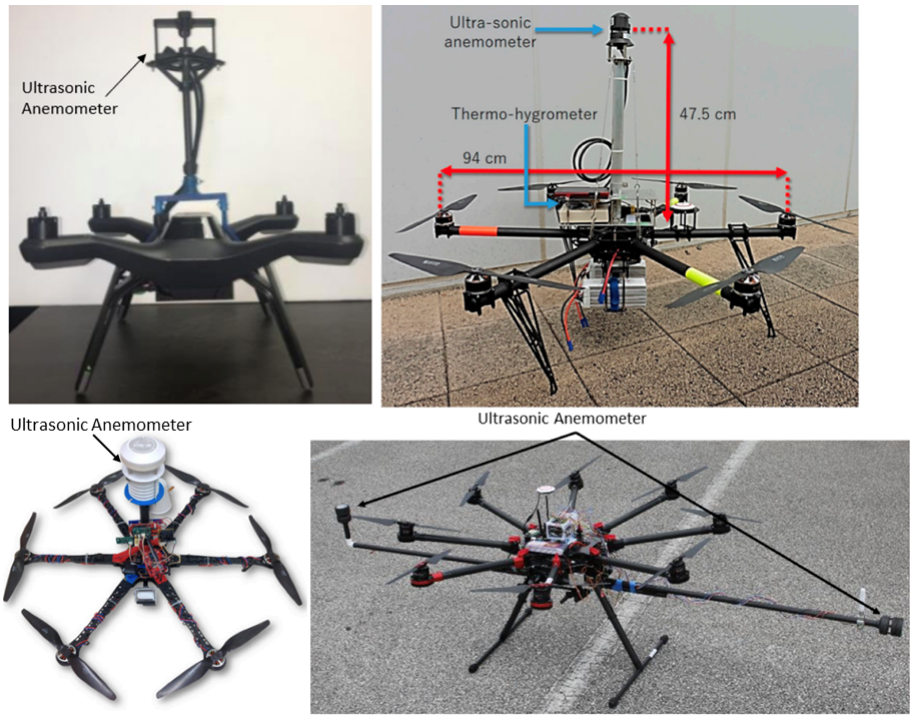


Figure 2.4 Top Left: Tri-Sonica ultrasonic anemometer attached to quadrotor [90]. Top Right: FT702 ultrasonic anemometer attached to hexarotor [321]. Bottom Left: DS-2 ultrasonic anemometer attached to hexarotor [255]. Bottom Right: FT205 ultrasonic anemometer attached to octorotor [9].

on the time taken for the pulse to echo back to the emitter. The ultrasonic wavefield is affected by the fluid field.

An ultrasonic anemometer features multiple pairs of ultrasonic transducers located at a known distance L. In Figure 2.5, each transducer has a Transmitter

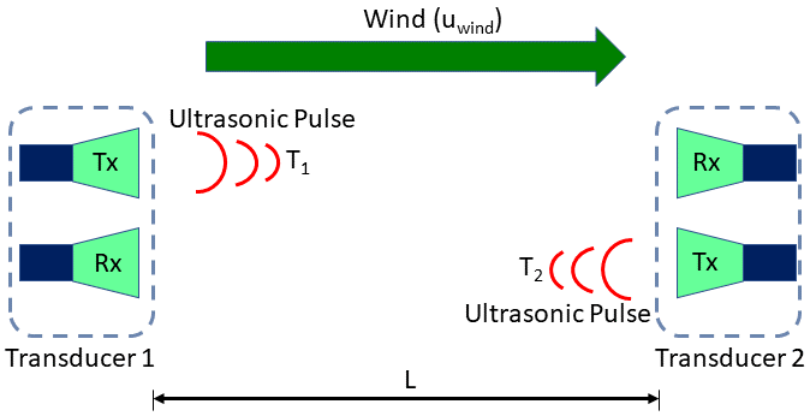


Figure 2.5 Working of an ultrasonic anemometer.

(Tx) and a Receiver (Rx) component. The ultrasonic anemometer first sends an ultrasonic pulse from the Tx of Transducer 1 to the Rx of Transducer 2. The time between the transmission from Tx-Transducer 1 to Rx-Transducer 2 is measured. Next, a second pulse is transmitted from Tx-Transducer 2 to Rx-Transducer 1, and the time between this transmission and reception is measured.

In the absence of wind, these two time periods would be the same for the path from Tx-Transducer 1 to Rx-Transducer 2, and Tx-Transducer 2 to Rx-Transducer 1. However, in the presence of wind, these two-time measurements are different, and the pulse traveling opposite to the direction of the wind will take longer to reach the Rx component [142]. This time difference is then used to estimate wind speed and direction.

Acoustic Resonance Anemometer (AcuRes): Another type of ultrasonic anemometer uses acoustic resonance to measure wind speed and direction independent of variations in air temperature and pressure. Acoustic resonance occurs when the frequency of a wave produced by a transducer matches the natural frequency of an acoustic cavity, resulting in a wave with increased amplitude. This technology was patented by Kapartis in 1999 [155].

Figure 2.6 Left Image: Alignment of transducers on the upper reflector for an acoustic resonance ultrasonic anemometer. Right Image: Side view of the FT205 acoustic resonance ultrasonic anemometer.

Figure 2.6 depicts an acoustic resonance anemometer. The sensor consists of upper and lower reflector plates that house the ultrasonic transducer pairs. The gap between reflector plates is called the region of resonance. Figure 2.6 also shows the position of the transducers labeled here as A, B, and C mounted on the upper reflector.

Transducer A generates an acoustical standing wave at an eigenfrequency of the cavity to excite an acoustic resonance. Transducer B senses the transmitted acoustical signal. The phase difference between the transmitted and received acoustical signals is proportional to the velocity of the airflow along the line adjoining the two transducers A and B. The triangular orientation of the transducers enables the resolution of the wind vector components along the lines adjoining the transducer pairs A - B and B - C. The linear relationship between phase and wind speed is independent of the ambient air pressure and temperature as it is accounted for at the time of calibration. In addition, operating at resonance improves the signal to noise ratio [333].

Table 2.3 Representative Studies on Wind Speed and Direction Measurements using Ultrasonic Anemometers mounted on Multi-Rotor sUAVs

#	Author	Working Principle	Application	Wind Sensor	Experiment Results	Multi-rotor Platform
1	*Palomaki [255]*	TOF	Atmospheric and meteorological study	DS-2 2D ultrasonic anemometer	RMSE $0.27ms^{-1}$ - $0.67ms^{-1}$ and 25° - 56° under wind speed conditions $1ms^{-1}$ – $5ms^{-1}$	DJI Flame Wheel F550 hexarotor with Pixhawk
2	*Donnel [90]*	TOF	Atmospheric and meteorological study	Tri-Sonica Mini 2D ultrasonic anemometer	RMSE $1.13ms^{-1}$ and 133.36° (uncorrected) under wind speed conditions $6.75ms^{-1}$	3DR Solo Quadrotor
3	*Shimura [321]*	AcuRes	Atmospheric and meteorological study	FT702 2D ultrasonic anmeometer	$0.6ms^{-1}$ and 12° under wind speed up to $11ms^{-1}$	Hexarotor SPIDER CS6
4	*Adkins [9]*	AcuRes	Atmospheric and meteorological study	FT205EV ultrasonic anemometer	-	DJI S1000 octorotor

Ultrasonic Anemometer for Multi-Rotor sUAVs: Table 2.3 summarizes representative studies on wind measurement using ultrasonic anemometers mounted on a multi-rotor sUAV [9,90,255,321]. All these studies used the sUAV for atmospheric and meteorological measurements. The mobility options offered by multi-rotor sUAV make them a suitable platform for collecting wind measurements and measuring the vertical profile of wind [321]. The accuracy of ultrasonic anemometer measurements in these studies was cross-checked in outdoor flight tests by piloting the sUAVs near a benchmarking wind measuring device attached to a ground-based tower. Wind

measurements collected from the sUAV were compared with the measurements from the ground tower, and root means square error (RMSE) values were calculated. The RMSE calculations documented in these studies are applicable in situations where the multi-rotor sUAV is hovering at the same height as the wind measuring device on the ground-based tower. Differences in the altitudes between a multi-rotor sUAV and the reference wind measuring device should be accounted for in other situations. The RMSE values of wind speed and direction are shown in Table 2.3.

A common approach to mounting the ultrasonic wind sensor is using a pipe mount atop the multi-rotor vehicle along the center of the rotor-body frame. In [90], particle image velocimetry (PIV) measurements were performed to examine the flow field around the multi-rotor frame to determine the ideal placement for the wind sensors. The best placement of wind sensors was found to be along the body centerline furthest away from the downwash of the rotors. The study, however, did not consider inflow effects caused by the rotor, which can affect placement consideration. In [90], the authors equipped a 3DR solo quadrotor with the Tri-Sonica Mini ultrasonic anemometer at the height of 25 cm above the rotor plane. The Tri-Sonica Mini ultrasonic anemometer weighs less than 50 grams a length of 5.2 cm [26]. In [321], a 2D ultrasonic anemometer FT702 from FT Technologies [348] was attached to a SPIDER CS6 hexarotor. The ultrasonic anemometer is 16.1 cm long and weighs 350 grams. This sensor was placed at a height of 47.5 cm above the rotor plane using an aluminum pole. Three-wire stays were used to support the wind sensor to reduce vibrations. In [255], a 2D sonic anemometer, DS-2 from Decagon Devices [83], was attached on top of a DJI Flame Wheel F550 hexarotor using a 30 cm pole mount. The ultrasonic anemometer weighed 500 grams with a length of 7.5 cm.

Remarks on Ultrasonic Anemometers: Ultrasonic anemometers are mostly used for wind surveying, wind profiling in urban environments, and surveying

wind in topographically cluttered areas. These sensors, when mounted on a multi-rotor sUAVs, provide a relatively inexpensive option compared to setting up a meteorological observation tower. Their compact form factor makes them suitable for physically mounting on a multi-rotor sUAV. In contrast to standard flow sensors, an ultrasonic anemometer can estimate wind speed and direction simultaneously.

On the other hand, their weight and pole-mounting requirement limit the overall payload carrying capacity of the sUAV. Ultrasonic transducer readings are sensitive to precipitation, where raindrops may vary the speed of sound. As such, these sensors should be constantly calibrated to reduce the errors caused by environmental factors such as changing temperature, precipitation, and ambient pressure. As is seen in Fig 2.4, different studies have mounted these sensors in different manners, and there is a need for a more rigorous understanding of optimal placements of these sensors to account for wind disturbances due to rotors.

Open-source Code for Real-Time Ultrasonic Wind Sensor Data Acquisition: For the benefit of the readers, open-source code to interface with an FT205 ultrasonic sensor with an embedded computer using a universal asynchronous receiver and transmit (UART) is provided in [78]. We conducted flight tests with this sensor by mounting it on a DJI M100 quadrotor sUAV. As shown in Figure 2.7 the sensor

Figure 2.7 FT-205 wind sensor mounted on the DJI Matrice 100 sUAV to collect wind data during flight test.

was mounted on the quadrotor using 20 inches long 3D printed pole. The FT205 was sampled at a frequency of 2 Hz. The test was conducted with the DJI M100 quadrotor hovering in an open field for a duration of 5 minutes. Figure 2.8 is a windrose plot of wind data from the FT-205 wind sensor. The plot depicts the frequency of winds blowing from particular directions. The color bands correspond to the wind speed ranges in meters per second. The direction of the longest spoke shows the wind direction with the highest frequency.

2.2.3 IMU and GPS based Airspeed Estimation using Tilt-Angle Approach

An Inertial Measurement Unit (comprised of a 3-axis accelerometer, 3-axis gyroscope, and magnetometer) and a GPS module are part of the standard configuration onboard sensor suite of almost all sUAVs operating in outdoor environments. Several studies

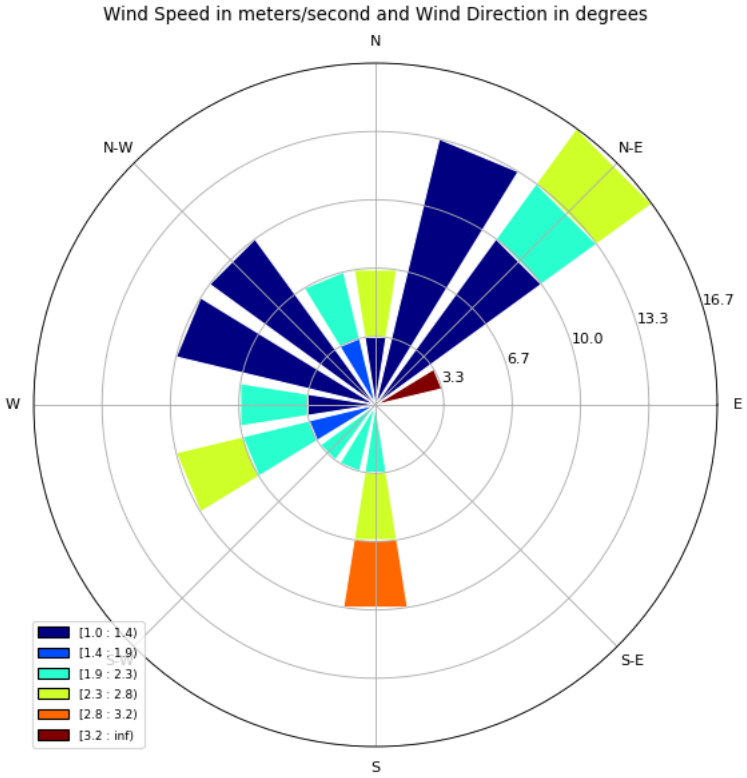


Figure 2.8 Windrose plot for wind speed and direction data recorded from a FT-205 wind sensor mounted on the DJI Matrice 100 sUAV.

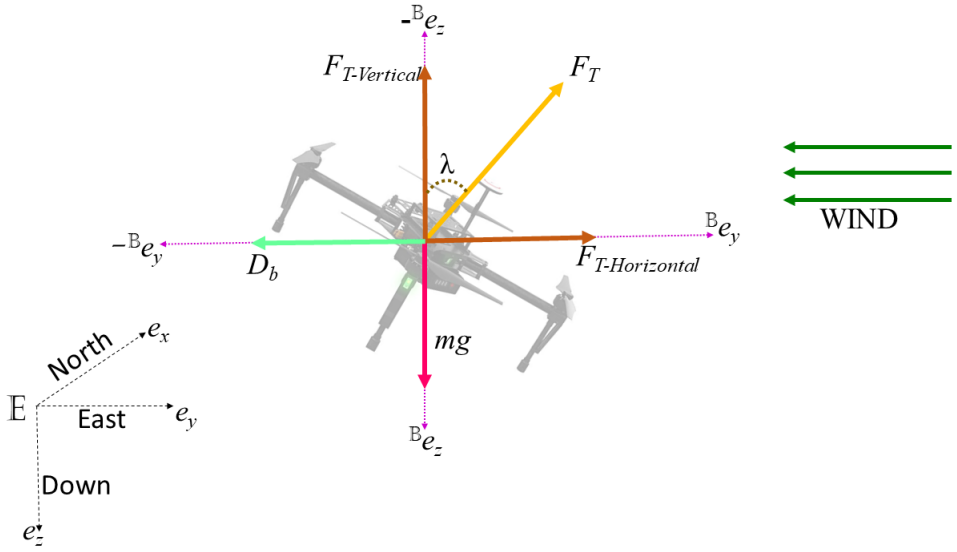


Figure 2.9 Relation between tilt angle and drag force.

have shown the use of this sensor suite (IMU and GPS) to estimate airspeed ${}^{\mathbb{E}}u_{quad-air}$ of a quadrotor.

This section describes the *tilt angle approach* used to estimate ${}^{\mathbb{E}}u_{quad-air}$ using IMU and GPS found onboard a quadrotor [241]. This approach is valid under steady-state conditions, that is when the quadrotor is hovering.

In Figure 2.9, a quadrotor's body frame of reference $[{}^{\mathbb{B}}e_x, {}^{\mathbb{B}}e_y, {}^{\mathbb{B}}e_z]$ is defined. The axes of this reference frame are aligned with the inertial frame of reference (IRF) $[{}^{\mathbb{E}}e_x, {}^{\mathbb{E}}e_y, {}^{\mathbb{E}}e_z]$ with the origin fixed at the center of gravity (CG) of the quadrotor. The GPS provides sUAV ground speed measurements in IRF, and the IMU provides body velocity components and attitude measurements in the body frame described by unit vectors ${}^{\mathbb{B}}e_x, {}^{\mathbb{B}}e_y, {}^{\mathbb{B}}e_z$.

Measuring Quadrotor AirSpeed from Quadrotor Tilt-Angle: When a quadrotor is hovering at a position in space, it experiences a drag force due to wind, which causes the quadrotor to tilt [54, 90, 241, 255]. This drag force and other forces acting on a quadrotor during hovering are illustrated in Figure 2.9.

The tilt angle λ shown in Figure 2.9 is a combination of roll(ϕ) and pitch(θ) angles. The roll and pitch angles are measured from the vertical $-^{\mathbb{B}}e_z$ as shown Figure 2.10. The unit vector that represents the roll angle is $b_y = \begin{bmatrix}0 & cos(\phi) & sin(\phi)\end{bmatrix}^T$. The unit vector that represents the pitch angle is $b_x = \begin{bmatrix}cos(\theta) & 0 & -sin(\theta)\end{bmatrix}^T$. The tilt angle λ is used to estimate the attitude of the quadrotor and estimate the magnitude of quadrotor air speed ($|^{\mathbb{E}}u_{quad-air}|$). The tilt angle λ is calculated as

$$\lambda = cos^{-1}\left(\frac{-n_{b_x-b_y} \cdot (b_y \times b_x)}{|n_{b_x-b_y}| \cdot |b_y \times b_x|}\right), \tag{2.5}$$

where $-n_{b_x-b_y}$ is the unit vector normal to the plane formed by $b_x - b_y$ as illustrated in Figure 2.11. $n_{b_x-b_y} = [0, 0, 1]^T$.

Under steady-state conditions (quadrotor hovering or vehicle operating under slow varying wind fields), the attitude of the quadrotor will likely remain unchanged. In turbulent environments or fast-moving translational motion of the quadrotor, steady-state assumptions do not hold [381]. Figure 2.9 shows the relationship between quadrotor tilt and the drag force. The drag force D_b due to wind can be calculated using.

$$^{\mathbb{B}}F_{drag} = g \cdot m \cdot tan(\lambda) \tag{2.6}$$

As described in [90, 241, 255, 381], the magnitude of airspeed of the quadrotor ($|u_{quad-air}|$) can then be calculated using the principle of conservation of energy for a fluid in motion, namely

$$|u_{quad-air}| = \sqrt{\frac{2 \cdot {}^{\mathbb{B}}F_{drag}}{\rho A_{proj}(\lambda) C^D(\lambda)}} \tag{2.7}$$

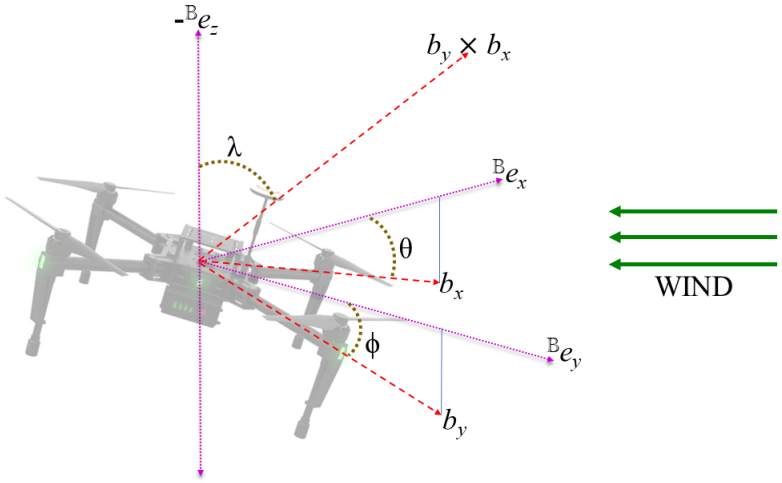


Figure 2.10 Relation between roll(ϕ) and pitch(θ) angles with tilt angle λ.

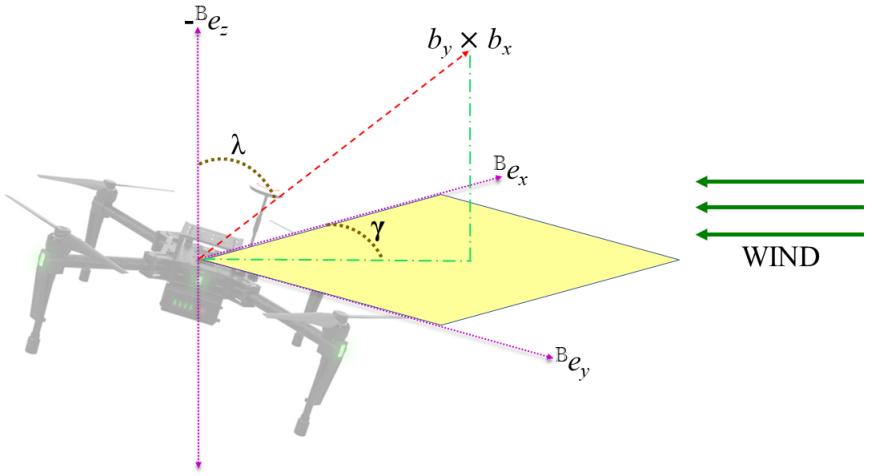


Figure 2.11 Projection of $b_x \times b_y$ on ${}^{\mathbb{B}}e_x - {}^{\mathbb{B}}e_y$ plane.

Figure 2.12 Steps to estimate the magnitude (speed) and direction of ${}^{\mathbb{E}}u_{quad-air}$.

Here, ρ is the density of the fluid (air in this case), A_{proj} is the area of the quadrotor frame exposed to the wind, which is a function of tilt angle λ and is also referred to as the projected area. C^D is the drag coefficient of the quadrotor body. The projected area (A_{proj}) and drag coefficient (C^D) both vary as a function of the tilt angle. Under hover condition, $|u_{quad-air}|$ is equal to the wind speed $|{}^{\mathbb{E}}u_{wind}|$ experienced by the quadrotor [54, 255, 381]. In [54, 241], the projected area was

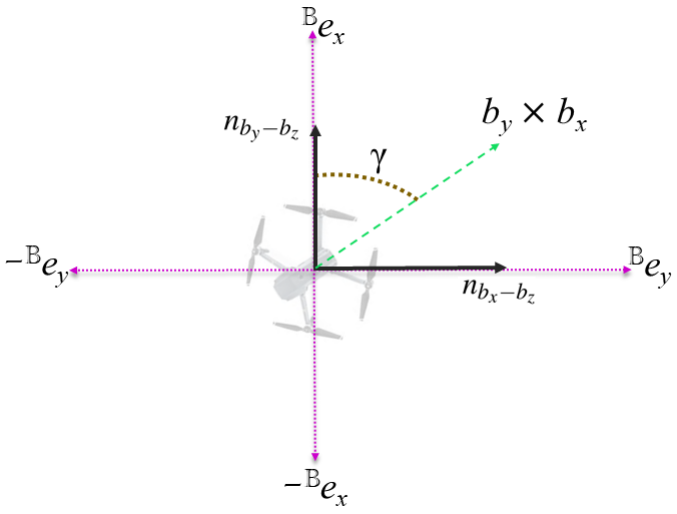


Figure 2.13 Top view of $^{\mathbb{B}}e_x - ^{\mathbb{B}}e_y$ plane.

obtained by constructing a 3D model of the quadrotor body. Then, using image processing, the reference area at each inclination angle as a count of pixels was calculated. In [364], the quadrotor frame was modeled as a cylinder. The area exposed to wind was calculated as the sum of the lateral surface area and the area of bases. The roll and pitch angles were included in the area calculation to get an estimate of the effective influence area of wind. The drag coefficient was computed by wind tunnel experiments, as discussed in [90, 241, 307] or indoor tests, as reported in [355].

Measuring Quadrotor Heading Direction: Figure 2.13 shows the top view of projection of the vector $b_y \times b_x$ on $^{\mathbb{B}}e_x - ^{\mathbb{B}}e_y$ plane. To calculate the quadrotor heading direction $\Psi_{quad-air}$, the angle γ between the heading direction of the quadrotor and the projection of $b_y \times b_x$ on the $^{\mathbb{B}}e_x - ^{\mathbb{B}}e_y$ plane needs to be calculated using (2.8).

$$\gamma = cos^{-1}\left(\frac{n_{b_y-b_z} \cdot (b_y \times b_x)}{|n_{b_y-b_z}| \cdot |(b_y \times b_x)|}\right) \tag{2.8}$$

The viewing direction considered here is the normal vector $n_{b_y - b_z} = [1, 0, 0]$ normal to the plane of ${}^{\mathbb{B}}e_y - {}^{\mathbb{B}}e_z$. To determine which side of the quadrotor the projection of $b_y \times b_x$ lies, (2.9) needs to be solved. The result of (2.8) lies in the interval $[0°, 180°]$.

$$n_{b_x - b_z} \cdot (b_y \times b_x) = \begin{cases} < 0 & \text{if}(b_y \times b_x)\text{is left side} \\ > 0 & \text{if}(b_y \times b_x)\text{is right side} \\ = 0 & \text{otherwise} \end{cases} \tag{2.9}$$

The heading direction $\Psi_{quad-air}$ can be calculated using (2.10).

$$\Psi_{quad-air} = \begin{cases} 360° - \gamma & \text{if } \left(n_{b_x - b_z} \cdot (b_y \times b_x)\right) < 0 \\ \gamma & \text{otherwise} \end{cases} \tag{2.10}$$

Calculating Wind Velocity ${}^{\mathbb{E}}u_{wind}$: The wind speed ${}^{\mathbb{E}}u_{wind}$ is calculated using the WTR described in (2.1) and the law of cosines. The drift angle (α) shown in Figure 2.1 is equal to the difference between the track angle $\Psi_{quad-ground}$ and aircraft heading $\Psi_{quad-air}$. The magnitude of wind speed $|{}^{\mathbb{E}}u_{wind}|$ is calculated based on the relationship expressed in (2.11).

$$\begin{aligned} |{}^{\mathbb{E}}u_{wind}| = \Big\{ & |{}^{\mathbb{E}}u_{quad-air}|^2 + |{}^{\mathbb{E}}u_{quad-ground}|^2 \\ & - 2 \cdot |{}^{\mathbb{E}}u_{quad-air}| \cdot |{}^{\mathbb{E}}u_{quad-ground}| \cdot cos(\alpha) \Big\}^{1/2} \end{aligned} \tag{2.11}$$

The wind direction Ψ_{wind} is calculated by solving (2.12) and (2.13). Here β is the angle between the wind speed vector ${}^{\mathbb{E}}u_{wind}$ and ground speed vector ${}^{\mathbb{E}}u_{quad-ground}$.

Equations (2.11) and (2.13) calculate the wind speed $|{}^{\mathbb{E}}u_{wind}|$ and direction Ψ_{wind} for $0 < \alpha < 180°$. The cases $\alpha = 0°$ and $180°$ are considered separately. The sign of β depends on the quadrotor heading direction $\Psi_{quad-air}$. If the heading direction is within the interval $\left[\Psi_{quad-ground} + 180°, \Psi_{quad-ground}\right]$ then the angle is positive $(+\beta)$, otherwise it is negative $(-\beta)$.

$$\beta = cos^{-1}\left(\frac{|u_{quad-air}|^2 - |{}^{\mathbb{E}}u_{quad-ground}|^2 - |{}^{\mathbb{E}}u_{wind}|^2}{-2|{}^{\mathbb{E}}u_{quad-ground}| \cdot |{}^{\mathbb{E}}u_{wind}|}\right) \tag{2.12}$$

$$\Psi_{wind} = \left(\Psi_{quad-ground} + 180° \pm \beta\right) \tag{2.13}$$

Experimental Results of the Tilt Angle Approach: A list of representative studies that describe the tilt-angle based approach for wind and airspeed measurement is provided in Table 2.4. The relationship between the multi-rotor tilt angle and the drag force it experiences is dependent on the construction of the aerial vehicle. Extensive wind tunnel tests need to be carried out to collect data required to determine the tilt-angle and drag force relation that is specific to the multi-rotor vehicle [90, 112, 241]. The experimental results summarized in Table 2.4 represent the RMSE values for wind speed and direction.

Table 2.4 Representative Studies on Wind and Airspeed Measurement for Multi-rotor SUAVs using the Tilt-angle Approach

#	Author	Application	Experiment Results	Multi-rotor Platform
1	*Neumann and Bartholmai* [241]	Gas source localization and gas distribution mapping	RMSE $0.6ms^{-1}$ and 14.2°	AirRobot AR100-B quadrotor
2	*Xingyu Xiang et al.* [381]	Wind profiling for energy harvesting	RMSE $3.981ms^{-1}$ and 241°	DJI Matrice 100 quadrotor
3	*Caroline Brosy et al.* [47]	Atmospheric and Meteorological study	RMSE $0.71ms^{-1}$ and 14.5°	DJI 550 hexacopter with Pixhawk 3DR
4	*Yao Song et al.* [327]	Improve Flight Performance	Standard Deviation $0.0671ms^{-1}$ and 31.4°	QuadRotor Simulator
5	*Moyano Cano, Javier* [54]	Wind profiling for wind farms	RMSE $4.41ms^{-1}$ and 195°	ArduCopter
6	*Ross T. Palomaki et al.* [255]	Atmospheric and Meteorological study	RMSE $0.5ms^{-1}$ and 30° under wind speed conditions $0-5ms^{-1}$	DJI Flame Wheel F550 hexarotor with 3D Robotics Pixhawk
7	*Geoffrey W. Donnel et al.* [90]	Atmospheric and Meteorological study	RMSE $1.09ms^{-1}$ and 81.68° under wind condition of $2ms^{-1}$ - $7ms^{-1}$	3DR Solo Quadrotor

- In [241], the quadrotor used in outdoor flight tests was the AirRobot AR100-B quadrotor. A Young 81000 ultrasonic anemometer mounted on a tower at the height of approximately 2 meters provided a reference to the measurements from the quadrotor. Two outdoor flight tests were conducted to validate the proposed estimation technique. In the first flight test, the quadrotor was hovering at a horizontal distance of 2 meters to 5 meters from the anemometer at an Above Ground Level (AGL) height of 2 meters for a duration of 20 minutes. In the second flight test, the estimation technique was validated under moving flight conditions where the quadrotor was made to move around the anemometer tracing a square of size 30 meters × 30 meters. The RMSE for wind speed and direction when the quadrotor was hovering was found to be $0.6ms^{-1}$ and 14.02°, respectively. The RMSE for wind speed and direction when the quadrotor was

moving was found to be $0.36ms^{-1}$ and 14.77°, respectively. The authors posit these RMSE values could be attributed to inaccuracies in the GPS sensor. The changing position of the quadrotor about the anemometer could also result in deviations in measurements.

- In [381], outdoor flight tests were performed using a DJI Matrice 100 quadrotor. Validation of the wind measurements was performed using the HoldPeak HP866b digital anemometer mounted on a mast of unspecified height. Two flight-test were conducted to verify the estimation technique under hover and forward flight. The wind measurements when the quadrotor was in forward flight condition were found to be inaccurate. The authors attributed this inaccuracy to the use of a quadrotor dynamic model that did not take into account inflow effects. The results from the hovering flight showed an RMSE of $0.6ms^{-1}$ with respect to the reference anemometer measurements.

- In [47], the outdoor flight test was performed using the DJI 550 Flame Wheel quadrotor equipped with a PixHawk flight controller. Validation of wind measurements from the quadrotor was done using a uSonic 3D ultrasonic anemometer from Metek GmbH mounted on a $9m$ tower. The quadrotor was made to hover for a duration of 5 minutes at a distance of approximately $5m$. This height was the same as the observation tower. A 10 seconds moving average filter was applied to the measurements from the quadrotor and the anemometer. The resulting RMSE values for wind speed and direction were $0.7ms^{-1}$ and 14.5°, respectively. The authors note that the larger volume of the quadrotor compared to the anemometer meant that it could not react to small, turbulent wind gusts (eddies) and therefore, could not capture the full range of wind speeds.

Remarks on the Tilt Angle Approach: An advantage of the tilt-angle approach to wind estimation is that it does not rely on measurements from augmented sensors to estimate wind speed and direction. This method solely relies on the IMU and position sensors equipped on-board a multi-rotor sUAV. The tilt-angle approach enables the multi-rotor platform to become a wind measuring device without compromising the payload carrying capacity of the aerial vehicle. The multi-rotor sUAV using the tilt-angle approach can aid in wind profiling for wind farms, atmospheric and meteorological study, and gas source localization.

A limitation of this approach is the hover model assumption for wind speed estimation, which makes this model unsuitable for wind speed estimates under fast-moving conditions. Another limitation of this approach is the requirement of wind

tunnel tests to determine the drag force and tilt angle relationship. The relationship determined is unique to that specific multi-rotor vehicle. A wind tunnel to perform tests may not be available to everyone, thereby limiting the usability of this technique.

2.2.4 Comparison and Future Directions

Three wind and airspeed measurement techniques were discussed. In the following, we compare these three techniques and discuss future directions.

- Cost of sensors: Of the three methods discussed here, the tilt angle approach uses sensors onboard the flight controller of the multi-rotor sUAV. As such, there is no cost of additional sensors. The pressure-flow sensors are found to be less expensive than ultrasonic anemometers [91, 131, 314]. It is expected that the cost of these sensors will continue to drop as their adoption increases.

- Speed and direction data: Pressure flow sensors provide only wind speed data in the direction they are facing. 3D ultrasonic anemometers provide wind speed and direction measurements simultaneously in all directions [9]. The tilt angle approach is also able to provide wind speed and direction measurements simultaneously. However, unlike 3D ultrasonic anemometers, the tilt angle approach cannot be used to obtain wind measurements in a vertical direction.

- Sensor resolution: The pressure-flow sensors provide airspeed based on the measured differences in pressure. This pressure difference can become difficult to resolve when the sUAV is hovering, or the magnitude of affecting wind is too small. As such, pressure flow sensors exhibit relatively lower resolution in these situations. On the other hand, modern ultrasonic anemometers feature relatively higher resolution. For example, the FT 205 sensor has a wind speed resolution of 0.1 ms^{-1}. It is expected that manufacturers and researchers will continue to invest efforts in creating higher resolution sensors.

- Physical mounting: Both pressure flow sensors and ultrasonic anemometers are required to be mounted on the multi-rotor sUAV. In [90], an optimal location for the ultrasonic anemometer was determined to be along the center of the multi-rotor sUAV. Despite this, a bias was observed in the wind measurements from the ultrasonic anemometer mounted at this location. Mounting these sensors on the vehicle such that they experience minimum rotor disturbance or do not affect the stability of the vehicle is still an ongoing area of research.

2.3 Simulation Models for Wind Generation

In the previous section, sensor-based wind speed and direction measurement/estimation techniques were discussed. In several studies, wind needs to be generated using physical or simulation techniques. In this section, techniques used to model wind in a simulation environment are presented.

These simulation models are used for evaluation of flight dynamics and performance of the multi-rotor sUAV in the presence of wind and can be embedded in a variety of simulation scenarios. In the following discussions, $u_{wind} = [u_{wind-x}, u_{wind-y}, u_{wind-z}]$ is the wind vector with magnitude $|u_{wind}|$ and azimuth angle Ψ_{wind}.

To study the wind effects on quadrotors, it is first necessary to generate realistic wind disturbances likely to be encountered by the quadrotor. Wind can be described, classified, and modeled in a variety of ways. The type of wind relevant to the study of quadrotor performance under wind disturbances is called a *gust*. A wind gust (also known as turbulence) is described by brief changes of wind velocity from a steady value caused by changes in atmospheric pressure and temperature. Table 2.5 depicts three techniques to generate wind in simulation, along with their associated benefits and challenges.

Table 2.5 Summary of Wind Generation Techniques

Technique	Model Type	Key Benefits	Challenges
Continuous gust [17, 217, 355, 376]	Wind Sheer	Can model wind behavior in open skies	Cannot model wind behavior near obstacles
Discrete gust [16, 364]	Step or Pulse Change	Can model gust of known amplitude and duration	Statistics of the model need to be known or assumed
CFD [238, 281, 342, 374]	Wind circulation and sheer	Can model wind behavior in urban, cluttered environment	Computationally intensive

Figure 2.14 Filtering process for Dryden and von Kármán wind generation models.

2.3.1 Continuous Gust Modeling

Continuous gusts models are used to simulate smooth variations in wind gust behavior. They are specified in terms of a continuous probability distribution. In [292] and [122], the authors make the following assumptions to simplify the mathematical description of continuous gusts:

- The probability distribution of gust velocities is Gaussian.
- The statistics of gust velocities do not vary with time i.e.; the statistics are stationary.
- The statistics of gust velocities do not depend on the location of the quadrotor in space or path flown by the quadrotor i.e. the statistics are homogeneous with respect to quadrotor motion.
- Ensemble averages of gust velocities equal single sample time averages i.e. gusts are ergodic random processes.
- The spatially varying gust velocity field is frozen i.e. does not vary with time.
- For high altitude gusts, the statistics do not depend on the orientation of the coordinate axes and are independent of quadrotor direction i.e. the statistics are isotropic.

Dryden and von Kármán are the two most commonly used models for continuous wind gusts. They are also referred to as turbulence models. Traditionally, these models have been used to study the wind effects on aircraft however, taking into consideration the size of the quadrotor, these models are simplified by making appropriate approximations [376]. In [345], the authors implemented the Dryden models to study the effect of turbulence on miniature aerial vehicles.

Functionally, both these models are pulse shaping filters that work, as shown in Figure 2.14. A unit variance, band-limited white noise signal, is passed through a shaping function to generate an output signal with spectral properties defined by the shaping function [345]. The filter uses specific shaping functions that output a wind gust velocity signal with desired spectral properties [178, 344, 376]. Both of these turbulence models specify a power spectral density (PSD) function to define the turbulence spectra for along-wind, crosswind, and vertical wind directions.

As shown in Figure 2.15, the along-wind direction is wind flowing along the path of the quadrotor movement either in the direction of the quadrotor (tailwind) or against it (headwind). Crosswind is the wind flowing across the path of the quadrotor movement. Vertical-wind is the wind flowing in the direction perpendicular to the plane of the quadrotor movement. The three wind directions are orthonormal to each other, and the relevant variables are modeled as an independent.

Both these models are standardized with their PSD functions specified in design and simulation studies published by the U.S. Federal Aviation Administration (FAA) and the U.S. Department of Defense [228, 236]. MIL-F-8785 is another military standard with its own specifications for the Dryden and von Kármán PSD functions.

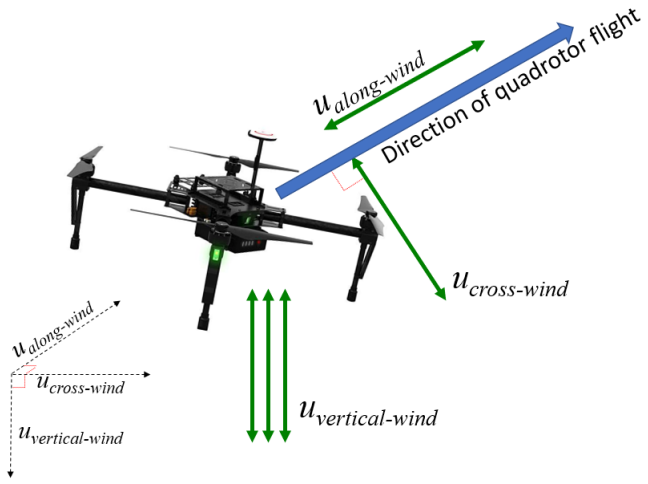


Figure 2.15 Orthonormal along-wind, cross-wind, and vertical-wind directions for a quadrotor sUAV.

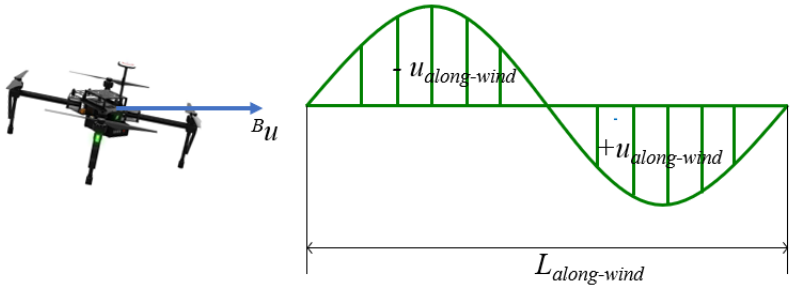


Figure 2.16 Illustration of quadrotor flying into sine-type turbulence field with scale length $L_{along-wind}$.

Readers are referred to as [236] and its references for more details on the PSD functions.

Figure 2.16 shows two important parameters of wind gust behavior, as described by the Dryden and von Kármán models. The first is the scale length $L_{along-wind}$ of the wind turbulence in the along-wind (headwind/tailwind) direction of motion. u_{a-w} is the along-wind gust speed. u_{c-w} is the crosswind gust speed, and u_{v-w} is the vertical wind gust speed. Additionally, σ_{a-w}, σ_{c-w}, σ_{v-w} are the RMS gust speeds (also called turbulence intensities) in the along-wind, crosswind, and vertical directions respectively.

Since the general flight altitudes for quadrotors does not exceed 1000 feet mean sea level (MSL), they are classified as low altitude aerial vehicles according to the US military standards [227, 236]. The approximate scale lengths and turbulence intensities in the spectral forms have been defined for an altitude less than 1000 feet MSL as (2.14), (2.15), (2.16) and (2.17) respectively. h is the altitude from sea level (in feet). The wind speed at 20 feet AGL $|u_{wind-20}|$ is set to 15, 30, and 45 knots for light, moderate and severe turbulence conditions, respectively. These values were experimentally determined in [132].

$$L_{a-w} = 2L_{c-w} = \frac{h}{0.177 + 0.000823h^{1.2}} \tag{2.14}$$

$$L_{v-w} = \frac{h}{2} \tag{2.15}$$

$$\sigma_{a-w} = \sigma_{c-w} = \frac{\sigma_{v-w}}{0.177 + 0.000823h^{0.4}} \tag{2.16}$$

$$\sigma_{v-w} = 0.1|u_{wind-20}| \tag{2.17}$$

In [376], these approximations were used to generate the turbulence wind profile for their study of wind effects on quadrotors. In [355], the turbulence intensities and scale lengths for height less than 300 meters obtained from [132] were used.

The Dryden model is used with translational velocity components. The effect of the rotational velocity disturbance components from the Dryden model is small enough to be neglected [89]. The wind gusts are limited to the translational components $[u_{a-w}, u_{c-w}, u_{v-w}]$.

Remarks on Dryden or von Kármán models: The shaping functions are obtained from the square root of the PSD functions $S_{a-w}(f), S_{c-w}(f)$, and, $S_{v-w}(f)$ of the model used (Dryden or von Kármán). As such, these must be factorizable. The Dryden PSD functions are rational and hence exactly factorizable. On the other hand, the von Kármán PSD functions are irrational and hence can only be approximated, with an exact factorization requiring a filter of infinite order [89,94,292,345]. However, the von Kármán PSD matches real wind gust turbulences found in nature more closely than the Dryden model. Thus the choice between Dryden and von Kármán models depends on the choice between engineering convenience and physical correctness. Table 2.6 notes representative studies that use the Dryden model to study several aspects of multi-rotor sUAVs. To the best of our knowledge, the von Kármán model

Table 2.6 Use of Dryden Models for Multi-rotor SUAV-based Simulations

Author	Motivation	Software
Tran [355]	Evaluate quadrotor flight performance of PID Controller	MATLAB
Massé [217]	Evaluate quadrotor flight performance of LQR controller	MATLAB
Allison [17]	Modeling quadrotor trajectory performance	MATLAB
Waslander [376]	Evaluate quadrotor position control	-

has not been used yet to study the wind effect on quadrotors. For more information about the von Kármán model, readers are referred to [94, 108, 345].

The Dryden and von Kármán turbulence models are generalized to open skies, and do not account for the presence of urban structures, where airflow around such structures is described with significant degrees of circulation and sheer [281]. These models rely on spatial and temporal scales larger than what is encountered in urban settings. The turbulence spectrum statistics are assumed to be stationary in time and space, which implies that the disturbances are induced only when the quadrotor is in motion relative to the turbulent field. The open skies and stationary assumption works well for fixed-wing aircraft. However, as is observed in [17,217,355,376], several modeling assumptions need to be made for simulating multi-rotor sUAVs in urban environments or hover flight. Wind fields in urban environments pose a challenge to the operation of multi-rotor sUAV given the complexity of wind fields generated around buildings.

Open-source Python Implementation of Dryden Model: The Python implementation of the Dryden wind turbulence model can be found in [78]. The inputs to the Dryden model are altitude and airspeed of the sUAV. As an example, turbulence wind field was generated for 10 seconds of flight time of a sUAV. The sampling rate for this particular dataset was 43Hz. Users can specify their own sampling rate in the code. Figure 2.17 depicts the magnitude of wind velocity in m/s generated in

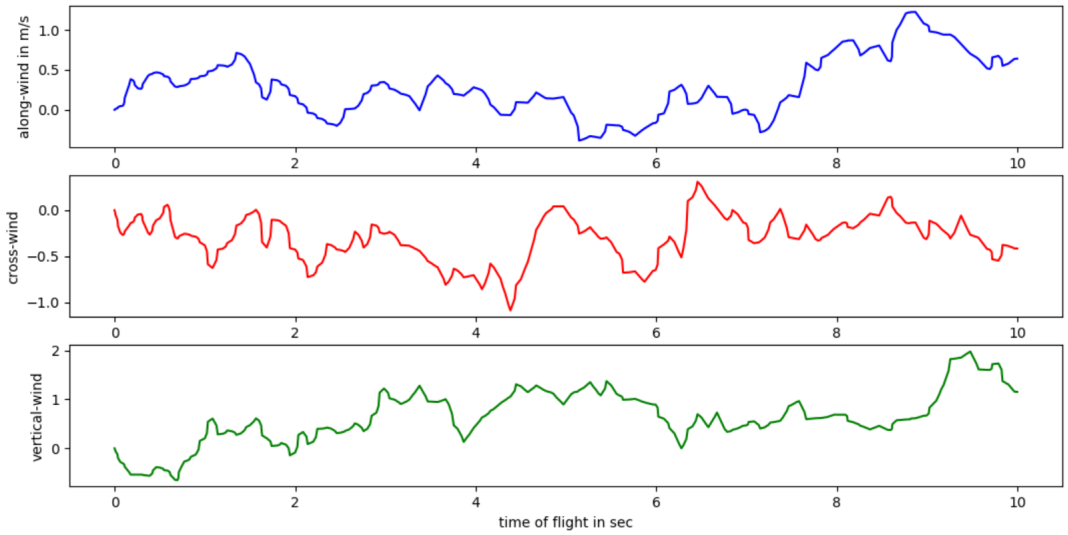


Figure 2.17 The magnitude of wind velocity in the along-wind, cross-wind, and vertical wind directions.

the along-wind (head-wind and tail-wind depending on +/- sign), cross-wind, and vertical-wind for a flight duration of 10 seconds.

2.3.2 Discrete Wind Gust Modeling

Discrete gusts are isolated changes in wind velocity and are modeled as a step function or a pulse. In [364], the authors provide a discrete gust model that considers wind as a random vector and use it to study wind effects on quadrotors. In [16], the authors use this model to evaluate the performance of the quadrotor with tilted rotors. The discrete gust model defined in [364] takes into consideration the following factors:

- The effect of wind speed change: the discrete gust model accounts for a mean wind speed value before the discrete gust (step) and also accounts for the decrease in wind speed after the discrete gust (step). Step refers to the sudden increase/decrease in wind speed from its mean value.

- Gust duration: The duration of the gust is defined as a discrete random variable d_{ni}.

- Change in wind speed with respect to altitude: The relation between the change in altitude and wind speed is described in (2.18). $|u_h|$ is the mean wind speed at altitude h. $|u_{ref}|$ is mean reference wind speed at altitude h_{ref}. p is an

empirically derived exponent that depends on atmospheric conditions.

$$|u_h| = |u_{ref}|\left(\frac{h}{h_{ref}}\right)^p \tag{2.18}$$

- Change in wind direction: The azimuth ψ_{wind} defines the direction from which the wind blows. It is measured from the north point through the east—wind direction changes at each velocity step. The wind direction values are computed at each discrete time step n. They are obtained by adding a random variable $\Delta\Psi_{wind}$ to $\Psi_{wind(n)}$ as shown in (2.19).

$$\Psi_{wind(n+1)} = \Psi_{wind(n)} \pm \Delta\Psi_{wind} \tag{2.19}$$

The gust speed $|u_{wind}|$ is described in (2.20). t_{max} represents the maximum flight time. n discrete random variable to determine the number of wind steps for t_{max}. $|u_{wind0}|$ is the wind speed before the gust (step) and $|u_{windM}|$ is the gust speed. t_0 discrete random variable to determine each gust step start. t_{ni} discrete random variable to determine duration of gust.

$$|u_{wind}| = \begin{cases} |u_{wind0}|, \quad n \leq t_0 \\ |u_{wind0}| + \frac{|u_{windM} - u_{wind0}|}{2}\left(1 - cos\left(\frac{\pi(n-t_0)}{t_{ni}-t_0}\right)\right), \\ \text{where } t_0 < t \leq t_{ni} \text{ and } |u_{windM}| \geq |u_{wind0}| \\ |u_{wind0}| + \frac{|u_{windM} - u_{wind0}|}{2}\left(cos\left(\frac{\pi(n-t_0}{t_{ni}-t_0}\right) - 1\right), \\ \text{where } t_0 < t \leq t_{ni}, |u_{windM}| < |u_{wind0}|, t \leq t_{max} \end{cases} \tag{2.20}$$

Using (2.20) and (2.19), the discrete gust speed and direction can be calculated at each discrete time instant, respectively. The model was implemented using MATLAB [364]. For more detailed discussed on how this model is applied to a quadrotor, readers are referred to [364] and its references.

2.3.3 Computational Fluid Dynamics Modeling

Wind fields are defined by the boundary layers, which is the thin layer of fluid close to the body surface. The boundary layers for urban wind fields are defined by the building structures and terrain features and are called the urban canopy layers [206, 247,342,374]. The wind field in this region is dependent on the geometry of the surface roughness as created by building size and pattern [43, 45]. The wind profile around urban structures is variable in time and space and, therefore, difficult to accurately estimate using only local measurements.

Computational fluid dynamics (CFD) has been used to determine the wind field and turbulence generated around urban buildings [271]. The approach to use CFD analysis in the urban canopy layer has been extensively applied in the field of pollutant/chemical dispersion within urban settings to determine wind loads on buildings and assess pedestrian comfort [113, 129, 175, 304, 347, 366]. CFD equations can calculate the wind speed and direction at each instance of time and space and are thus able to accurately account for the complexity derived from wind flow around urban structures. The methods used to solve the CFD equations fall into three categories. They are Large Eddy Simulation (LES), Reynolds Averaged Navier Stokes (RANS) and, Direct numerical simulation [342].

In general, due to the high computational complexity of generating wind field using CFD methods, it is challenging to generate wind field model in parallel with the flight simulation [110]. As such, CFD solvers are used to generating wind speed and heading values and store them in a database. The wind velocities stored in the database can then be used as inputs to the flight simulation setup to simulate wind turbulence experienced by sUAVs [110, 238, 281, 342].

In [374], the authors use a CFD equation solver presented in [115] called QUIC-CFD. This solver is a lightweight implementation of the Navier Stokes equation solver and is capable of generating wind field estimates of large, complex, urban

environments. The wind field data generated by this solver was used to simulate wind disturbances in a multi-rotor sUAV flying in the urban canopy layer. The authors in [374] noted that wind fields around urban structures are governed by environmental features that are defined by a length scale of less than $1km$. The general flow pattern of wind is still defined by the prevailing wind in the upper boundary layers. This dependency of wind around urban structures on the upper boundary layers was leveraged as inputs to the QUIC-CFD solver to determine the turbulent wind flow around such structures. The QUIC-CFD solver prioritized solution time over the accuracy and provided quick estimates of the solution to the Reynolds Averaged Navier Stokes equation. The estimates of the QUIC-CFD model were compared to wind data collected around the university campus. The authors noted an error of less than 10% for wind speed errors. Heading errors were observed to be less than 15 degrees for 36% of the readings.

In [238], the authors generated a wind field model around a twin building configuration to study the wind effects on a multi-rotor based sUAV. The CFD solver used the RANS method implemented in CFD software ANSYS FLUENT and STAR-CCM+. The dynamics of the multi-rotor sUAV was shown to be affected by the turbulent wind conditions. It was observed that a spike in the wind speed value could cause a sudden increase or decrease in the thrust of the aerial vehicle.

The authors in [342] used the Large Eddy Simulation (LES) technique to develop their CFD model for wind turbulence around a single square body. The square body was representative of a building around which the wind flow was analyzed. Two essential features captured by their wind model were vortex shedding and flow instabilities. Vortex shedding is the property of wind to produce an oscillatory flow with symmetric vortexes when flowing around a body opposing its flow. At higher wind speed, the vortexes lose their symmetry and create flow instabilities. The LES technique is a time-dependent three-dimensional technique that also allows for the

visualization of the wind wake behind a body (vortex shedding and flow instability), which is not possible with the RANS technique [353]. An open-source CFD software Open-FOAM (Open-Field Operation and Manipulation) was used to implement the LES technique to build the wind model around a square building. The wind profile generated using the LES technique was compared with wind data as obtained through outdoor experiments carried out in [224]. In [224], wind profile measurements were obtained around a square building at four locations. The average error was computed through the mean of errors between experimentally obtained data and numerical value at the same spatial location. The average error for each location was found to be 12.9%, 17.6%, 9.7%, and 17.4%.

Remarks on CFD based models: The time required for CFD based numerical computation of the wind model depends on the resolution and domain (geographical area) size needed. A large domain size combined with higher resolution could result in a simulation time in the order of days to weeks. At the same time, with the rising use of multi-rotor sUAVs, it is evident that such vehicles are being increasingly used in urban settings and that CFD based wind generation models are capable of generating real-world urban wind conditions.

2.3.4 Comparison and Future Directions

Three approaches to modeling wind for simulation purposes are discussed. In the following, we compare these methods and provide future directions.

- Mission type: Simulation mission type: The discrete gust model provides a limited scope to simulate wind disturbances as it only considers intermittent wind gusts and does not account for the continuous variation in wind speed and direction. The Dryden and von Kármán turbulence models are suitable for simulating multi-rotor sUAV missions in large open areas as they do not account for infrastructures such as buildings and trees. Additionally, in the Dryden and von Kármán turbulence models, wind disturbances are assumed to be stationary in time and space. The disturbances are applied when the

sUAV is in motion relative to the simulated wind field. Thus, under hovering conditions, the Dryden and von Kármán turbulence models are not applicable. For simulations in urban or cluttered environments, CFD based simulations should be used as these can capture circular wind flow with varying shear around trees and urban structures.

- Simulation complexity: CFD based simulations are time-consuming and require higher computational resources relative to the Dryden and von Kármán turbulence models. As noted earlier, CFD based simulations have been implemented using specialized numerical solvers, whereas the Dryden and von Kármán turbulence models can be implemented using popular programming languages such as Python and MATLAB. For both techniques, wind speed and direction data for each instance of time and space are first generated and stored in a database. This data is then used in multi-rotor sUAV flight simulation. Moving forward, it is expected that CFD based simulation solvers will be optimized to use the computational power of graphical processing units (GPUs) to reduce simulation times required for generating high-quality wind data.

2.3.5 Note about Physical Wind Generation

From a physical wind generation perspective, a low-cost and relatively easier method to generate wind in physical testing is using electric fans. In [15, 371], different brands and configurations (numbers, positions) of electric fans have been used to generate wind disturbances. In several studies, wind tunnels have also been used to evaluate flight performance in multi-rotor sUAVs [258, 307, 323]. Wind tunnels provide strong controls over the experiment. At the same time, they are relatively expensive and require significant instrumentation. In some studies, outdoor flight tests expose the multi-rotor sUAV to environmental wind disturbances, which provide flight performance results without the need for expensive wind tunnels or industrial grade fans [112, 324].

2.4 Conclusion

As multi-rotor sUAVs continue to gain popularity, research to study and mitigate wind effects on their flight performance continues to grow. In this chapter, we presented a

systematic review of two important dimensions of wind research in multi-rotor sUAVs viz. measurement/estimation and simulation.

For wind measurement/estimation, a review of three key techniques was presented. The first technique uses flow sensors, the second technique uses ultrasonic anemometers, and the third technique, called the "tilt angle method," uses the onboard IMU. For wind simulation, a review of three modeling techniques was presented. The first technique adapts the popular Dryden wind turbulence model for multi-rotor sUAV operational regimes in open and uncluttered environments. The second technique uses computational fluid dynamics (CFD) for simulating wind behavior in urban and cluttered environments. The third technique uses a probabilistic approach to generate wind gusts.

Each of these techniques was presented in terms of their operating principles, representative studies, key benefits, and operating challenges. Open-source Arduino code to interface with the FT205 ultrasonic anemometer was provided for readers to quickly prototype quadrotor applications. Open source Python code to simulate the Dryden wind gusts was provided for readers to incorporate in their simulations. We anticipate that the review presented in this chapter will provide important information for researchers and practitioners in their efforts.

CHAPTER 3

IMPLEMENTATION OF DECENTRALIZED REINFORCEMENT LEARNING-BASED MULTI-QUADROTOR FLOCKING

3.1 Introduction

Paper: *P. Abichandani, C., Bucci, D., Mcintyre, W., and Lobo, D. "Implementation of Decentralized Reinforcement Learning-Based Multi-Quadrotor Flocking," in IEEE Access, 9, 132491-132507, 2021*

The problem of coordinated movement of the sUAVs between waypoints while maintaining a set of desired kinematic behaviors and avoiding collisions has been well studied using model-based techniques [4, 222, 223, 372]. Although recent works demonstrate decentralization can be achieved given online construction of kinematically feasible trajectories, decentralized path planning at scale remains an area of active research [5, 197].

A popular alternative to these model-based methods involves emergent control techniques based on multi-agent flocking [341]. The flocking problem was initially introduced by Reynolds in [285] and later elaborated on by Vicsek in [363], and Couzin in [72]. Flocking is a set of bio-inspired rules – separation, alignment, and cohesion – that define the characteristics of multiple sUAVs interacting with one another and exhibiting common collective behavior. One of the defining characteristics of flocking is that each agent performs localized actions that contribute to an overall coordinated behavior of the system. Separation involves agents moving away from each other to avoid collisions and is also termed collision avoidance. Alignment involves agents moving along the average heading of the flock. Each agent in the system exhibits the same relative velocity. Cohesion involves agents staying near the average position (i.e., centroid) of the group.

Wind conditions being stochastic in nature, play a crucial role in the proper functioning of coordinated sUAV operations outdoors. An overview of the impact of external stimuli such as wind gusts on sUAV operations is provided in [6]. Performing wind-agnostic motion planning for sUAVs may produce a sizeable cross-track error if the wind on the planned route leads to actuator saturation [118,259]. In a multi-sUAV system, each sUAV has to locally counter the wind disturbance while maintaining the safety of the system [392]. Such continuous manipulation of the control effort for multiple sUAVs under uncertain environmental conditions is computationally taxing and can lead to reduced efficiency and safety concerns [6].

Several works in the literature propose approaches to solve multi-agent system flocking problems [62, 123, 187, 328, 389]. In our previous work, we demonstrated a novel reinforcement learning technique using the multi-objective formulation of the decentralized flocking control problem for up to 25 fixed-wing UAVs [330]. The training was based on a relative state-space construction of obstacles and waypoints in software simulation. In comparison to existing behavioral swarm controllers, our controller learned via object-focused greatest mass state-action-reward-state-action (OF-GM-SARSA) and was shown to be generalizable across multiple flocking scenarios.

While meaningful experiments on of flocking algorithms on ground robots have been reported [357], those for sUAVs have emerged only recently. A select few papers document real-time experiment evaluation of their flocking system [73,127,170,205,357,360]. Only [174] reports on accounting for wind using a simplified Gaussian noise model. Most other publications provide an evaluation of their flocking approaches in the multi-sUAV simulation environments such as Ardupilot, Q-ground control, Gazebo, and ROS [92, 133, 204] or numerical simulation using Python and MATLAB [123, 171, 186, 187, 322, 339] thus leaving a gap in the literature regarding

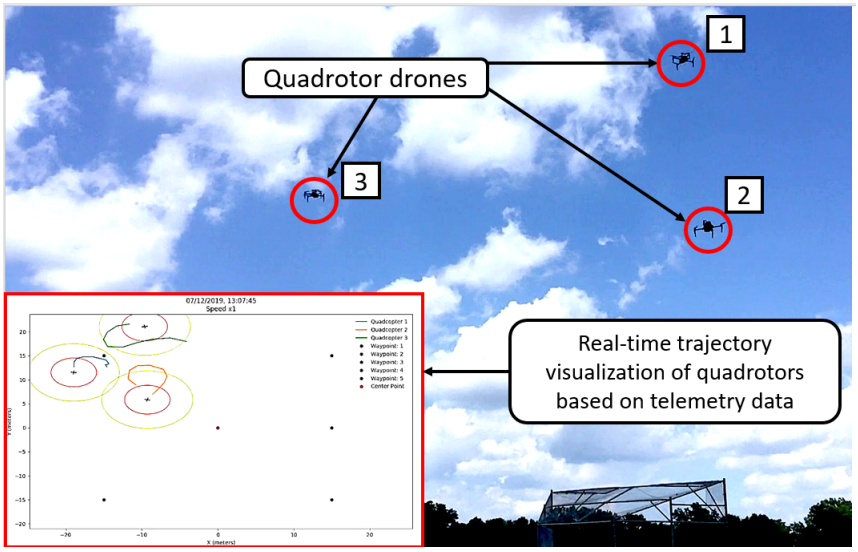


Figure 3.1 Three quadrotor OF-GM-SARSA outdoor flight experiment and telemetry display.

the hardware/software approaches required for implementing flocking based motion planners in real-world outdoor flights.

3.1.1 Main Contributions

This chapter fills the above-mentioned gap by focusing on the relatively untouched area of experimental implementation of flocking algorithms for outdoor multi-sUAV systems. Specifically, in this work, we leverage our previously developed OF-GM-SARSA-based path planner for flight testing the coordinated motion of multiple quadrotors to reach waypoints while maintaining the flocking behaviors. A snapshot of the outdoor flight tests is shown in Figure 3.1. The following are the main contributions of this chapter:

1. A method to incorporate realistic wind gusts using the Dryden Wind Gust Model for evaluating flocking algorithm performance in hardware-in-the-loop (HWIL) tests.
2. Experimental evaluation and validation of a decentralized OF-GM-SARSA based hardware/software architecture via outdoor flight tests involving up to 4 DJI M100 quadrotors operating in the presence of natural wind gusts.

Additionally, a detailed discussion of the hardware/software architecture used to implement a multi-objective, reinforcement learning (RL) based decentralized flocking planner for multiple quadrotors is provided. The OF-GM-SARSA technique was used to learn a flocking planner for the group of up to 4 DJI Matrice M100 quadrotors that use DJI's N1 flight controller [87]. Each quadrotor was also fitted with a transceiver radio module operating in the 915MHz license-free frequency bands, providing half-duplex bi-directional RF links at 300Kbps. The OF-GM-SARSA planner used the quadrotors position and velocity information, as reported by the DJI telemetry streams of the one-hop neighbors. The output of the planner was local heading and speed setpoints to each quadrotor's flight controller.

The rest of the chapter is divided as follows, Section 3.2 presents a discussion on the representative works in model-based and reinforcement learning-based flocking approaches. Section 3.3 discusses the flocking problem formulation in terms of the quadrotor dynamics, the inter-quadrotor communication setup, the Dryden wind model, and the flocking behaviors. Sections 3.4 and 3.5 presents the algorithm for the flocking planner and the methodology used for training the RL algorithm. The algorithm was first evaluated in simulations described in Section 3.6 before HWIL and outdoor flight tests. The hardware and software architecture for HWIL and outdoor flight tests are described in Section 3.7, followed by the flight test results in Section 3.8. Finally, the observations and insights gleaned from this study are noted in Section 3.9.

3.2 Related Work

The aggregate motion of small birds is known as cluster flocking in the literature [35]. In such cluster flocking applications, each individual bird may only observe the neighboring bird at any time. The flocking formation considered here is, in essence, inspired by flocks of birds where each agent (bird or sUAV) aggregates around a

geometric centroid and is independent of a leader agent. The bio-inspired behaviors implemented here posits that each learning agent strives to maintain the integrity of the flock using its sensory input (in this case, the position and velocity of its neighbors). While there exists a mobile ad-hoc network (MANET) for wireless communication of sensor data, the execution of the motion-planning policy is fully decentralized, where each agent runs an identical copy of the flocking policy to achieve a common goal [306]. In contrast to centralized approaches, such decentralization of motion planning offers computational tractability as the number of agents increase and eliminates a single point of (centralized decision maker) failure [2, 202].

Several centralized and decentralized approaches to solving the multi-robot flocking problem have been reported in the literature. These approaches can be broadly classified into model-based flocking methods and model-free methods.

3.2.1 Model-based Flocking

Model-based approaches to solving the multi-vehicle flocking problem continue to be studied in the literature [128, 133, 174, 205, 328, 389]. These approaches involve formulating the kinematic and dynamic behaviors of the system in the environment.

In a multi-sUAV system, the environment can include obstacles and environmental disturbances such as wind. The authors in [174] modeled wind disturbances as Gaussian noise, a simplified approximation of real-world turbulence. In [133], the authors defined obstacles as special agents, enabling sUAVs to effectively evade them by avoiding collisions while maintaining the flocking formation.

Distributed or decentralized model predictive control (MPC) based formulations to the multi-sUAV flocking problem have been a popular approach due to their ability to accommodate different systems, mission requirements, and low computational overhead [205, 274]. In [205], the authors evaluated a consensus-based MPC approach for flocking experimentally using 5 Crazyfile 2.0 mini-quadrotors in an indoor

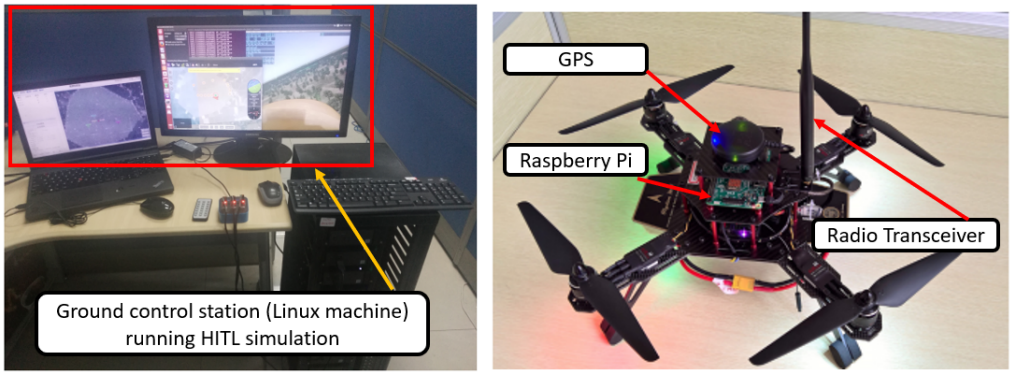


Figure 3.2 (Left) Multi-sUAV simulation software Qground control and Ardupilot running on Linux machine [62]; (Right) Custom built quadrotor for flocking experiments [274].

environment. The authors of [274] tested a decentralized MPC algorithm using a flock of five outdoor quadrotors. Figure 3.2 depicts a custom-built sUAV used for flocking experiments in [274].

Additional control-theoretic approaches such as high-frequency feedback robust control [389], Particle Swarm Optimization (PSO) [73], and PID controllers [170] have also been applied to solve the multi-sUAV flocking problem.

Model-based flocking approaches rely on the accuracy of the model of the multi-vehicle system. The accuracy of the model decreases when the number of agents in the system increases. The modeling becomes especially challenging if the real-world characteristics of wind disturbances are also taken into consideration.

3.2.2 Reinforcement Learning (RL) based Flocking

Among the model-free methods, RL-based flocking approaches have been presented in several studies as a means for overcoming platform and environmental modeling restrictions while maintaining decentralized operations [60, 130, 145, 198, 257, 273].

Q-learning and state-action-reward-state-action (SARSA) have been implemented in multi-vehicle robotics problems because they do not require modeling the complicated flight dynamics of the system. In [62], the authors used a SARSA-based

approach to successfully tackle the enemy sUAV avoidance problem in multi-sUAV systems. Figure 3.2 depicts the hardware-in-the-loop simulation setup used to evaluate their SARSA-based collision avoidance algorithm for multi-sUAV systems. In [172], the authors simulated a Q-learning-based approach for search-and-rescue operations using sUAVs. They considered an indoor scenario where the sUAV relies on RF signals emitted by a smart device owned by the target victim. In [391], the authors proposed a SARSA-based approach to deploy a multi-sUAV assisted wireless network. The goal of the reinforcement learning was to enable the sUAVs to learn the features of the environment and plan trajectories accordingly to provide wireless service in disaster-hit areas.

One of the differences between Q-learning and SARSA is the policy based on which rewards are designated. In Q-learning, the reward policy is greedy and always fixed to favor the maximum achievable reward. In SARSA, the policy changes are based on the current state and action pair, favoring a more optimal solution than Q-learning. Few authors have provided a comparison of Q-learning and SARSA-based approaches for multi-UAV flocking [242, 322]. In [242], the authors compared Q-learning and SARSA to reduce power consumption for multi-sUAV systems in the presence of wind. Simulations were carried out using a combination of ROS and Gazebo. The wind field was simulated using the physics engine in Gazebo and actual wind data. Both RL approaches were compared to a naive planner that selected the shortest paths irrespective of the wind fields at each time step. The results showed that the RL approaches reduced the power consumption by about 30% compared with the naive planner. In [322], the authors provided a comparison of Q-learning and SARSA for global path planning for mobile ground robots using Python simulations. It was observed that while the Q-learning emphasized the minimum number of actions necessary to reach the goal, the SARSA algorithm gave priority to security and found the optimal safe distance that avoided the risk of collisions/accidents. Therefore,

Q-learning showed faster convergence rates, and SARSA provided a safer path for the ground robot.

Neural networks (NNs) have been proposed to produce generalized results that take into account the dynamic behavior of the flight environment [186, 204, 279]. In [279], the authors simulated a centralized, deep-Q-learning based leader-follower approach to solving the flocking problem for multi-sUAV systems using the Hungarian algorithm [150]. The proposed algorithm showed feasible convergence times for different flocking formations such as circle, v-shape, and star (12 ms). However, the algorithm took a significantly longer time when the number of sUAVs increased to 100. In [204], a deep-SARSA approach involved combining the traditional SARSA algorithm with a NN instead of Q-tables for storing the states and predicting the best action. The NN used was implemented in Keras and contained three dense layers with 549 trainable parameters [67]. Data was generated based on the training process, which was executed for 4000 simulation runs. The trained model was then successfully evaluated in a simulation testbed built using ROS and Gazebo.

In [186], the authors provided a deep-RL approach for sUAV ground target tracking in the presence of an obstacle. They used a deep deterministic policy gradient (DDPG) to generate the path plan around large-scale and complex environments. Simulation experiments were carried out in TensorFlow 2.0 using Python [1]. It was observed that the improved DDPG algorithm improved the success rate for target tracking from 70.0% to 91.8% in the sparse environment and 13.6% to 67.5% in a dense environment. In [368], the authors formulated the fixed-wing sUAV flocking problem as a Markov Decision Process (MDP). A NN was used to train the model using TensorFlow and Keras, with the number of training episodes set to 50000 and a maximum time step of 30 seconds. The authors also implemented a hardware-in-the-loop simulation using an X-plane flight simulator and two PX4 Pixhawks. The flight simulator modeled weather changes and wind disturbances. The flight simulation

showed that the proposed RL algorithm could deal with environmental changes and completed the simulation mission.

As observed, a majority of these studies are limited to simulations. By contrast, this chapter provides detailed discussions on the hardware/software implementation and validation of OF-GM-SARSA applied to a multi-sUAV system to learn flocking using HWIL and outdoor flight tests.

3.3 Problem Formulation

In this section, the key elements of the problem formulation are elucidated.

3.3.1 Quadrotor Motion

Consider a group of n quadcopter vehicles with 6 degrees of freedom operating inappropriately defined right-handed inertial, body, and body-fixed frames of reference. The non-linear rigid body dynamics of quadrotors have been well studied and documented [209].

Each quadrotor $i = 1, \ldots, n$ has a given start (origin) point o^i and a given end (goal) point e^i. O is the set of all start (origin) points. $o^i \in O, i = 1, \ldots, n$. E is the set of all end points. $e^i \in E, i = 1, \ldots, n$. The Euclidean distance between two quadrotors i and j is denoted by d^{ij}, $i = 1, \ldots, n, j \neq i$.

3.3.2 Communication Model

Each quadrotor is equipped with a mobile ad-hoc network (MANET) radio used to exchange its trajectory information with other quadrotors. The communication connectivity is maintained directly (one-hop) or indirectly (multi-hop relaying). The quadrotors are modeled to communicate if they are within the η_d distance of one another. This constraint is equivalent to the Signal to Noise Ratio (SNR) experienced by the receiver quadrotor being above a threshold [3].

3.3.3 Dryden Wind Model

Wind gusts are modeled using the continuous Dryden turbulence model [122]. The Dryden wind turbulence model has been commonly used to model continuous wind gusts effects on large-scale aircraft. Since quadrotor sUAVs are significantly smaller in size compared to such aircraft, appropriate approximations have been reported [345, 376]. Detailed information on the Dryden wind gust modelling is described in Chapter 1.2 Section 2.3.1.

3.3.4 Flocking Behavior

The rules for multi-agent flocking behavior have been defined according to Reynolds flocking rule set. These flocking rules were encapsulated as reward functions that motivated the agents to exhibit cohesion, alignment, separation, target seek, and obstacle avoidance. Similar to our previous work [330], the thresholds for each reward were selected based on running a uniform search (sweep) on the parameter space and evaluating the reward curves.

- Cohesion (r_{COH}): allows the flock to move inward towards an estimated centroid position. The cohesion objective seeks to minimize the distance between a quadrotor sUAV and its one-hop neighbors while maintaining an inter-sUAV separation distance k_{sep}. The flock radius was set proportional to the square root of the flock size. States were given a reward of $r_{\text{COH}} = +1$ for staying within k_{sep} centered at the flock centroid. r_{COH} was equal to -1 otherwise.

$$r_{\text{COH}} = \begin{cases} +1, & d^{ij} < k_{sep} \\ -1, & otherwise \end{cases} \tag{3.1}$$

- Alignment (r_{ALN}): allows the quadrotor sUAV in the flock to match their velocity headings such that each member of the flock moves as a single unit. The velocity alignment objective seeks to minimize the velocity heading between the local quadrotor sUAV and the average heading of its one-hop neighbors. States were given a reward r_{ALN} within the interval of $[-1; +1]$ as a function of velocity heading difference which was calculated using the following relationship:

$$r_{\text{ALN}} = 2 \cdot \left(0.5 - \frac{1}{\pi \Delta\theta} \right) \tag{3.2}$$

where $\Delta\theta$ is the heading difference in degrees between the platform and the average heading of the flock.

- Separation or Collision Avoidance (r_{COL}): provides a minimum safe distance k_{col} between the flock members such that they avoid collision with each other. Two levels of negative rewards are defined based on the degree of k_{col} violation. The agent is severely penalized when it is within 0.2 distance units of a neighbor and is mildly penalized when they are 1.5 distance units. This differentiation was necessary since the agents might come close to one another due to external disturbances. However, extremely close distances should be avoided under all circumstances. States were given a reward of $r_{\text{COL}} = -1$ for getting within 1.5 distance units of a neighbor, $r_{\text{COL}} = -100$ for getting within 0.2 distance units of a neighbor, and $r_{\text{COL}} = 0$ otherwise.

$$r_{\text{COL}} = \begin{cases} -1, & d^{ij} < 1.5 \cdot k_{col} \\ -100, & d^{ij} < 0.2 \cdot k_{col} \\ 0, & otherwise \end{cases} \tag{3.3}$$

- Target Seek (r_{TGT}): The target seek objective seeks to minimize the distance between a local quadrotor sUAV and the current waypoint of the flock. It allows for waypoint tracking of the flock. The target seek reward is defined such that the agents reach the waypoints as quickly as possible [385]. The quadrotors are considered to have reached a waypoint if they are within a distance k_{reach} from that waypoint with a reward r_{TGT} as follows:

$$r_{\text{TGT}} = \begin{cases} +10, & d < k_{reach} \\ -1, & otherwise \end{cases} \tag{3.4}$$

- Obstacle Avoidance (r_{OBS}): This objective was formulated to ensure safe operations in an environment with obstacles. The quadrotors were rewarded based on their distance d^{io} from an obstacle and an obstacle avoidance threshold k_{obs} as follows:

$$r_{\text{OBS}} = \begin{cases} -1, & d^{io} < 3 \cdot k_{obs} \\ -100, & d^{io} < 1 \cdot k_{obs} \\ 0, & otherwise \end{cases} \tag{3.5}$$

3.4 Object Focused Multi-Objective SARSA Flocking Planner

Modular Q-learning is used to model multiple flocking behaviors as a composite Markov Decision Process (MDP) with each behavior being treated as a distinct module, $m \in M$, where $M \in \{\text{COH, ALN, COL, TGT}\}$ [30, 298, 331]. A single Q-table characterizes each module m. The optimal policy selects the action with the largest weighted sum of Q-values across all Q-tables. During training, the Q-tables are updated using the SARSA algorithm. The overall procedure described here was elucidated previously in [330].

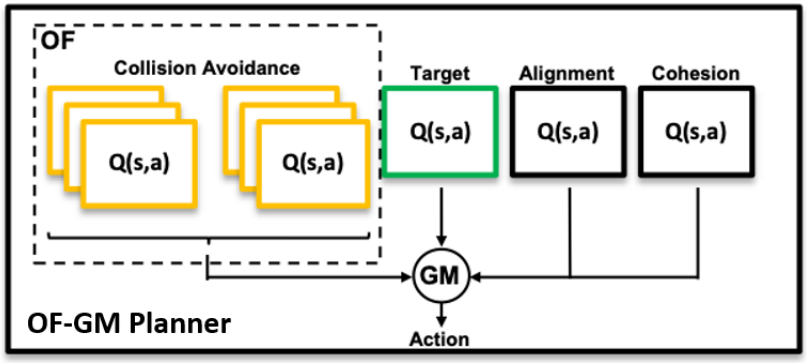


Figure 3.3 Graphical description of OF-GM-SARSA planner.

Figure 3.3 depicts the graphical representation of the OF-GM-SARSA planner, where s and a denote the state and action, respectively. Each flocking behavior is described as an objective represented by a Q-table. The collision avoidance objective is represented by multiple copies of a single Q-table per inter-sUAV pairing to avoid the exponential growth in the size of the Q-table. The greatest mass operation involves selecting the maximizing action under a weighted average of the corresponding Q-values.

3.4.1 State Space Representation

For a given sUAV, the states included in the state space representation of the OF-GM-SARSA planner are estimates of relative position (i.e., range and bearing) and velocity heading of neighboring sUAV. Let the state $\overline{s}$ denote the total collection of these measurements for each module,

$$\overline{s} = \bigcup_{m \in M} s^{(m)} \tag{3.6}$$

Figure 3.4 depicts an example of the state-space construction for four neighboring quadrotor sUAVs, with one waypoint. The $s^{(COH)}$ states are quantized distance and bearing estimates of the sUAV with respect to the estimated centroid of all

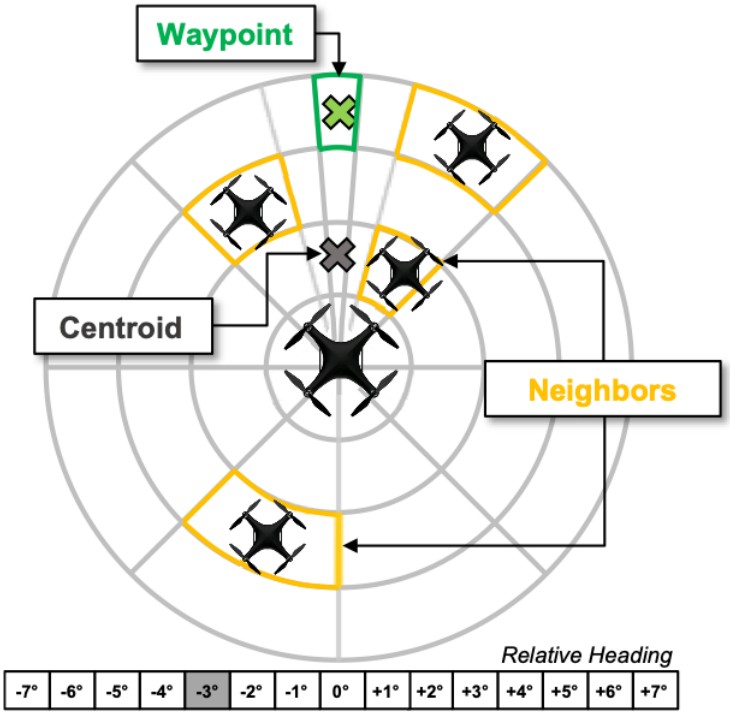


Figure 3.4 Local state space discretization for each sUAV.

neighboring sUAVs. The $s^{(TGT)}$ states are quantized distance from the currently assigned waypoint and bearing towards it. The $s^{(COL)}$ states are quantized distance, bearing, and velocity heading difference measurements between a given sUAV and its neighboring sUAVs. Finally, the $s^{(ALN)}$ states are quantized velocity heading differences between the sUAV's current heading and the average heading of all neighboring sUAVs.

The object-focused formulation of the learning problem further decomposes the collision avoidance states into states for each sUAV,

$$s^{(COL)} = \bigcup_{o \in O^{(COL)}} s_o^{(COL)} \tag{3.7}$$

where $O^{(COL)}$ is the set of neighbor sUAVs.

A single Q-table was learned for the collision avoidance behavior and shared across all sUAVs. The control policy was evaluated based on the separation distances queried from this Q-table. Each sUAV has a different inter-sUAV distance, and therefore receives different rewards but as specified from the single learned Q-table. A linear sum of the learned Q-values was then performed to get a single composite Q-value for the collision avoidance criterion. The measurement discretization used was specific to each module. The distance partitioning for the collision avoidance module was set to higher resolution at closer ranges. The target seeks module depended on long-range distances, and its discretization was uniform.

3.4.2 Action Space Representation

The action space consisted of discrete velocity setpoints for the DJI N1 flight stack. The heading angles were quantized to be within sufficiently smooth yaw rates achievable by the DJI M100 quadrotor and N1 flight stack. The discretized action space A for small roll and pitch angles is represented as the Cartesian product of the heading angles (degrees) and the speeds (m/s) as follows:

$$A = \{0^\circ, \pm 10^\circ, \pm 20^\circ, \pm 30^\circ, \pm 40^\circ\} \times \{0.1, 0.5, 1\} \tag{3.8}$$

3.4.3 OF-GM Policy

The OF-GM policy uses the learned Q-tables per module, denoted $Q_m(s_o^{(m)}, a)$ to compute a total weighted Q-value, denoted $Q_{GM}(\overline{s}, a)$. For a given sUAV's state, $\overline{s}$, this weighted Q-table is calculated for each action $a \in A$ as

$$Q_{GM}(\overline{s}, a) = \sum_{m \in M} \sum_{o \in O} w_m Q_m(s_o^{(m)}, a), \tag{3.9}$$

Initialize Q(s, a);
repeat
  Initialize quadrotor sUAVs, obstacles, and waypoints;
  repeat
    for *each quadrotor sUAV in swarm* **do**
      Take action a;
      Observe next state, s;
      Collect the total reward;
      Get next a using softmax exploration;
      for *each module m* **do**
        for *each object o* **do**
          $Q_m(s_o^{(m)}, a) \leftarrow$
          $Q_m(s_o^{(m)}, a) + \alpha\left[r_o^{(m)} + \gamma_m Q_m(s_o^{(m)}, a) - Q_m(s_o^{(m)}, a)\right]$
        end
      end
      $s \leftarrow s$;
      $a \leftarrow a$;
    end
  until *each time step*;
until *for each iteration*;

Algorithm 1: OF-GM-SARSA training procedure

Where the module weights, w_m, sum to one. These values were set manually. Cohesion and alignment were weighted less than the other three modules.

3.5 State Exploration and Model Training

The OF-GM-SARSA training procedure is shown in Algorithm 1. After the quadrotor sUAVs and waypoints are initialized, each quadrotor sUAV selects an action according to the softmax exploration rule. This exploration rule uses the progressively learned Q-values from each table to compute the total weighted Q-value, from (3.9). The actions are then sampled using the softmax probability mass function,

$$P(a|\overline{s}) = \frac{e^{Q_{GM}(\overline{s},a)/T}}{\sum_{a \in A} e^{Q_{GM}(\overline{s},a)/T}}, \tag{3.10}$$

where larger values of T encourage exploration. State-space updates are applied sequentially to each quadrotor sUAV, and the following action is chosen according

to the same on-policy exploration rule. During training, only discrete actions were used. The Q-table for each module was then updated using SARSA. This process was repeated until the sUAVs reach the waypoint or until a maximum time was reached.

3.5.1 Training Methodology

The learning rate, α, is the same for all Q-table updates for each flocking condition, whereas the discount factors, γ_m, are unique. The discount factors decide how much the future outcomes influence the behavior of the learning rule for each module. Target seeks, for example, requires information over a longer horizon. Other modules, however, are more reactive, making immediate rewards more important. Specifically, the collision avoidance discount factor was set to 0, and the cohesion and alignment discount factors were set to 0.01. The discount factor for target seek was set to 0.9.

The task of multi-agent flocking consists of non-convex criteria [385]. As such, employing a training method that is efficient is imperative. In [37], the authors showed that starting the training procedure with easier examples of a learning task followed by a gradual increase in the difficulty improved the speed of convergence and the generalizability of the results. Similarly, the training process of modular Q-learning is faster when working with subsets of modules rather than all at once. As such, the training procedure employed in this study consisted of three stages with increasing levels of task difficulty. The training was executed using the 6-Degrees of Freedom (DOF) quadrotor rigid body dynamic model.

First Stage: A single quadrotor sUAV was used during the first stage to learn the Q-table for the target seek objective with the OF-GM-SARSA target seeks objective weight set to 1.0. A quadrotor rigid body dynamic simulation was implemented that was tuned to match the specific construction of the DJI M100. The starting location of the sUAV and the waypoint were randomized uniformly during each training episode/iteration.

Second Stage: In the second stage, a fixed set of 25 sUAV was used to learn the Q-tables for the flocking objectives (i.e., collision avoidance, velocity alignment, and cohesion). The OF-GM-SARSA training weights for this second stage were set to 0.40 for the collision avoidance objective, 0.20 for the velocity alignment objective, and 0.40 for the cohesion objective.

Third Stage: In the final stage of training, the target seeking and flocking training procedures were combined to update the Q-tables of all four objectives jointly. The training weights used in the third stage were 0.40 for the collision avoidance objective, 0.40 for the target seek objective, 0.10 for the velocity alignment objective, and 0.10 for the cohesion objective.

One thousand training iterations were used for each of the first two stages, whereas the third stage used ten thousand training iterations. Q-table updates were performed at the end of every time step of the simulation. A training episode ended when the sUAV reached the waypoint or a maximum elapsed time for the first and third training stages. For the second training stage, a training episode ended after a fixed maximum elapsed time. The learning rate was kept fixed at 0.20 during all three stages of training.

Moving-average estimates of the reward curves observed during training are shown in Figure 3.5. The figure demonstrates the convergence of the OF-GM-SARSA learning algorithm across all three stages, with slower convergence in the third stage due to the complexity of the flocking and target seek tasks.

3.5.2 On Convergence of OF-GM-SARSA

The OF-GM-SARSA algorithm is used to develop a behavioral flocking controller with a defined set of component rules that govern selection of control output (e.g., acceleration, heading) in outdoor windy environments based on the relative position and velocity differences between neighboring sUAVs [31]. Since the sUAVs operate

in a dynamic and stochastic environment, there is no guarantee of convergence to a globally optimal policy. Singh et al. in [325] proved that the SARSA learning algorithm converges to an optimal Q^* if the policy is greedy in the limit of infinite exploration (GLIE). GLIE implies a learning policy in which, eventually, the probability of selecting the optimal action over a random action becomes 1 (greedy). This requirement can be met with both the ϵ-greedy and the Boltzmann (softmax) exploration. Russell and Zimdars [298] then extended this result to the use of an arbitrator and local Q-functions (modular GM-SARSA), in that, given that the arbitrator satisfies GLIE, individual module updates Q_m converge to optimal Q_m^*, and then converge to a global optimum Q^*. Cobo et. al. [71] addressed OF

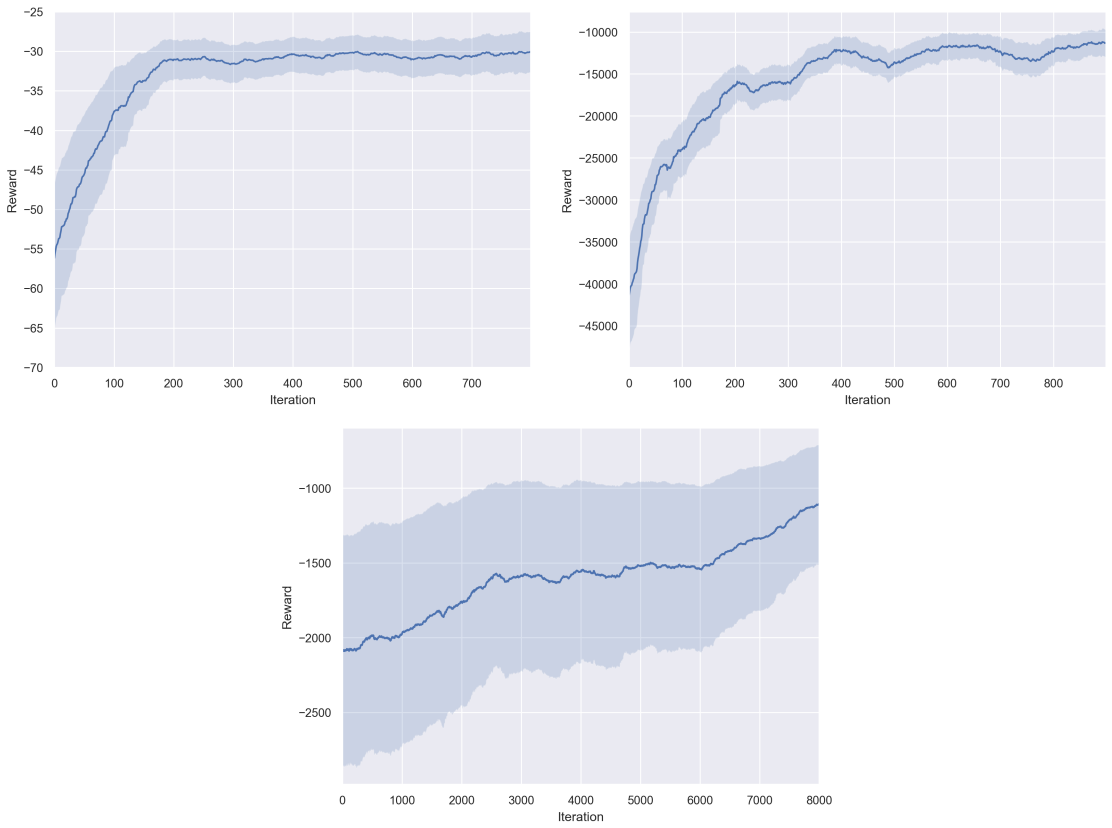


Figure 3.5 Moving average and standard deviation of returned reward over all training iterations with sliding window size of 200 from each of the three training stages. Stage 1 training reward (target seek – top-left). Stage 2 training reward (flocking – top-right). Stage 3 training reward (all objectives – bottom).

Q-learning, where they demonstrated that OF Q-function estimates Q_o converge to the true Q-functions, Q_o^*, in that several objects in the same class can be seen as independent episodes of the same Markov Decision Process (MDP). These results assumed a static and deterministic state space.

The first and foremost goal of the training was to obtain Q-values that maintain generalizability and ensure that the sUAVs maintain safe flight formation in the presence of realistic stochastic disturbances. Figure 3.5 depicts the sliding window of the mean and standard deviation for the average returned reward values per iteration for each training phase. As observed in Figure 3.5, the first two phases demonstrate strong convergence. In the third stage, when training for all objectives, the mean of the rewards obtained demonstrates slow but sufficient convergence indicating that the policy is learning the desired behaviors. Note that some variation in reward during training is also due to randomized scenarios with respect to initial distances from the target in addition to quantity and location of obstacles per iteration.

3.6 Simulations using Dryden Model

The OF-GM-SARSA flocking controller was evaluated with and without wind disturbances in Python simulations. The wind disturbances were modeled using the Dryden wind turbulence model discussed. Figure 3.6 depicts an example of the wind speed in m/s generated in the along-wind (headwind and tailwind depending on +/- sign), crosswind, and vertical-wind for a flight duration of 10 seconds. The sampling rate for this particular dataset was 43Hz. The wind turbulence generated was for a quadrotor sUAV operating at an altitude of $5m$ above the ground and moving with an airspeed between $0.25m/s$ to $1m/s$.

Waypoints were considered achieved if sUAVs were within 3.0 meters of them. The collision avoidance minimum and desired separation distances were set to 3.0 and 6.0 meters, respectively.

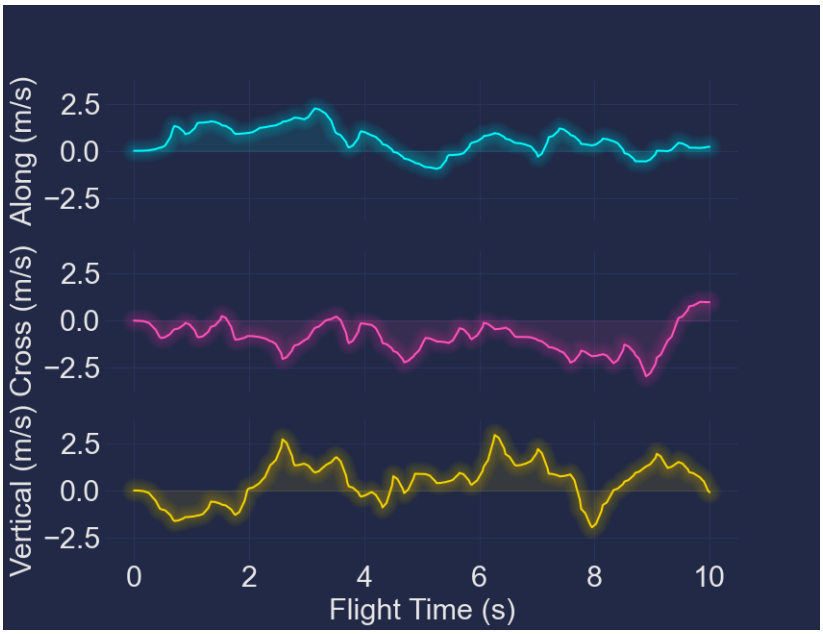
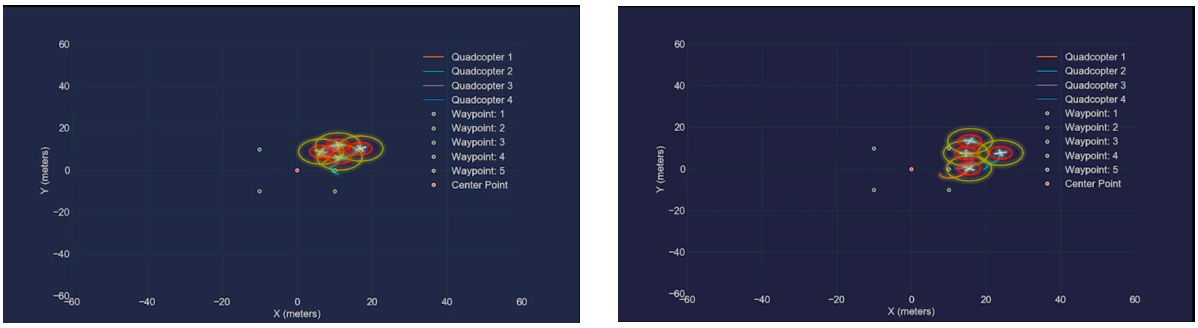


Figure 3.6 The magnitude of Dryden model based wind speeds in the along-wind, cross-wind, and vertical wind directions in m/s.

100 runs of four quadrotor sUAVs flocking along a 5-waypoint rectangular area while maintaining the flocking rules of cohesion, alignment, and collision avoidance were simulated. Figure 3.7a depicts the multi-sUAV flock at the third waypoint when there were no wind disturbances. Figure. 3.7b depicts the multi-sUAV flock at that same point when wind disturbances were applied. From Figure. 3.7b, it is observed that the quadrotors had overshot the waypoint due to the wind disturbance.



(a) No wind disturbances added **(b)** Wind disturbances added

Figure 3.7 Four quadrotor sUAV flock around a 5-point rectangular path in simulation with and without the application of Dryden wind gust model.

However, as observed in the accompanying video, successful flocking behavior was maintained throughout the simulation. The multi-sUAV formation completed the mission with and without wind disturbances while avoiding collision and reaching all 5 waypoints. A simulation based comparison of the OF-GM-SARSA flocking planner presented here with the genetic algorithm-based behavioral flocking planner was conducted in our previous work [330]. The comparison suggested high generalizability of the trained model and proved the model's capability of keeping the number of collisions to a minimum.

3.7 Testbed and Methodology

This section describes the experimental hardware and software architecture and test procedures. The HWIL tests used table-top setups of the quadrotors that ran the flocking planner for 5-waypoint missions. The waypoints were spread out in a square pattern over a 40 meter by 40 meter area for both the HWIL and the outdoor flight tests.

3.7.1 Flocking Quadrotor Architecture

Each quadrotor sUAV was equipped with a Raspberry Pi 4 flight computer, an RFM69 radio module, and the DJI N1 flight controller as depicted in Figure 3.8. The associated software architecture is also depicted in Figure 3.9. The DJI M100 N1 flight controller was responsible for position and velocity control using velocity and yaw-rate inputs generated by the flocking algorithm. The N1 flight controller was connected to the Raspberry Pi running the DJI Onboard SDK via a universal asynchronous receiver transmitter (UART) connection. The OF-GM-SARSA software and Q-tables learned in training (Section 3.5) ran on the Raspberry Pi in a Python ROS node for each platform. The local telemetry information was obtained by interfacing with the DJI N1 flight controller. Both telemetry information and velocity control were achieved

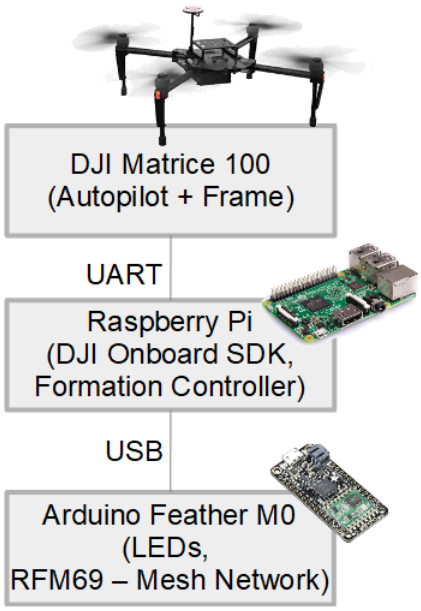


Figure 3.8 Quadrotor hardware setup for the flocking system. The DJI M100 quadrotor platform featured a Raspberry Pi 4 computer for controls and motion planning and a 915MHz mesh radio for networking with other quadrotors. The mesh network was used to communicate position, velocity, acceleration, and other relevant controls data.

using the DJI SDK ROS interface. An Arduino micro-controller was connected to the Raspberry Pi over USB. The Arduino provided a bridge for the RFM69 915 MHz wireless radio transceiver.

A ground station computer hosted a web-based user interface that provided status monitoring information and high-level experimental control for each experiment.

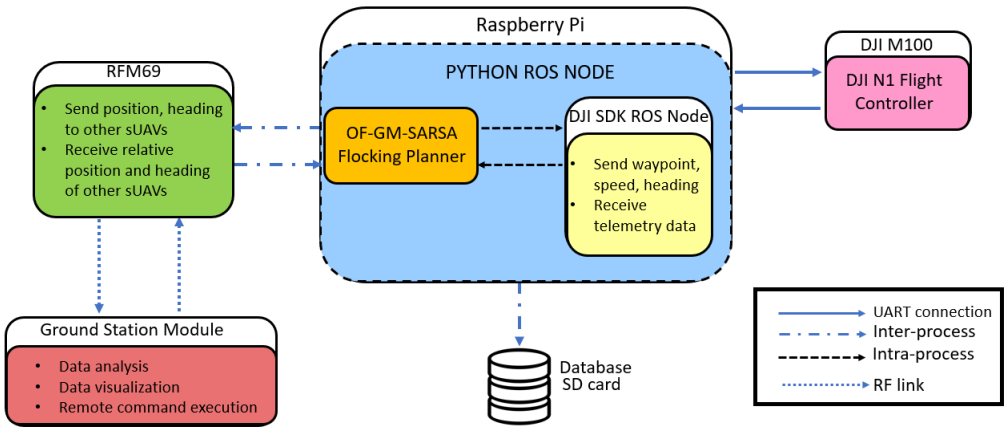


Figure 3.9 Software architecture for the flocking system.

The ground station also used an RFM69 915 MHz wireless radio transceiver to acquire telemetry information and perform clock synchronization for all quadrotors. A time division multiple access (TDMA) data link protocol was implemented to achieve low latency wireless communication between neighboring sUAVs and the ground station. Each data payload included neighbor position, yaw heading, and network connectivity information about the transmitting quadrotor. Before starting a mission, communication slot assignments were determined, with the ground station always configured to use the first communication slot.

All HWIL tests were executed using the DJI Assistant 2 flight simulation software, and the same compute and communication payloads. The software provided a real-time emulation of the DJI M100 rigid body dynamics to simulate the telemetry outputs and velocity and yaw rate inputs into the N1 flight controller. The DJI SDK ROS topics were published to the Raspberry Pi shown in Figure 3.8 to test the OF-GM-SARSA flight planner. During HWIL testing, the three quadrotors and ground stations were placed in line with approximately 0.5 meters between each quadrotor and approximately 1.5 meter separation to the ground station to minimize packet loss.

3.7.2 Wind Measurement

An FT205 ultrasonic anemometer sensor with an embedded computer was used to perform wind measurements during the flight tests, as shown in the Figure. 3.10. The sensor was mounted on a DJI M100 quadrotor using 20 inches long 3D printed pole. The FT205 was sampled at a frequency of 2 Hz. The wind measurements were performed with the quadrotor hovering at the height of 12.5 meters above ground level. The quadrotor was flown approximately 50 meters away from where the flocking quadrotors were operating to maintain safety.

Figure 3.10 FT-205 wind sensor mounted on the DJI Matrice 100 sUAV to collect wind data during flight test.

3.7.3 OF-GM-SARSA Configuration

For all tests, the OF-GM-SARSA planner was configured with desired separation distance of $k_{sep} = 6$ meters and a minimum separation distance of $k_{col} = 3$ meters. The reach distance for waypoints was set to 6 meters. The cohesion radius for the cohesion objective was set to 30 meters. Several weight combinations were used for the OF-GM-SARSA objectives, with each combination summing to 1. All quadrotors were required to be within the reach distance to the current waypoint before moving to the new waypoint. The update rate for the OF-GM-SARSA planner was set to 4 Hz, which was selected to match the update rate of the DJI N1 flight telemetry sampling.

3.7.4 Flight Procedure

Each HWIL and outdoor flight test consisted of the following three stages:

Flight Stage 1: Take-off and position In the first stage, the DJI M100 quadrotors would take off one at a time and move along a pre-specified path to a nearby location denoted by the center point. The quadrotor would then move into a relative formation as determined by a position offset from the center point, after which the next quadrotor would be cued to take off. Example initial formations included a

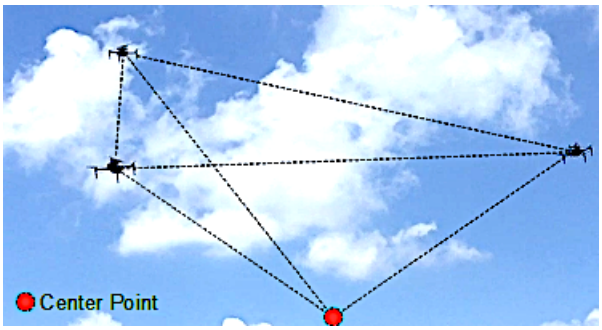


Figure 3.11 Snapshot of a 3 quadrotor triangular formation with 3.5 meter height offsets. The red circle is the center point around which the formation is aligned.

straight line or a triangle as pictured in Figure 3.11. Field tests were performed with a minimum height offset of 3.5 meters between quadrotors for safety reasons.

Flight Stage 2: Target Seek Once the quadrotors reached their initial deterministic formation, the OF-GM-SARSA planner would start the flocking process over the series of pre-specified waypoints.

Flight Stage 3: Return-To-Home (RTH) Once the OF-GM-SARSA planner achieved all waypoints, the quadrotors remained in place and executed an RTH procedure. This stage involved each quadrotor returning to its initial formation location and landing at its original takeoff location. The order the quadrotors returned to their takeoff points was determined by their distance to the center point. The quadrotors were assigned a wait time before starting their RTH to allow enough separation between them.

3.8 Experimental Results

This section presents the experimental results and discussion on the performance achieved by the OF-GM-SARSA planner for 3-sUAVs and 4-sUAVs scenarios. The OF-GM-SARSA planner was evaluated under HWIL and field experiments. All metrics reported did not include telemetry measurements from the takeoff and RTH stages. The experiments were performed five times each to ensure repeatability.

3.8.1 Key Evaluation Metrics

The experimental evaluation of the OF-GM-SARSA motion planner, including the HWIL simulation and outdoor flight tests, provides actionable insights into the practical application of the proposed method. The following metrics were used to measure the system performance in HWIL and field test experiments:

1. Inter-sUAV distance d^{ij}: the Euclidian distance between two sUAVs at any given time.
2. k_{col} violation: an instance where the minimum d^{ij} was below the specified k_{col}.
3. Takeoff + Target Seek + Return-To-Home (RTH) time $\tau_{mission}$: The total time to complete the mission from takeoff to landing.
4. Velocity alignment deviation: the difference between the velocity heading of each sUAV and the average headings of the flock (closer to zero is better).
5. Cohesion distance deviation: the difference between the position of each sUAV and the average position of the flock (lower is better).
6. Total radio packet loss ζ_{loss}^{total}: the fraction of total transmitted radio packets that were received across all quadrotors during the flight test.
7. Average pairwise packet loss ζ_{loss}^{avg}: the total packet loss per quadrotor pair averaged across a total number of quadrotor pairs during the flight test.

3.8.2 Inter-sUAV Distances

3-quadrotor test: The summary statistics associated with 3 quadrotors for d^{ij} are shown in Table 3.1. The top left and top right graphs in Figure 3.12 depict the inter-sUAV distances for 3 quadrotors throughout the entire duration of the HWIL experiment and field test respectively. The average wind speed during this the HWIL test was ~6.54 m/s with a standard deviation of ~0.72 m/s. The average wind speed during the field test was relatively higher at ~7.23 m/s with a standard deviation ~2.49 m/s.

4-quadrotor test: The summary statistics associated with 4 quadrotor for d^{ij} are shown in Table 3.2. The bottom left and bottom right graphs in Figure 3.12 depict

Table 3.1 Inter-sUAV Distance d^{ij}, for 3 Quadrotors Flocking Square in Meters Across 5 Tests. k_{col} Violations for the Test Run depicted in the Top Rows of Figure 3.12.

HWIL			
sUAV Pair	Mean d^{ij}	Min d^{ij}	k_{col} Violations
(1,2)	9.27m	3.83m	0
(1,3)	8.99m	1.03m	2
(2,3)	8.93m	2.7m	1
Field			
sUAV Pair	Mean d^{ij}	Min d^{ij}	k_{col} Violations
(1,2)	8.12m	3.39m	0
(1,3)	17.29m	2.06m	2
(2,3)	18.52m	7.46m	0

the inter-sUAV distances for one set of quadrotor pairs in a 4 quadrotor scenario throughout the entire duration of the HWIL experiment and field test respectively. The average wind speed during this the HWIL test was ~6.54 m/s with a standard deviation of ~0.72 m/s. The average wind speed during the field test was recorded to be ~6.79 m/s with a standard deviation of ~1.77 m/s.

As is observed from the tables and the graphs, the mean d^{ij} distances for both these tests were well above the desired separation distance of 6 meters. However, during the HWIL and field test experiments, there were rare instances where the

Table 3.2 Inter-sUAV Distance d^{ij}, for 4 Quadrotors Flocking Square in Meters across 5 Tests. k_{col} Violations for Test Run depicted in the Bottom rows of Figure 3.12.

HWIL			
sUAV Pair	Mean	Min	k_{col} Violations
(1,2)	8.49m	1.94m	2
(1,3)	9.88m	3.52m	0
(1,4)	7.85m	1.67m	5
(2,3)	12.09m	3.50m	0
(2,4)	10.09m	1.00m	6
(3,4)	11.61m	0.60m	1
Field			
sUAV Pair	Mean	Min	k_{col} Violations
(1,2)	8.55m	1.58m	1
(1,3)	8.87m	1.94m	1
(1,4)	7.71m	1.94m	2
(2,3)	10.24m	3.54m	0
(2,4)	10.67m	3.50m	0
(3,4)	11.88m	2.77m	1

minimum inter-sUAV distance was below the specified k_{col} of 3 meters. For example, the pair (1,3) witnessed a minimum inter-sUAV distance of 1.03 meters in HWIL and 2.06 meters in the field test for the 3-quadrotor scenario. Similarly, several pairs in the 4-quadrotor scenario came closer to the k_{col} distance to each other. In all these instances, the quadrotors quickly corrected (in less than 2.5 seconds) these violations.

The inter-sUAV distance performance recorded in our outdoor flight experiments was benchmarked against the inter-sUAV distance performance obtained using the model-based approach incorporating an evolutionary optimization framework reported in [360]. The following observations are made:

1. Using the approach of [360], the average inter-sUAV distances observed varied between 12 meters and 30 meters. From Table 3.4, the average inter-sUAV distances varied from 8 meters and 18.52 meters for our experiments.
2. Using the approach of [360], the minimum inter-agent distance remained between 5 meters and 15 meters. From Table 3.4, the minimum inter-agent distance for or experiments did not exceed 8 meters.

3.8.3 Total Mission Time

For the three quadrotor test, the mean $\tau_{mission}$ was 487.86 seconds for the HWIL simulation. For the field test, this duration decreased to 477.22 seconds.

For the four quadrotor test, the mean $\tau_{mission}$ was 692.44 seconds for the HWIL simulation. For the field test, this duration decreased to 578.35 seconds.

The most significant factor associated with $\tau_{mission}$ variations was the presence of unpredictable wind conditions in the field test. During windier conditions, it was found that the total execution time varied significantly as the quadrotors would be aided more by the wind gusts in some directions compared to other directions as they tried to maintain collision avoidance, velocity alignment, and cohesion. These wind effects can be readily observed in the flight test video.

Figure 3.12 Top Row: Inter-sUAV distances for 3 quadrotors (Top Left: HWIL and Top Right: Field Test) Bottom Row: Inter-sUAV distances for 3 pairs in the 4 quadrotors test (Bottom Left: HWIL and Bottom Right: Field Test). Both results demonstrate that the flocking controller maintained the required separation k_{sep} and collision avoidance k_{col} distances for most of the test duration. Any minor breaches of the k_{col} were corrected quickly and safely.

Table 3.3 Observed Communication Packet Loss for 3 and 4 Quadrotors

# of sUAV	ζ_{loss}^{total}	ζ_{loss}^{avg}
HWIL		
3	27.11%	9.5%
4	42.2%	6.02%
Field		
3	37.8%	16.21%
4	63.12%	18.02%

3.8.4 Communication Packet Loss

Table 3.3 depicts the ζ_{loss}^{total} and ζ_{loss}^{avg} for the 3-sUAV and 4-sUAV tests. It is observed that both ζ_{loss}^{total} and ζ_{loss}^{avg} are lower for HWIL when compared to outdoor field tests.

Table 3.4 Velocity Alignment Deviation and Cohesion Distance for 3 Quadrotor Averaged over 5 Tests

(a) Velocity Alignment Deviation (Degrees)

HWIL		
sUAV	Mean	Median
1	-7.96°	-4.15°
2	4.72°	-0.40°
3	3.24°	4.13°
Field		
sUAV	Mean	Median
1	-10.54°	-17.01°
2	-4.73°	-0.86°
3	27.48°	18.09°

(b) Cohesion Distance in Meters

HWIL			
sUAV	Mean	Median	St. Dev
1	5.25m	5.42m	1.15m
2	4.99m	5.44m	1.52m
3	3.97m	3.58m	1.48m
Field			
sUAV	Mean	Median	St. Dev
1	4.21m	4.08m	1.59m
2	6.47m	5.52m	3.56m
3	3.97m	6.17m	4.65m

Table 3.5 Velocity Alignment Deviation and Cohesion Distance for 4 Quadrotors Averaged over 5 tests meters

(a) Velocity Alignment Deviation (Degrees)

HWIL		
sUAV	Mean	Median
1	-3.88°	4.06°
2	-2.39°	-5.09°
3	-1.76°	3.57°
4	-11.68°	-5.30°
Field		
sUAV	Mean	Median
1	16.93°	19.69°
2	-15.7°	-20.00°
3	36.27°	41.32°
4	-37.49°	-65.52°

(b) Cohesion Distance in Meters

HWIL			
sUAV	Mean	Median	St. Dev
1	4.26m	4.2m	2.01m
2	6.78m	6.72m	1.94m
3	6.87m	7.05m	1.92m
4	4.33m	3.65m	2.27m
Field			
sUAV	Mean	Median	St. Dev
1	4.43m	4.34m	1.79m
2	5.64m	5.40m	1.94m
3	7.09m	6.86m	2.95m
4	5.47m	5.57m	2.73m

For outdoor field tests, when the pairwise line of sight is not readily established among quadrotors, ζ_{loss}^{total} increases as the number of quadrotors in the mission increase. However, it is also observed that ζ_{loss}^{avg} remains low i.e. majority of the communication happens through multi-hop indirect propagation. In case of 4 quadrotors operating outdoors, ζ_{loss}^{total} is particularly severe at 63% while ζ_{loss}^{avg} remains around 18%.

3.8.5 Velocity Alignment and Cohesion Deviations

The summary statistics for sUAV velocity alignment and cohesion deviations for 3-quadrotor formation flight are shown in Table 3.4. Similarly, the summary statistics

for sUAV velocity alignment and cohesion deviations for 4-quadrotor flight are shown in Table 3.5. From the tables, it is observed that for the 3-quadrotor scenario, the flock achieves stronger cohesion and velocity alignment performance as compared to the 4-quadrotor scenario. It is also observed that the velocity alignment deviation in HWIL tests is significantly lower compared to that in the field tests. Several factors need to be considered: 1) The average wind speed during field tests was higher than during the HWIL experiments. 2) The radio packet losses (ζ_{loss}^{total} and ζ_{loss}^{avg}) were significantly higher in field tests than the HWIL tests. A higher packet loss results in slower/variable telemetry refresh rates, affecting flight control quality. 3) As the number of sUAVs in the flock increases, each sUAV has to accommodate the "wind-disturbed" positions and orientations of an increasing number of neighbors in the flock.

The velocity alignment performance recorded in our outdoor flight experiments was benchmarked against the velocity alignment performance obtained using the deep deterministic policy gradient (DDPG) approach presented in [369]. The mean velocity alignment observed in our outdoor flight tests with three quadrotors varied between -7.96° and 4.72° , indicating the agents maintained their heading relative to each other with minimum deviations. Based on the DDPG approach of [369], for the three-quadrotor test, the mean velocity alignment varied between -108° and 63°.

3.9 Discussion and Future Research

Several future research directions for experimental implementation of RL based motion planning algorithms emerge based on this study.

Energy-Aware Planning: The energy requirements of sUAVs directly affect the practicality of any motion planning algorithm [315, 340]. This requirement is especially true when quadrotors operate in windy outdoor environments where they need to compensate for gusts. An exemplar study in this context is [243], wherein

the authors developed an RL approach that combined the effects of the power consumption and the object detection modules to develop a policy for object detection in large areas with limited battery life. The quadrotors used in this study (DJI M100) were evaluated in hover flight tests to have a flight time of approximately 1200 seconds, which proved sufficient to test the OF-GM-SARSA approach in flight-test experiments. However, the inclusion of sUAV power constraints as an objective in the OF-GM-SARSA paradigm using an approach similar to [243] will be explored in future efforts.

Scalabilty and Collision Avoidance Guarantees: Future efforts will explore guarantees of collision avoidance with outdoor experimental tests involving groups of up to 12-15 quadrotors using Control Barrier Functions (CBFs) [22]. CBFs are being increasingly used to verify and enforce safety properties in the context of safety-critical controllers. They have the potential to provide a computationally tractable approach to combining learning with safety guarantees.

Actor-Critic methods: Despite a relatively slow convergence in the final training stage, the trained OF-GM-SARSA behavioral controller was more than capable of generating flocking behaviors for multiple sUAVs flying in formation outdoors in the presence of wind disturbances. Among existing reinforcement learning techniques, SARSA-based approaches have been shown to have poor convergence performance for real-world applications because they are oriented towards finding the deterministic policy, whereas the optimal policy is stochastic [343]. Alternatively, reinforcement learning approaches based on the actor-critic paradigm have been proven to have good convergence properties [167]. Actor-critic methods have been shown to be scalable to multiple robots with more than 3-degrees of freedom, and they can also be used model-free [264]. In future work, such actor-critic approaches will be explored. It is well known that despite favorable convergence properties, actor-critic

methods are difficult to implement, and their performance is highly dependent on the hardware/software implementation [93]. Towards this end, we plan to use Flightmare, a quadrotor simulator that provides OpenAI gym-style wrappers for several RL algorithms, including actor-critic methods such as Proximal Policy Optimization (PPO) algorithm [310, 329].

3.10 Conclusion

This chapter focused on experimental evaluation of the OF-GM-SARSA planner to address the flocking problem in small rotor-based multi-UAVs. The chapter presented a description of the background of the algorithm, along with the training procedure. The flocking controller was experimentally evaluated in HWIL and field tests for 3-quadrotor and 4-quadrotors missions. The controller's performance was also evaluated under windy conditions in HWIL simulations using the Dryden wind turbulence model. The controller behavior observed in windy conditions in HWIL and outdoor tests was similar to the simulations, suggesting that the technique presented here generalizes the behaviors trained in simulation to real-time interactions.

CHAPTER 4

IMPLEMENTATION OF A MIXED INTEGER NONLINEAR OPTIMIZATION-BASED DECENTRALIZED MOTION PLANNER FOR MULTIPLE QUADROTORS IN THE PRESENCE OF WIND GUSTS

4.1 INTRODUCTION

Paper: *P. Abichandani, D. Lobo, M. Muralidharan, N. Runk, W. McIntyre, D.J. Bucci, and H.Y. Benson. "Implementation of a Mixed Integer Nonlinear Optimization-based Decentralized Motion Planner for Multiple Quadrotors in the Presence of Wind Gusts," submitted to IEEE Robotics and Automation Letters, 2022*

Safe and reliable motion-coordination of multiple rotor-based unmanned aerial vehicles (UAVs) is constrained by their kinematics and dynamics, and the need to avoid collisions while ensuring sufficient wireless communication Quality-of-Service (QoS). Approaches reported in literature to address multi-rotor UAV motion planning include Probabilistic Road Maps (PRMs) [139], Rapidly-exploring Random Trees (RRTs) [266], Velocity obstacles [28], Particle Swarm Optimization (PSO) [40], Artificial Neural Networks (ANNs) [370], Signal Temporal Logic [50, 193], and Mathematical Programming (MP) [5].

Since the seminal studies by *Schouwenaars et al.* and *Richards and How* [291,309], the body of work that presents centralized and decentralized Mathematical Programming (MP) or Mathematical Optimization-based approaches for solving motion planning problems (MVMPs) for ground [160, 286], underwater [7, 373], and aerial robot systems [320, 390] has steadily grown. Mixed-integer nonlinear programs (MINLPs) can be used to capture all MVMP constraints effectively. However, the solutions to MINLP in the context of MVMP problems are computationally intensive (NP-hard), thus rendering real-time implementations and scalability challenging [277].

One approach to address this challenge is the use of receding horizon (RH) or model predictive control (MPC) methods. These methods have proven to be suitable for online and real-time motion planning for multi-robot systems in dynamic and partially unknown environments [5, 173]. A key benefit of using MP-based approaches is that they allow for effectively studying the trade-off between mobility (speed, acceleration, waypoints), wireless communications (bandwidth, buffer size, path loss, delay), and energy consumption [64, 320].

For quadrotors operating outdoors, solutions to MVMPs are exacerbated in the presence of wind disturbances. Despite significant work on multi-quadrotor motion planning, simulation or experimental approaches that explicitly incorporate effective strategies to counter wind gusts are only recently being reported in the literature [66, 259, 350]. Simulation approaches use wind models such as the Dryden and Von Karman models to incorporate realistic wind gusts during motion planning. Experimental approaches to estimate wind speed and direction during motion

Figure 4.1 Snapshots of a 2 quadrotors (top row left), 3 quadrotors (bottom row left), 6 quadrotors (top row right) operating outdoors. Bottom row right image shows the FT-205 wind sensor mounted on the DJI Matrice 100 quadrotor to collect wind data during flight test.

planning use sensors such as ultrasonic anemometers, pitot tubes, accelerometers, gyroscopes, and barometers [296]. In this work, we use the Dryden model to simulate realistic wind gusts in simulations and Hardware-in-the-loop (HWIL) experiments.

Wind gusts continually disturb the quadrotors off their planned positions, thus requiring fast re-computation of the MP solutions to mitigate any increased risk of collisions [207]. Such repeated computations limit the effectiveness of MP frameworks in windy situations. As such, there is a need for a novel approach to reduce the repeated computation of MPs while ensuring that mission safety is not compromised. In this chapter, we present a novel MINLP solution approach that combines heuristic and optimal methods, namely simulated annealing and interior-point methods, respectively, to handle discrete variables and nonlinearities feasibly in real-time. Figure 4.1 depicts outdoor flight tests conducted with 2, 3, and 6 quadrotors to validate our approach.

Control Barrier Functions (CBFs) are being increasingly used to provide guarantees of collision avoidance for multi-vehicle systems [23]. CBFs have been applied to single UAV problems to guarantee collision avoidance while exploring a cave environment and seamlessly integrate with an existing motion control algorithm [111]. CBFs have also been used for collision-free multi-quadrotor systems operating indoors to demonstrate formation control and safe human teleoperation [192, 382]. Recently, learning-based approaches have been used to synthesize CBFs for multiple aircraft [294]. This study extends these use cases by applying CBFs for the safe transit of quadrotors subjected to potentially dangerous wind gusts.

4.1.1 Main Contributions

This chapter presents the first implementation of an RH-MINLP framework that leverages CBFs for multiple, networked quadrotors operating outdoors in windy environments to the best of our knowledge.

1. A RH-MINLP and CBF-based framework to implement collision-free MVMP for multiple, networked quadrotors operating outdoors in the presence of wind gusts.
2. A fast MINLP solution approach that combines Simulated Annealing and Interior point methods for generating feasible real-time solutions.
3. Integration of the Dryden wind gust model with RH-MINLP and CBFs to perform realistic simulations and HWIL testing with DJI Matrice M100 quadrotors.

The framework is validated in simulations featuring up to 50 quadrotors and Hardware-in-the-loop (HWIL) experiments, followed by outdoor field tests featuring up to 6 DJI M100 quadrotors. Extensive performance results of the proposed framework are documented in this chapter.

The rest of the chapter is organized as follows: Section 4.2 describes the problem formulation. Section 4.3 presents the RH-MINLP algorithm followed by the optimization solution technique in Section 4.4. Next we present the experimental setup and results in Section 4.5, and a brief summary of the chapter in Section 4.6.

4.2 Preliminaries

4.2.1 Quadrotor Motion and Paths

Consider a group of n wirelessly networked quadrotor vehicles operating in right-handed NED, body, and body-fixed frames of reference. Each quadrotor $i = 1, \ldots, n$ has a start (origin) point o^i and a end (goal) point e^i. O is the set of all start (origin) points. $o^i \in O, i = 1, \ldots, n$. E is the set of all end points. $e^i \in E, i = 1, \ldots, n$. The Euclidean distance between two quadrotors i and j is denoted by d^{ij}, $i = 1, \ldots, n, j \neq i$.

All quadrotors are required to maintain a minimum safe distance d_{safe} to avoid collisions with each other. For each quadrotor i, the total arc length of its path is denoted by U^i. The linear and angular speeds of quadrotor i along its fixed path are denoted by s^i and ω^i, each expressed in the body-fixed frame of the quadrotor.

$\mathbf{p}^i(t) = (x^i(t), y^i(t), z^i(t))$ is the location of quadrotor i on its path at time t, and is calculated using the speed $s^i(t)$. The path for quadrotor i is represented by a three-dimensional (3D), piecewise 7^{th} order spline curve $\mathbb{SP}^i(u)$, where the parameter u is the normalized arc length along the curve. The curves are obtained by combining three one-dimensional piecewise 7^{th} order splines. The piecewise 7^{th} order spline curves feature continuous first derivatives (slope), second derivatives (curvature), third derivatives (jerk), and fourth derivatives (snap) along the curve. Fig. 4.2 depicts the spline curve paths for a 6-quadrotor scenario.

The differential flatness of quadrotor dynamics, use of 3D piecewise 7^{th} order spline curves, and the constraints on the speed, acceleration, snap, and yaw rate results in kino-dynamically feasible speed profile for quadrotors [221, 293]. A detailed discussion on spline curve design and analysis can be found in [183].

Let $\kappa^i(u)$ be the curvature along the spline curve i. The angular speed $\omega^i(u)$ can be computed from the linear speed and spline curvature as follows:

$$\omega^i(u) = s^i(u)\kappa^i(u). \tag{4.1}$$

4.2.2 Receding Horizon (RH)

T_{hor} denotes the length of the receding horizon and is determined by the capabilities of the embedded flight planning and control system. At any discrete time step t, each quadrotor must calculate its plan for $t+1, \ldots, t+T_{\text{hor}}$ and communicate this plan with other quadrotors in the network. This plan is denoted by $\mathcal{P}^i(t) = (\mathbf{p}^i(t+1), \ldots, \mathbf{p}^i(t+T_{\text{hor}}))$. Only the plan for time $t+1$ is implemented, and the planning process is restarted at time step $t+1$. At time step T_{mission}, the mission is completed when the last-arriving quadrotor reaches its end point. If a quadrotor reaches its goal point before T_{mission}, it hovers there until the mission is over.

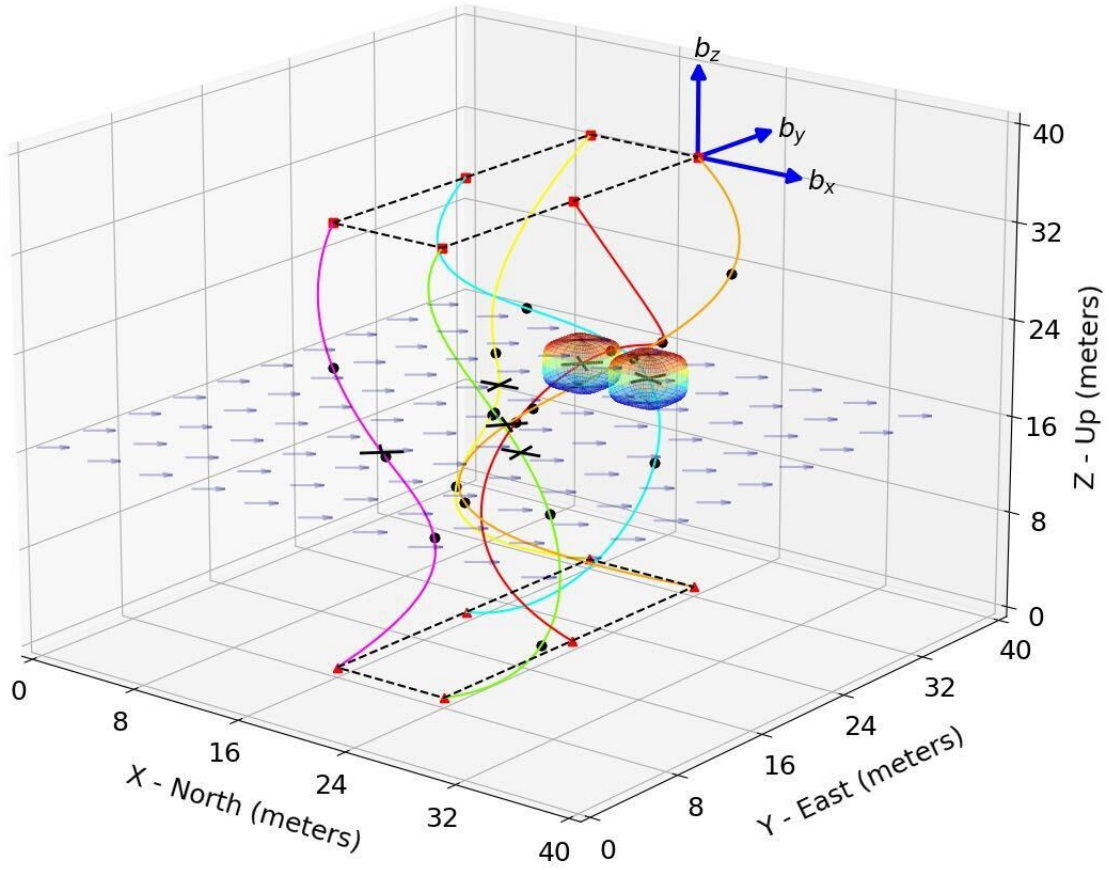


Figure 4.2 A multi-quadrotor mission with six quadrotors moving along their spline paths (represented by solid, colored lines) in the presence of wind. Control Barrier Functions (CBFs) and associated safety certificates visualized as super-ellipsoids provided robust collision avoidance in the face of wind disturbances during transit. The vector field (blue arrows) indicates wind gusts generated using the Dryden wind model. Red triangular and square markers indicate the start and endpoints of the spline paths arranged in a geometric formation. Black round markers indicate spline-path waypoints. The North, East, Down (NED) frame and the Body frame (b_x, b_y, and b_z) are shown. The quadrotors operate in a 40 m x 40 m x 40 m airspace.

4.2.3 Communication Modeling and Ordering

Each quadrotor is equipped with a mobile ad-hoc network (MANET) radio to exchange RH-MINLP solutions with other quadrotors. MANET communication is accomplished via direct one-hop connection or multi-hop relaying, thus resulting in a fully connected network. A round-robin scheduling technique for sequential wireless communication is used for fair distribution of MANET bandwidth and fault tolerance.

The sequence of the quadrotors for the round-robin scheduling is determined randomly at the beginning of the mission. The first quadrotor in the round-robin is also the first to take off.

4.2.4 Dryden Wind Model and Spline Regeneration

The continuous Dryden turbulence model is used to simulate wind gusts and study their effects on the motion planning solutions [122]. The open-source implementation of the Dryden wind turbulence model can be found in [79]. When the quadrotors are displaced off their pre-defined spline paths due to wind gusts, a rapid spline regeneration process ensures that each quadrotor has an updated spline path starting from the displaced location to the endpoint. Spline regeneration is effectively visualized in the accompanying video submission.

4.2.5 Control Barrier Functions and Safety Barrier Certificates

Each quadrotor is encapsulated in a super-ellipsoidal, Exponential Control Barrier Function (ECBF) depicted in Fig. 4.2. For a pair of quadrotors (i, j), the pairwise safe set $\mathcal{B}^{ij}$ and the pairwise super ellipsoid $h^{ij}(q^i, q^j)$ are defined as:

$$\mathcal{B}^{ij} = \{(q^i, q^j) | h^{ij}(q^i, q^j) \geq 0\}, \tag{4.2}$$

$$h^{ij}(q^i, q^j) = (x^i - x^j)^4 + (y^i - y^j)^4 + (\frac{z^i - z^j}{c})^4 - d_{safe}^4 \tag{4.3}$$

Where q^i represents the full state of the quadrotor i, d_{safe} is the safety distance, c is the scaling factor along the Z-axis caused by airflow disturbance. The two quadrotors are considered safe when the two super-ellipsoids do not intersect. Readers are referred to [372] and its references for information about the mathematical

construction, verification, and incorporation of ECBFs in motion planners through the use of Quadratic Programming (QP).

4.3 Optimization Model

The optimization (MINLP) problem $\mathcal{O}(t)$ solved by each quadrotor is expressed in (4.4) - (4.14). The optimization is performed for speed s^i of the quadrotor along its specified path and auxiliary variables are used for position (x^i, y^i, z^i), arc length (u^i), acceleration (a^i), and communication connectivity (C^{ij}, $j \neq i$). Each quadrotor i uses the most current plans of all other quadrotors while solving $\mathcal{O}^i(t)$.

Problem Size: $\mathcal{O}(t)$ has $(n+5)T_{\text{hor}}$ decision variables, $(2n+5)T_{\text{hor}}$ constraints, and variables bounds. The problem size grows linearly with the number of quadrotors n.

4.3.1 Objective Function

The objective function (4.4) forces the quadrotors to minimize the total distance (arc length) between their current location and the goal position over the entire receding horizon. It prioritizes the quadrotors' movement to their goal positions.

$$\underset{s,x,y,z,u,a,C}{\text{minimize}} \quad \sum_{k=t+1}^{t+T_{hor}} (U^i - u^i(k)) \tag{4.4}$$

subject to $\quad \forall k \in \{t+1, \ldots, T_{\text{hor}}\}$

$$(x^i(k), y^i(k), z^i(k)) = \mathbb{SP}^i(u^i(k)) \tag{4.5}$$

$$u^i(k) = u^i(k-1) + s^i(k)\Delta t \tag{4.6}$$

$$u^i(t) \leq u^i(k) \leq U^i \tag{4.7}$$

$$s^i(k) = s^i(k-1) + a^i(k)\Delta t \tag{4.8}$$

$$s_{\min} \leq s^i(k) \leq s_{\max} \tag{4.9}$$

$$a_{\min} \leq a^i(k) \leq a_{\max} \tag{4.10}$$

$$\forall j \in \{1, \ldots, n\}, j \neq i$$

$$\left(d^{ij}(k)\right)^2 \geq d^2_{\text{safe}} \tag{4.11}$$

$$\left(d^{ij}(k)\right)^2 \leq M(1 - C^{ij}(k)) + \eta_d^2 \tag{4.12}$$

$$\sum_{j:j\neq i} C^{ij}(k) \geq n_{\text{conn}} \tag{4.13}$$

$$C^{ij}(k) \in \{0, 1\} \tag{4.14}$$

4.3.2 Path (Kinematic) Constraints

Constraints (4.5)-(4.7) define the quadrotor's location along its path. Location can be defined in two ways: arc length traversed along the quadrotor's path (u^i) and the coordinates of its location (x^i,y^i,z^i). The former is used to define kinematic and dynamic constraints, while the latter is used to calculate Euclidean distances between quadrotors for collision avoidance and communication connectivity. Constraint (4.5) defines the 7^{th} order spline curves $\mathbb{SP}^i(u^i(k))$ that link these two sets of variables. Given its location at the beginning of the planning period, constraint (4.6) increments the arc length at each time step based on the speed of the quadrotor. Δt is the DJI

Matrice M100's flight controller update rate obtained directly during actual tests. Constraint (4.7) ensures that the quadrotor will remain between its location at the beginning of the planning period and its goal position at all times.

4.3.3 Speed and Acceleration (Dynamic) Constraint

Given the quadrotor's linear speed s^i at the beginning of the planning period, constraint (4.8) increments the speed at each time step based on its acceleration a^i. Constraints (4.9)-(4.10) are dynamic constraints that ensure that the linear speed (and hence, the angular speed) and acceleration for each quadrotor at each time step are sufficiently bounded. The bounds are determined by the capabilities of the DJI M100 quadrotor and the curvature of the spline paths to ensure that the quadrotor motors do not saturate. Since the 7^{th} order spline curves are generated while minimizing snap (4^{th} derivative), the paths allow for smooth trajectory following. The angular speed required by the quadrotors corresponding to the optimal linear speed is always achievable and is determined using a relationship (4.1).

4.3.4 Collision Avoidance Constraint

The non-convex constraint (4.11) defines the required minimum distance d_{safe} between each pair of quadrotors at all times to avoid collisions.

4.3.5 Communication Connectivity Constraint

The binary variable $C^{ij}(t)$ indicates whether two quadrotors are in one-hop communication range η_d of each other as shown in:

$$C^{ij}(t) = \begin{cases} 1, & \text{if } d^{ij} \leq \eta_d, \\ 0, & \text{otherwise.} \end{cases} \tag{4.15}$$

Equation (4.13) represents the communication connectivity constraint, which ensures that each quadrotor is in a one-hop communication range of n_{conn} other quadrotors. In the proposed model (4.15) is replaced with equation (4.12). This Big-M constraint (along with (4.14)) forces $C^{ij}(t)$ to be 0 when $d^{ij} > \eta_d$, but leaves it free to be 0 or 1 when quadrotors i and j are within communication distance. Constraint (4.13) then ensures that at least n_{conn} of the free $C^{ij}(t)$ values must be set to 1. Equation (4.13) can also be represented as an equality constraint without the need to change the solution.

4.4 OPTIMIZATION SOLUTION TECHNIQUE

The numerical optimization solution technique for solving the set of equations (4.4) - (4.14) is divided into three levels – the outer level, the middle level, and the lower level. At each time step t, the outer level, iterates through the quadrotors in a pre-determined order and manages the decision updates. In the middle level, the algorithm handles the discrete variables and communication constraints of each quadrotor's problem $O^i(t)$ via Simulated Annealing. The lower level solves the underlying continuous nonlinear programming problem using interior-point methods to determine each quadrotor's speed during the time horizon. This three-stage approach ensures that the multi-UAV system performs efficiently and is scalable.

4.4.1 Outer Level: Decentralization

All quadrotors are in the communication range of each other at time 0. As discussed, a round-robin decision-making and communication order is enforced among quadrotors. The ordering can be randomly assigned or can be assigned *a priori*. It is assumed without loss of generality that the ordering is represented as $1, \ldots, n$.

Each quadrotor i sequentially solves the problem $\mathcal{O}^i(t+1)$ at time t by taking into account the following plans:

- Plans $\mathcal{P}^j(t+1)$ for quadrotors $j < i$, as these quadrotors have already calculated their new plans, and

- Plans $\mathcal{P}^j(t)$ for quadrotors $j > i$, as these quadrotors have yet to calculate their new plans.

If $\mathcal{O}^i(t+1)$ is feasible, the new plan becomes $\mathcal{P}^i(t+1)$. If it is infeasible, then the quadrotors continue to use $\mathcal{P}^i(t)$ for the remainder of the horizon. This plan is then communicated to the rest of the quadrotors.

4.4.2 Middle Level: Simulated Annealing for Discrete Variables and Communication Constraints

The outer level requires the solution of $\mathcal{O}^i(t+1)$ for each quadrotor i at each time step t. The communication variables and the connectivity constraints (4.12) contribute significantly to the computational complexity of $\mathcal{O}^i(t+1)$: the communication variables are binary, there are $n*T_{\text{hor}}$ of them in each instance (the remaining variables do not change by fleet size), and there are $n * T_{\text{hor}}$ poorly-scaled communication constraints at each instance. The communication variables and constraints are handled at this second level.

When deciding upon an appropriate method to handle the communication variables and constraints, it should be first noted that a feasible solution to $\mathcal{O}^i(t+1)$ is implementable, even if it is not optimal. From a solution approach perspective, a well-established method such as branch-and-bound (BB) with deep dives can be used to solve MINLPs [181,189], but this may still require the solution of multiple instances of the RH-MINLP (and hence more computational effort/time) before identifying a feasible solution, especially as the problem size grows.

By contrast, Simulated Annealing (SA) provides a probabilistic technique for generating incumbent solutions to approximate the global optimum of a given function, and it can be designed to promote integer feasibility. To implement SA, the

1: **Input**: The sets $\{1, \ldots, N-1\}$ and $\{1, \ldots, T_{hor}\}$
2: **Output**: $\mathcal{C}_{k'}$
3: Randomly select $k' \in \{1, \ldots, T_{hor}\}$.
4: Randomly select $j' \in \mathcal{C}_{k'}$ and $j'' \notin \mathcal{C}_{k'}$.
5: Set $C^{ij'}(k') = 0$, $C^{ij''}(k') = 1$.
6: Update: $\mathcal{C}_{k'} \leftarrow \mathcal{C}_{k'} \setminus j' \bigcup j''$.

Algorithm 2: Generating a neighbor of an incumbent solution for Simulated Annealing

binary values for the communication variables C^{ij} at each iteration were chosen and fixed such that (4.13) was satisfied with equality as given in the following equation:

$$\sum_{j:j\neq i} C^{ij}(k) = n_{\text{conn}}$$

To generate a solution with the given equality, for each time period k, the set $\{1, \ldots, N-1\}$ was sampled without replacement to generate n_{conn} indices. This set of indices is denoted as $\mathcal{C}_k$ and set

$$C^{ij}(k) = \begin{cases} 1, & \text{if } j \in \mathcal{C}_k \\ 0, & \text{otherwise.} \end{cases}$$

The goal of selecting the binary values for the communication variables was to choose binary values that attain or promote the solution's feasibility. The condition $C^{ij} = 0$ does not always imply that quadrotor i and j are not within communication distance. This condition indicates that the quadrotors do not need to be within communication distance. The incumbent solutions which satisfy the n_{conn} constraint with equality indicate that the quadrotors are required to maintain at least n_{conn}, and not exactly n_{conn}. This method for generating a neighbor of the incumbent solution is depicted as Algorithm 2.

Once the communication variables are fixed and constraint (4.13) satisfied, a reduced form of $\mathcal{O}^i(t)$, denoted by $\mathcal{O}^i_{\text{reduced}}(t)$ is created where constraints 4.12, 4.13, and 4.14 are replaced by

$$\left(d^{ij}(k)\right)^2 \leq \eta_d^2 \text{ if } C_{ij}(k) = 1 \tag{4.16}$$

This replacement leads to better scaling than (4.4)-(4.14) due to the lack of big-M constraints.

Reduced Problem Size: The resulting optimization problem is also significantly smaller in size, with $5T_{\text{hor}}$ variables and $(N + n_{conn} + 4)T_{\text{hor}}$ constraints.

4.4.3 Lower Level: Nonlinear Optimization

$\mathcal{O}^i_{\text{reduced}}(t)$ is a non-linear program and can be solved using an interior-point method for nonlinear optimization adapted from [316]. Readers are referred to [316] and its references for further details on this nonlinear optimization technique.

4.5 Experimental Setup and Results

The RH-MINLP framework was implemented in numerical simulations, HWIL, and outdoor flight tests. The optimization model $\mathcal{O}^i$, optimization solver (all three levels), and CBFs were implemented in Python. Custom spline generation functions implemented in Python were used to generate the minimum snap 7^{th} order splines passing through N pre-selected waypoints. The Dryden Model to generate wind gusts was also implemented on the ground station computer in Python.

Numerical simulations were carried out using a desktop computer fitted with an i5-6400 Intel CPU 2.70 GHz Quad-core processor and 8 GB RAM. HWIL and outdoor flight tests were conducted with DJI M100 quadrotors fitted with a Raspberry Pi 4 (Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz and 4 GB RAM)

computer. Each quadrotor was fitted with a transceiver radio module operating in the 915MHz license-free frequency bands, providing half-duplex bi-directional RF links at 300Kbps. A ground station computer (Intel Dual Core i5 @ 3.3GHz Processor and 8 GB RAM) was part of the MANET and hosted a web-based user interface for mission configuration, clock synchronization, and mission upload. The GPS sensor's positional accuracy and sampling rate onboard the DJI M100 are 1 meter and 4 Hz, respectively. The fused telemetry stream data from the DJI M100 were sampled at 50 Hz to obtain state feedback (real-time locations and speeds).

The HWIL tests were executed using the DJI Assistant 2 flight simulation software. The software provided real-time emulation of the telemetry outputs and velocity yaw rate inputs to the DJI N1 flight controller based on the rigid body dynamics of the DJI M100. During HWIL testing, the quadrotors and ground station were placed in line with approximately 0.5 meters between each quadrotor and approximately 1.5-meter separation from the ground station.

The Python implementation interfaced with DJI M100 real-time telemetry data via UART and a Linux SDK provided by the DJI M100 N1 flight computer. The speed obtained by solving the $\mathcal{O}$ served as the set-point for the DJI M100's N1 flight controller, which exerted the appropriate control effort to achieve the set-point. During outdoor tests, a manual override was implemented to stall the quadrotor in place until a human operator took over in case of degraded MANET performance, heavy wind gusts, or critically low battery in the quadrotors. Table. 4.1 lists key motion and communications parameters.

Table 4.1 Motion and Communication Parameters

d_{safe}	3m	$s_{\min}$	0	$s_{\max}$	0.5m/s
a_{min}	-0.25 m/s^2	$a_{\max}$	0.25 m/s^2	n_{conn}	1
η_d	$5m$	P_{tr}	$100mW$	T_{hor}	3

Numerical simulations focused on exploring scalability (up to 50 quadrotors) of the RH-MINLP + CBF framework and comparing the computational efficiency of the SA + Interior Point Method solver with another established method (Branch-and-Bound (BB) + Interior Point Method). HWIL tests were performed with up to 6 DJI M100 to study 1) the effect of wind disturbances on safety violations, 2) the safety accorded by CBFs, and 3) the effect of wind disturbances and CBFs on T_{mission}. Actual flight tests followed the simulations and HWIL tests in outdoor, windy settings to validate the real-time feasibility of the RH-MINLP + CBF approach.

Each flight in numerical simulations, HWIL tests, and outdoor tests was conducted in a 40 m x 40 m x 20 m airspace and consisted of the following three sub-tasks:

- Take-off and Formation: The quadrotors would take off one at a time and move to a preset location to organize themselves in a geometric formation. Example initial formations included a straight line, triangle, and rectangle as shown in Figure 4.1. The initial height of the formation was 12.5 meters above ground level (AGL).

- Waypoint-based Transit: Once in formation, the quadrotors would start their transit while maintaining their flight formation, to the extent possible, and visit a total of 5 waypoints spread out across an area of 40 meters x 40 meters x 20 meters.

- Return-To-Home (RTH): Once the quadrotors visited all the waypoints, they sequentially executed an RTH procedure to land at their take-off locations safely.

Key evaluation metrics are defined as:

1. T_{mission}: denotes the transit time between the start of the multiple quadrotor formation and their return to their start points averaged over multiple runs.
2. n_{CBF}: number of CBF activations averaged over multiple runs and rounded to the nearest integer.
3. T_{comp}: Solver computation time averaged over all quadrotors, further averaged over multiple runs.
4. RR: Motion re-planning rate in Hz, average over multiple runs.

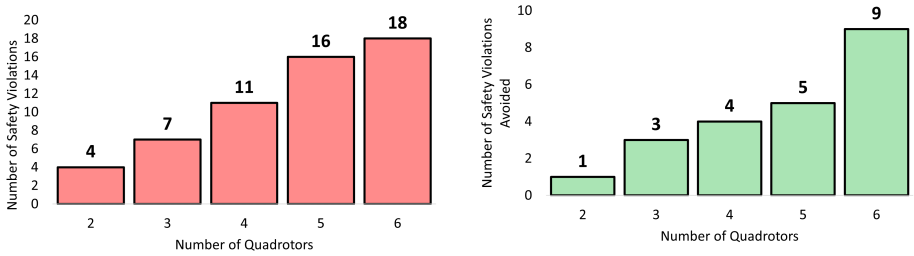


Figure 4.3 (Left) In HWIL tests without CBF, the number of safety violations is shown in the red bars. As CBFs were not used, none of these violations could be avoided. (Right) In HWIL tests with CBF, the number of CBF-activations n_{CBF} are shown in the green bars. The use of CBFs resulted in zero safety violations. Data averaged over 9 runs.

5. T_{saving}: denotes the percentage of T_{comp} time saved when using SA instead of the BB method.

6. $D_{\text{obj-gap}}$: denotes the percentage gap in the objective function values (as defined by (4.4)) between SA and BB.

Each numerical simulation was repeated five (5) times, HWIL experiments were repeated nine (9) times, and outdoor flight tests were conducted five (5) times. Dryden gusts with average 9 m/s speed and standard deviation of 1.23 m/s were used in numerical simulations and HWIL.

4.5.1 HWIL: Flight Safety Accorded by CBFs

For a fixed d_{safe}, Fig. 4.3 (left) depicts the number of times the inter-quadrotor distance (d^{ij}) was less than d_{safe} during windy conditions. These safety violations were reported because the wind gusts disturbed the quadrotors off their planned course despite recalculating $\mathcal{O}$. This observation underscores the need for additional safeguards to prevent in-flight collisions.

Once CBFs were added to the motion-planning process, as depicted in Fig. 4.3 (right), there were **no safety violations during flight** despite windy conditions. The number of potential safety violations that were prevented by adding the CBFs were significantly less than the case when no CBFs were present. This is due to the fact that the quadrotors were moving in a safer fashion from the very beginning of

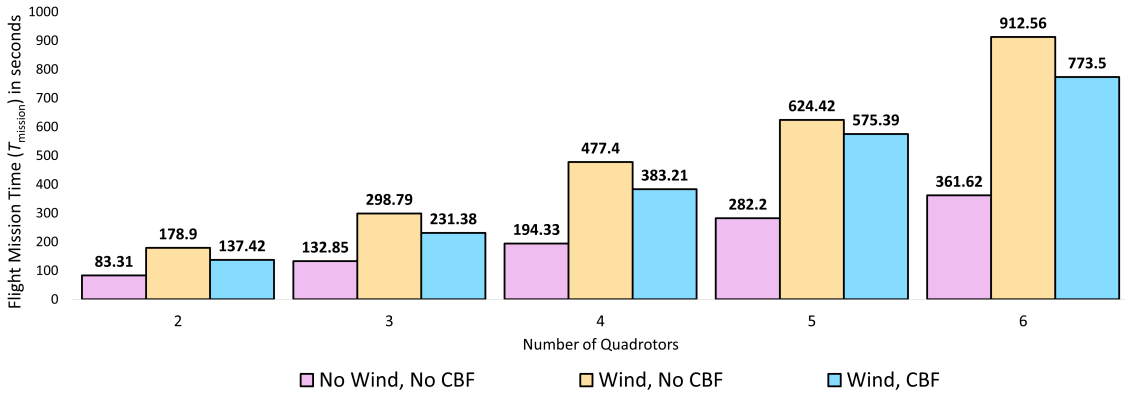


Figure 4.4 For $d_{\text{safe}} = 3$m, the above charts depict T_{mission} in seconds for different HWIL scenarios. The presence of wind increases the computational times and result in severe safety violations. Adding CBFs to the overall motion planning strategy reduces the overall computational times while resulting in **0** safety violations. Data averaged over 9 runs.

their mission due to the activation of the CBFs. There was a relatively lower number of potential safety violations in future time steps. We note that in the absence of Dryden wind gusts, no safety violations were observed while solving $\mathcal{O}$.

4.5.2 HWIL: Effect of Wind Disturbances and CBFs on $T_{mission}$

For a fixed d_{safe}, Fig. 4.4 depicts the change in T_{mission} in the absence and presence of wind disturbances. It is observed that T_{mission} increases significantly in the presence of the Dryden wind gusts because $\mathcal{O}$ is solved more often because the wind gusts cause the quadrotors to move off-course.

Fig. 4.4 also depicts the change in T_{mission} in the presence of wind disturbances while using CBFs (blue-colored data series). It is observed that the T_{mission} increases with CBFs in windy conditions as compared to the no wind case. However, it is also observed that in the presence of CBFs, T_{mission} is lower than the T_{mission} when no CBFs are present. This is due to the fact that the CBF activations afford collision-avoidance from the very beginning of the mission thereby reducing the number of times $\mathcal{O}$ needs to be resolved. A reduced T_{mission} also indicates lower use of quadrotor battery.

Table 4.2 Numerical Simulation Results to Study Scalability of the RH-MINLP Framework (Data Averaged over 5 runs)

n	SA		BB			
	T_{comp}	n_{CBF}	T_{comp}	n_{CBF}	T_{saving}	$D_{\text{obj-gap}}$
10	132.3	14	195.2	16	47.54%	-5.29 %
15	186.8	17	311.9	19	66.97%	-4.66%
20	271.1	19	648.5	18	139.21%	0.2%
25	383.9	22	902.4	21	135.06%	-3.10%
30	562.1	27	1376.2	30	144.83%	-1.98 %
40	819.3	31	2314.7	31	182.52%.	-7.81 %
50	1184.5	38	3920.7	42	231%	-6.52 %

4.5.3 Numerical Simulations for Scalability

Numerical simulations were performed for up to 50 quadrotors operating outdoors to study computational scalability in the presence of Dryden wind gusts. Table 4.2 notes results for each of these scenarios. The SA + interior-point numerical solution method was compared with a BB + interior point method as presented in [181, 189].

As is observed, for all cases, n_{CBF}, and T_{comp} increase with n. The increase is pronounced at higher n due to increased collision possibilities and an increase in the number of variables and constraints for $\mathcal{O}$.

It is observed that SA outperforms BB in terms of T_{comp} as indicated by T_{saving}. These T_{comp} savings can be attributed to the fact that BB fixes a subset of the binary variables (and leaves the rest as continuous, between 0 and 1) at any iteration which in turn, causes the nonlinear solver to face larger problems with BB than with SA. However, when it concludes, BB yields superior objective function values for all tests, which is expected since it focuses on global optimality. As the number of quadrotors is scaled up, BB may not conclude in real-time, either due to the size of the problem or the nonconvexity of the underlying nonlinear problem due to the collision avoidance constraints. On the other hand, SA is a search heuristic that focuses first on attaining feasibility and then on further improving the objective function value, and can be stopped within the real-time requirements.

4.5.4 Outdoor Flight Tests

Outdoor field tests were conducted with up to 6 DJI M100 quadrotors to validate the RH-MINLP + CBF framework. The test was conducted in an open field at a park in Philadelphia. The wind disturbances during the flight tests were measured using an FT205 ultrasonic anemometer mounted on a DJI M100 quadrotor using a 20-inch long 3D printed pole as shown in Fig. 4.1. The FT205 was sampled at a frequency of 2 Hz. The wind measurements were performed with the quadrotor hovering at a height of 12.5 meters above ground level. The quadrotor was flown approximately 50 meters away from the multi-quadrotor system to maintain safety. During the field test with 2 quadrotors, the average wind speed was 12.23 m/s with a standard deviation 3.49 m/s. During the field test with 3 quadrotors, the average wind speed was 10.33 m/s with a standard deviation 1.54 m/s. During the field test with 6 quadrotors, the average wind speed was 13.5 m/s with a standard deviation 2.0 m/s.

Figure 4.1 depicts a snapshot of a 2, 3, and 6-quadrotor tests. Table 4.3 depicts the average T_{mission}, n_{CBF}, T_{comp}, and re-planning rate (RR). T_{comp} varied from 26.8 ms to 74.6 ms. Re-planning rates varied between 12 Hz to 30 Hz.

Readers are referred to the accompanying video for footage of the flight tests. For the 2-quadrotor test, a CBF was activated when the quadrotors completed a concentric-spiral path mission and the two quadrotors were about to breach the safety distance threshold. One of the quadrotors came to a hover allowing the other quadrotor to pass. Similarly, for the 3-quadrotor test, the CBF was activated twice, once each between two different pairs. For the 6-quadrotor test, 4 CBFs were activated

Table 4.3 Outdoor Flight Tests Results Demonstrate the Feasibility of the RH-MINLP + CBF Framework. Data Averaged over 5 Flight Tests per n.

n	$T_{\text{mission}}(s)$	n_{CBF}	T_{comp} (ms)	RR (Hz)
2	141.52	1	26.8	30
3	258.10	2	37.2	24
6	829.47	4	74.6	12

between different pairs when the quadrotors came close to each other due to wind gusts.

4.6 Conclusion

This chapter comprehensively validated a decentralized, RH-MINLP + CBF based motion planner for multiple quadrotors operating in outdoor, windy environments. The validation was performed via numerical simulations, HWIL, and outdoor flight tests. A novel 3-level optimization solver that used Simulated Annealing was developed to expedite the optimization solution process resulting in feasible online motion-planning. The Dryden wind gust model provided realistic wind effects during the simulations and HWIL experiments. The use of CBFs provided significant robustness against safety violations and potential mid-flight collisions that could have resulted due to the wind disturbances, in turn reducing mission times. The simulation and experiment results demonstrated the substantial computational time savings accorded by the SA based numerical solution process when compared to a well-established method, thus demonstrating the scalability of this framework. Future work will focus on testing this framework in a densely cluttered urban environment, using a non-sequential MANET communication techniques, and outdoor experiments with upwards of 20 quadrotors.

CHAPTER 5

SECURE COMMUNICATION FOR MULTI-QUADCOPTER NETWORKS USING ETHEREUM BLOCKCHAIN

5.1 Introduction

Paper: *P. Abichandani, D. Lobo, S. Kabrawala, and W. McIntyre. "Secure communication for multiquadrotor networks using Ethereum blockchain," in IEEE Internet of Things Journal, 8(3), 1783-1796, 2020*

Blockchain is an open-source, distributed, immutable ledger that maintains a record of every information transaction among peers in a network. Every peer in the network has a copy of this ledger. It enables transactions between peers without an intermediate trusted central authority and verifies the transactions with the same amount of certainty as a central authority. The trustlessness provided is a key benefit of Blockchain technology – it eliminates the need for an intermediary to govern and verify interactions on the network. Blockchain technology has been applied extensively in multiple industries, including finance, medical, energy, and real estate [157, 158, 177, 231, 248, 337]. A slew of recent applications illustrate the

Figure 5.1 Three DJI M100 drones were fitted with the required hardware for a three-node Ethereum network.

growing ubiquity of blockchain based software systems for artificial intelligence, 5G, IOT security, identifying deepfake, and package delivery [57,125,161,301,338]. Figure 5.1 depicts three quadrotors flying in formation and as part of the Ethereum network.

An important aspect of blockchain technology is the implementation of smart contracts. A smart contract is an executable software program that governs the transactions between peers in the network. Rules for the network can be programmed into smart contracts to automate sophisticated information-centric tasks. The introduction of Ethereum Blockchain in 2013 provided the ability to program smart contracts in a Turing complete language called Solidity. Carefully developed smart contracts enable autonomous systems comprised of multiple agents (peers) that can perform decentralized decision making in applications such as IOT (Internet Of Things), multi-robot systems, smart cities, and artificial intelligence [68, 301].

In a multi-robot system, each robot performs a specific task as the group works together towards completing a larger goal. Multi-robot systems have been used in precision farming, creating interactive displays, IOT networks, and industrial robotics

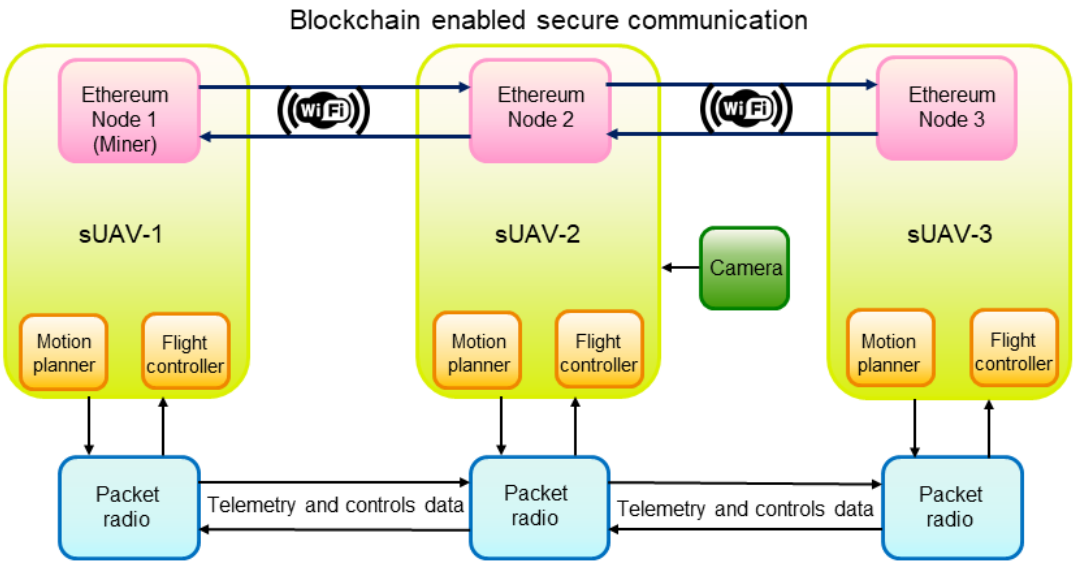


Figure 5.2 Each node included the flight control hardware and a separate computer running an Ethereum blockchain node. Flight control data was communicated via a 915Mhz mesh network. Ethereum blockchain data was communicated via WiFi.

[116, 179, 195, 244]. Blockchain can provide security for trust-sensitive multi-robot systems in the form of data confidentiality, integrity, entity authentication, and non-repudiation [106]. The distributed ledger of a blockchain ensures that decision making is distributed and collaborative missions can be easily programmed while eliminating a single point of failure for data storage.

This chapter reports on experiments involving a network of multiple quadrotor sUAVs that use Ethereum blockchain for securely sharing image data. Figure 5.2 depicts the conceptual architecture of the system. Figure 5.3 depicts the detailed hardware used to implement the blockchain-based communication system. The main experiment performed in this work is image transfer across a network of 3 Ethereum nodes in a networked multi-quadrotor flight. The image is acquired using a camera

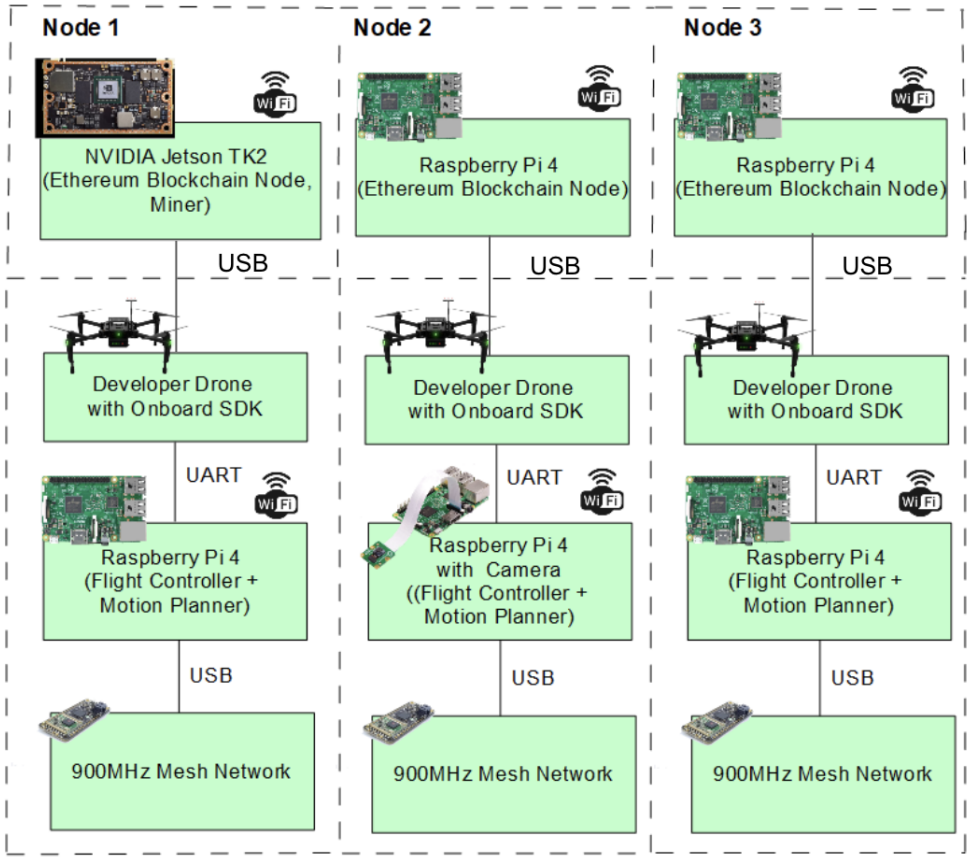


Figure 5.3 Hardware architecture for each each Node including the flight hardware and hardware running the Ethereum blockchain.

onboard one of the quadrotors. The quadrotor platform used in this study is the DJI Matrice M100. Each quadrotor was fitted with a transceiver radio module operating in the 915MHz license-free frequency bands, providing half-duplex bi-directional RF links at 300Kbps. The 915MHz network was used to transmit motion planning data. The Ethereum network hardware was added to the quadrotor hardware stack and used WiFi for establishing communication over the blockchain.

The main contributions of this work are:

1. Experimental validation of a multi-quadrotor Ethereum blockchain system for secure data transfer.

2. A software architecture that seamlessly integrates the Ethereum software stack with a quadrotor flight control stack.

3. Data-transfer time measurements for flight-tests involving different image sizes, Ethereum consensus algorithms, and varying difficulty levels.

4. Comparison of three different WiFi routers and its affect on the blockchain network.

5. Power consumption analysis due to increased payload by mounting blockchain hardware on the sUAV.

The remainder of this chapter is organzied as follows. Section 5.2 discusses existing literature with Ethereum blockchain applied to robotic systems. In Section 5.3, the hardware and software architecture used in the experimentation is described. Section 5.4 provides in-depth results of the experiments. Section 5.5 provides discussions about the results and future directions.

5.2 Related Work

The body of work that applies blockchain technology in robotics applications continues to grow [11,32,51,104,200,233,388]. A majority of these studies have covered blockchain-based robotic system design and associated simulations. A common theme

in these studies is improved security, behavior differentiation, and data integrity provided by using blockchain technology to enhance robotic operations [106].

5.2.1 Simulation Studies

The intersection of blockchain technologies and robotics has been the focus of several simulation studies [10, 11, 32, 153, 156, 233, 336, 388]. In [156], the authors simulated a communication protocol for multi-agent systems to participate in business activities using Ethereum blockchain and smart contracts. The communication protocol consisted of smart contracts interacting with autonomous agents running the Robot Operating System (ROS). The verification of transactions took place using Air-token and Ether [12]. The authors used the proposed communication protocol for an sUAV employee management system that consisted of a dispatcher node and an Air Traffic Control (ATC) node.

In [336], the authors simulated an approach to establish secure swarm coordination mechanisms for a group of robots and exclude byzantine members using smart contracts deployed on the Ethereum blockchain. The proposed approach was simulated in ARGoS (Autonomous Robots Go Swarming) robot swarm simulator interfacing with a Geth Ethereum client [267]. Smart contracts were used to register the robot members, define movement strategy, and select a decision making strategy using a voting system to create a consensus among the members. The smart contract implemented strategies that included a member's time limit to cast a vote, a renewal of the decision strategy each time a new strategy was selected, and check on different blockchain versions by verifying the hash value of the blocks. The performance of the blockchain approach proved superior over classical swarming approaches [359].

In [388], the authors proposed a cloud system architecture for an sUAV and sensors designed for surveillance of a dam site. The sUAV collected these data and delivered them to the dam monitoring center. The authors used the Bitcoin

framework and the Proof-of-Work (PoW) consensus protocol to provide data integrity, traceability of the sensor data in the wireless sensor network. The authors simulated several scenarios to demonstrate the time delays for secure sensor data.

In [11], the authors proposed a communication model based on the Ethereum blockchain for the Internet-of-Drones. Ethereum blockchain provided authorization, data integrity, and authentication for the data collected by the drones in the system model via Proof-of-Stake (PoS). A single drone was selected to be the forger node responsible for authenticating and validating the blocks in the blockchain. The computation cost for block creations was calculated as 384 bits, of which 128 bits represented the identity, and 256 bits is the hash value. The computation time for block creation and validation was found to be 0.023 milliseconds.

In [32], the authors developed a Bitcoin blockchain-based sUAS for monitoring and surveillance of critical infrastructures. Blockchain provided security against three types of breaches: adversaries targeting software, adversaries targeting hardware, and adversaries targeting communications. The authors provided a performance evaluation of the proposed system using the NS-3 simulator over an area of $16km^2$ with four ground control stations. The authors programmed the occurrence of ten events around five randomly distributed critical infrastructure. Detection rates in the order of 95% were observed in simulation with lower false alarm rates as compared to classical approaches [33, 159]. The authors concluded that the high detection ratios and reduced overhead showed that a blockchain-based UAS could provide accurate decisions which are crucial in monitoring critical infrastructure.

In [153], the authors proposed a reputation-based data sharing scheme for secure data sharing among vehicles to overcome the security vulnerabilities of vehicular edge computing servers. The scheme was based on a consortium blockchain where the nodes were pre-selected in a public blockchain to establish a shared and distributed database with smart contracts used to enable data management automation between

the vehicles. Security analysis of the proposed scheme ensured the security of the data storage and high-quality data sharing over traditional reputation schemes.

In [10], the authors proposed an Ethereum blockchain-based decentralized network to monitor, control, and log workflow events for a Cyber-Physical Production (CPP) System. The machines in the proposed CPP system communicated by exchanging cryptocurrency tokens. An essential feature in the system was that the cryptocurrency tokens would be distributed between the machines well in advance, according to the priority of services, thus creating a consensus where Nothing is at Stake automatically. The authors proposed implementing their system for a PCB manufacturing plant by determining the rules of operation.

In [233], the application of a multi-robot path-planning algorithm using blockchain technology was investigated. A Probabilistic Road Map (PRM), path planning algorithm, was implemented alongside Hyperledger Fabric, an enterprise-grade blockchain platform [137]. The average latency to commit a transaction to the Hyperledger Fabric was 112.46 milliseconds, thus validating that a blockchain platform has the potential for enabling secure and scalable distributed systems.

5.2.2 Experimental Studies

Experimental studies involving blockchain technology for robotics applications have been recently published [75, 105, 200]. The key to these studies is the integration of a blockchain software stack with a robotic system such as an industrial robotic manipulator arm, ground robot platforms, and quadrotors. Network latency is a crucial metric to be tracked in experimental studies as it directly affects the data refresh rates of a robotic system. Figure 5.4 shows the experimental platforms used in the representative experiment oriented works.

In [200], the authors proposed an experimental architecture to control a UR3 robotic arm where the robotic events are logged on the RobotChain blockchain and

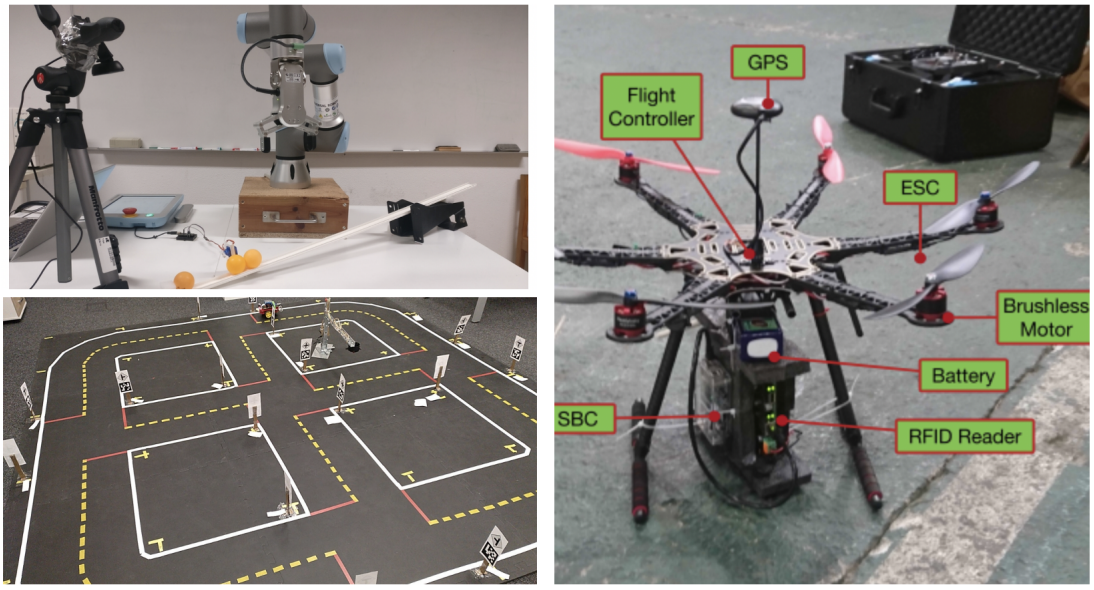


Figure 5.4 Prior experimental studies with blockchain and robotics involve use of robotic arms (top-left) [200], ground robots [75] (bottom-left), and single multi-robot sUAVs with RFIDs (right) [105].

its movement controlled with a smart contract program [104]. Logic programmed in the smart contract sent instructions to a robotic arm to pick and place objects based on data processing of images obtained using an RGB camera by an external computer. The arm speed in picking up the objects was measured to validate that the proposed blockchain-based architecture was capable of controlling the robotic arm in real-time.

In [75], the authors provide experimental validation procedures to identify flawed liability executions in order to suspend payments to questionable service providers using blockchain in a Duckietown environment [261]. The consensus protocol on blockchain technologies provides a method to detect malfunctioned agents in a network when agent behavior goes against the behavior consensus defined by the members in the network. The blockchain platform used in this work was Ethereum. A prototype of the proposed system was implemented where smart contracts validated the liability of the autonomous agents and directed their movements.

In [105], the authors document experiments for an automated inventory management system using RFID, sUAV, and Ethereum blockchain technology. Similar to the results presented in this paper, the authors evaluated two consensus protocols for the Ethereum blockchain, Proof-of-Authority (PoA) and PoW. A quadrotor sUAV was manually controlled to fly around the warehouse and tag the inventory items using the RFID tags. The time taken to identify the inventory items using a quadrotor and store their identification value on the database was measured. It was shown that the time taken to validate blocks averaged at around 15 seconds for PoA while for PoW, the validation times varied significantly from 5 seconds to 70 seconds.

These experimental works have paved the way for more sophisticated experimental robotics studies involving blockchain technology. Blockchain technology has also been applied to Internet-of-Things (IoT) systems to provide secure and decentralized operations in healthcare, agriculture, smart cities, and vehicular networks [109, 191, 203, 234, 319, 383, 386].

This chapter addresses the gap and expands on the above body of work by focusing on the physical implementation of blockchain technology for multiple quadrotor sUAVs operating outdoors. Specifically, this chapter documents the hardware and software architecture; the quadrotor, computing and radio hardware used; and the software development tools used to implement the system. On the experimental front, the chapter documents the results of 4 key studies: 1. the effect of varying blockchain difficulty on data transfer rates between 3 quadrotor sUAVs, 2. the effect of varying data size on data transfer rates between 3 quadrotor sUAVs. 3. the effect of Ethereum network disruption on between 3 quadrotor sUAV network connectivity, and 4. the effect of increased payload (Ethereum network hardware) on the flight-times of the quadrotor sUAV.

5.3 System Architecture

This section elucidates the experimental platform used for the Ethereum-based communication protocol used to exchange image files.

5.3.1 Quadrotor Setup for Motion Planning

Three DJI Matrice M100 quadrotors, referred to as Node 1, Node 2, and Node 3, were used [87]. Each quadrotor setup hosted a communication Node in the Ethereum blockchain network. The hardware setup on the quadrotor is the same as depicted in Chapter 2.4 Figure 3.8.

The quadrotors came equipped with a proprietary DJI M100 N1 flight controller with inertial sensors and GPS. The N1 flight computer was responsible for position control using velocity and yaw-rate inputs. Each quadrotor provided a real-time telemetry stream of accelerometer, gyroscope, magnetometer, and GPS data. The quadrotor position was defined in a local frame using latitude, longitude, and height with a pre-defined origin point. For motion planning and control, each quadrotor Node was equipped with a Raspberry Pi 4 computer (Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz, 4GB LPDDR4-3200 SDRAM). Motion commands and telemetry streams were communicated between the Raspberry Pi and DJI Flight stack using a UART connection using software function calls provided by the DJI's onboard C++ SDK. The flight processing Raspberry Pi 4 for Node 2 was equipped with a V2 camera module for image capture with 8 Megapixel still image resolution.

Each quadrotor was equipped with a 915 MHz transceiver radio module used for half-duplex bi-directional communication between quadrotors for motion data telemetry using RF links at 300Kbps. A round-robin scheduling technique for sequential wireless communication was implemented for the fair distribution of wireless bandwidth and fault tolerance. As part of this technique, only one quadcopter transmitted its information at a time. Figure 5.5 depicts the sequence

of information transferred between the quadrotor nodes. This information payload contained position and velocity data and was received by the remaining quadrotors either through one-hop direct communication or through indirect multi-hop relaying. The order in which the quadrotors communicated in the round-robin was randomly determined at the beginning of the mission.

The positioning of the quadrotor was defined in a frame with a pre-defined GPS origin coordinate, O, consisting of a latitude (radian), longitude (radian), and height from the ground (m). A quadrotor's behavior of motion during the experiments was broken into three components: Take-off, Offset Alignment, and Waypoint Following.

The first stage for each quadrotor was taking off to a height of 1 meter. The quadrotor would then move to the pre-defined origin coordinate, O, and then to its defined offset position, which included a height of 5 meters for all experiments. Once all three quadrotors were at their assigned offsets, communicated through the transceiver radio modules, the waypoint Following stage would begin in which all three quadrotors would proceed to the assigned waypoint, while maintaining their offsets.

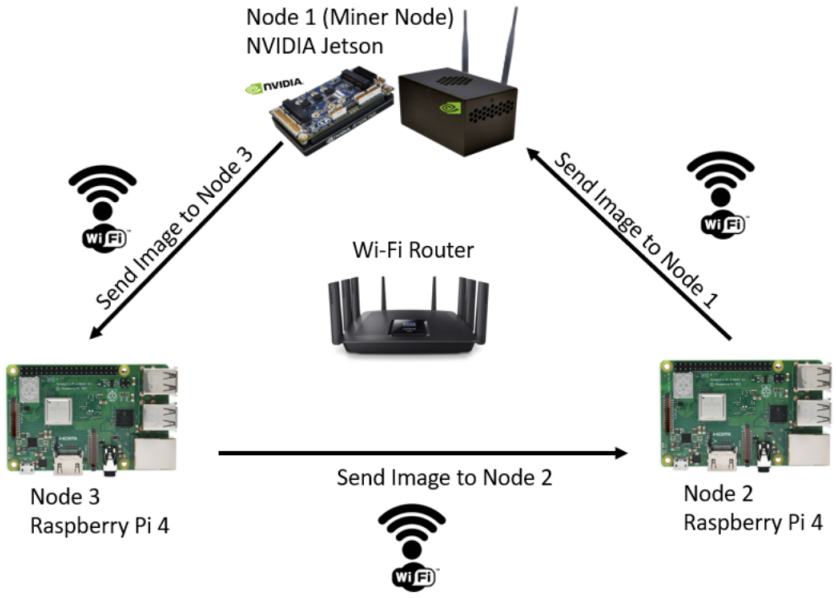


Figure 5.5 Three Ethereum Nodes were setup using 1 Nvidia Jetson TX2 and two Raspberry Pi 4 computers. The nodes ran an Ethereum Virtual Machine (EVM) and communicated with each other using a 802.11 network setup using WiFi routers.

A set of custom spline generation functions, implemented in Python, were used to generate minimum snap 7^{th} order splines passing through N pre-selected waypoints [5]. A ground station computer hosts a web-based user interface for mission configuration, clock synchronization, and mission upload.

5.3.2 Ethereum Hardware Setup

The Ethereum network was implemented using a separate set of computers that were mounted on the quadrotors. Keeping the blockchain computing hardware separate from the flight control hardware ensures modularity and avoids taxing any single computing system.

Miner Node: The transition of the Ethereum blockchain state takes place when a block is deemed valid. The validity of a block is determined through the process of mining. Node 1 was equipped with an NVIDIA Jetson TX2 computer (Quad-Core ARM Cortex®-A57, 256-core NVIDIA Pascal™ GPU, 32GB eMMC 5.1, 8GB 128-bit LPDDR4 Memory). The Jetson computer was selected as a miner due to its appropriate computational resources for validating and adding new blocks to the Ethereum. The Jetson was mounted on an Orbitty carrier board, and the assembly was mounted in a black colored OrbittyBox [252]. The miner node was also designated as the bootstrap node and was responsible for forming the overlay network with the nodes. The remaining nodes connected with each other over the TCP port of the bootstrap node.

Other Ethereum Nodes: Node 2 was equipped with a Raspberry Pi 4 computer (ARM Cortex - A72, 1.5 GHz, 4GB LPDDR4 Memory) and acted as a node in the Ethereum network. A separate process on the flight processing Raspberry Pi 4 used openCV for image capture. The captured images were transferred to the corresponding Ethereum blockchain node 2 Raspberry Pi 4 using a Python-based

Table 5.1 Range and Bandwidth for Radios used in the Experiments

Type	Radio Network	Range	Bandwidth
WiFi	Verizon mobile hotspot	20m	12 Mbps
WiFi	Tp-Link AC1750 Wireless Dual Band Gigabit Router	33m	1300 Mbps
WiFi	Linksys EA9500 Max-Stream AC5400 MU-MIMO Gigabit WiFi Router	45m	2166 Mbps

server-client WebSocket connection. Nodes 3 was also equipped with a Raspberry Pi 4 computer and acted as a node in the Ethereum network.

Radio Hardware: The quadrotors were equipped with RFM69HCW packet radios that operated at 915MHz. This low-bandwidth (up to 300Kbps), mesh network system was used to share controls, telemetry, and motion planning data and provided a 500 meter range.

The Ethereum network was established with the help of 2.4GHz WiFi routers. Given that the Ethereum node hardware was hosted on a set of sUAVs, it was prudent to evaluate options that provided network connectivity with respect to their mobility. Accordingly, three different WiFi routers were evaluated. Each experiment used a specific type of WiFi router. These included: a Verizon 4G LTE Wi-Fi mobile hotspot, a AC1750 wireless dual band gigabit router from TP-Link, and a EA9500 Max-Stream AC5400 MU-MIMO Gigabit WiFi Router from Linksys [194, 354, 362]. Table. 5.1, provides the ranges and bandwidth of these Wi-Fi routers.

The Verizon hotspot hardware provided a most mobile option out of the three as it could be mounted on the quadrotor itself. However, the bandwidth provided was relatively lower than the other two options.

5.3.3 Consensus Algorithm

Consensus algorithms provide blockchains with its characteristic features of decentralization – trustless security, immutability, privacy, and transparency. The Proof-of-Work (PoW) consensus algorithm is widely used in Bitcoin and Ethereum [168].

Other consensus algorithms include Proof-of-Stake (PoS) [95], Proof-of-Authority (PoA) [249], Practical Byzantine Fault Tolerance (PBFT) [162]. In our experiments, we evaluate the effect of PoW or PoA algorithms on time taken to transfer images through the Ethereum network. The PoW algorithm is based on the SHA-256 cryptographic hash function [380]. The PoA algorithm is implemented in Ethereum through a protocol called Clique [249]. An issue with PoW is that it is impossible to control the mining frequency and control the block times. On the other hand, PoA block validation times are consistent and do not vary significantly.

5.3.4 Ethereum Software Setup

A private blockchain network was set up between the three nodes (Nvidia Jetson and two Raspberry Pi) in the network with individual read and write permissions given to each node. Each node ran an Ethereum Virtual Machine (EVM) [96]. The EVM is the Turing complete virtual machine that processes and handles transactions being carried out in the Ethereum blockchain. Since this was a private blockchain, the identity of each Ethereum node was known and verifiable.

Geth Ethereum Client: The Geth Ethereum client was used in our software development. Geth is written in the Go language. There are two types of accounts in a blockchain: Externally Owned Accounts (EOA) and Contract Accounts (Smart Contracts). An Ethereum account is defined by a pair of keys, private key, and a public key, and each account is indexed by its address, which is derived from the first 20 bytes of the SHA3 hashed public key. [97]. In a private network, it is assumed that the participant node identities are known, and therefore, the Ethereum addresses are added to an access control list at the beginning of the smart contract code. The nodes in the private Ethereum network access the read and write functions of the smart contract based on their Ethereum address definition in the access control list. The smart contract generates warning events in the case that an unlisted Ethereum

address made an attempt to interact with the smart contract. A similar approach to providing trust management between the nodes in the network using an access control list to ensure trust between the nodes in the network can be found in [210]. A limitation in the approach, however, is the inability to dynamically update the access control list due to the permanent nature of the smart contract. Using the Geth console, an EOA account was created for each of the Nodes 1, 2, and 3.

Truffle Development Environment: The Truffle development environment was used to develop, verify, and deploy our smart-contracts. Truffle is a command-line interpreter (CLI) tool with a built-in compiler for smart contracts, automatic contract testing, scriptable, extensible deployment, and migrations framework. The Truffle environment provided methods to interact with the deployed smart contracts. An automation script written in Javascript called instances of the functions in the smart contract. Each node featured its automation script to interact with the smart contract. This approach was adopted to maintain the secure communication protocol and ensure that each node calls only the functions in the smart contract meant for it. The smart contract was coded in Solidity, which is an object-oriented programming language. After compiling the smart contract code, it is translated to bytecode, which is executable in the EVM.

5.3.5 Ethereum Smart Contract

Genesis Block: The first block in the Ethereum blockchain is called the Genesis block. The Genesis block contains parameters that define the blockchain. The Genesis file used in our experiments was written in JSON and is depicted in Figure 5.6. The blockchain parameters initialized in the genesis file included:

- `Difficulty`: This parameter is a measure of the computational complexity of mining a block to find the hash value. It determines the speed of mining of the blockchain network and can be used to calculate the deviation from the expected block time [85].

- `Gas Limit`: The maximum amount of `gas` that the sender is willing to spend for a particular transaction. Gas is a unit used to measure the effort required for a particular computation in an Ethereum Virtual Machine (EVM). Gas price is the value the transaction sender is willing to pay per `gas` unit and is measured in *Gwei*. Ether is the token used to pay for `gas`.

- `Alloc`: The alloc parameter is only used in the case of a private blockchain. This parameter is used to allocate funds (Ether) to the respective accounts in the network. The allocated fund `balance` should be significantly higher than the `gas` limit such that the accounts do not run out of `gas` when executing transactions. The allocated funds do not have any value outside of the private blockchain.

Nodes with the same Genesis file are connected to the same private blockchain network to maintain proper synchronization of the blocks. For these experiments, four genesis files were created with increasing orders of difficulty.

The block difficulty values were set to the following values:

- Difficulty 1: 0x1
- Difficulty 2: 0x10000
- Difficulty 3: 0x100000
- Difficulty 4: 0x1000000

The `alloc` and `gasLimit` parameters of the genesis file were left unchanged. Each run of the experiment was conducted using one of the genesis files copied in

```
{
        "config": {
                "constantinopleBlock": 0,
                "petersburgBlock": 0
        },
        "difficulty": "0x10000",
        "gasLimit": "0x8000000",
        "alloc": {
                "0x8FA8c9E58BcF0201f927b175363365ee71F5d621": {"balance":"1000000000000000000"},
                "0xE168fa9d240c552257C0860AcC60684a3c347866": {"balance":"1000000000000000000"},
                "0xa081657c7a35399e5D24E08a90C1f631834b429C": {"balance":"1000000000000000000"}
                        }
}
```

Figure 5.6 Example Genesis file with Ethereum blockchain parameters.

each of the nodes. Once the nodes were connected to the same private network and synced together, Node 1 then deployed the smart contracts to the private Ethereum blockchain.

In a private blockchain, it is up to the developer/administrator of the system to decide the gas price for each unit *Wei*. A private Ethereum network is not part of the public blockchain, so the gas prices are not affected by the publicly traded value of Ether. In private Ethereum networks, the miners do not need gas as an incentive and are thus configured to be zero or free gas [136]. In the system documented here, the gas price was set to 0 when setting up the geth environment, which means that the nodes were able to post transactions while the miner received 0 as payment.

Smart Contract Functions The smart contracts used in our experiments were written in Solidity Version 0.6.7. The contracts allowed for the secure transmission of images between different nodes in a manner that ensured that only the intended recipient received a specific image.

Figure 5.7 depicts the smart contract and decentralized storage functionalities that were encapsulated in `Get()` and `Set()`functions. The decentralized storage was implemented using Inter Planetary File System (IPFS). The Get() functions were used to read an image hash, while the Set() functions were used to store the image hash on the blockchain network. The Truffle scripts compiled and deployed the smart contract into the blockchain environment. Interactions with the smart contract on the Ethereum took place through transactions verified by the miner.

Also, shown in Figure 5.7 is the flow of data through the Ethereum network. Information about each node's user account was stored in the smart contract. A node could only access functions designated to it depending on its node number in the network.

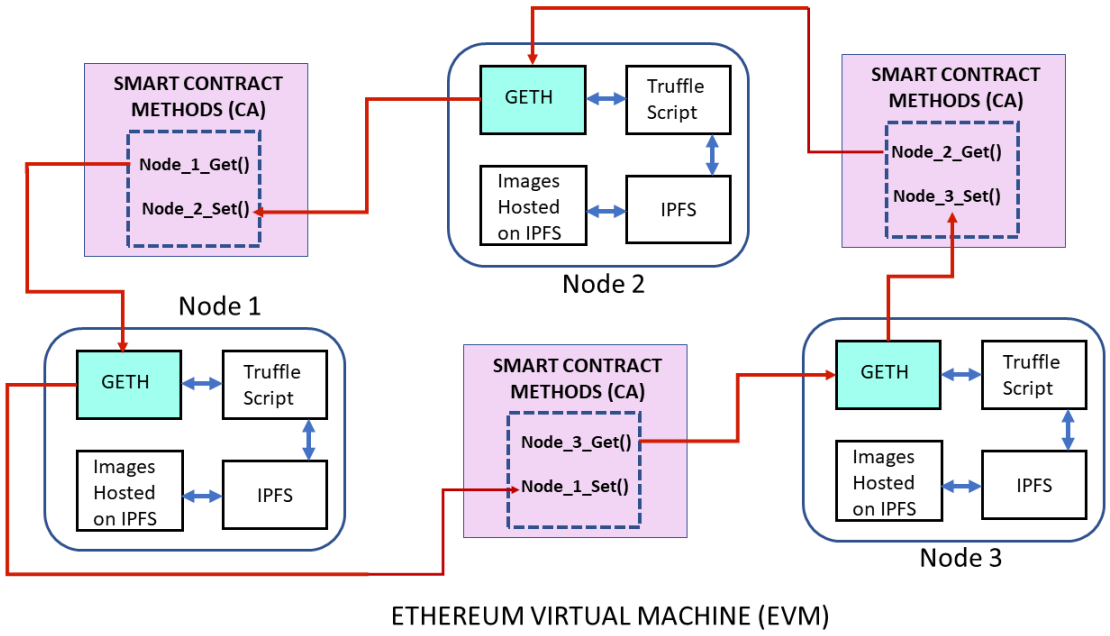


Figure 5.7 The Ethereum software setup used in our experiments and the smart contracts featured secure functions to ensure images can only be sent and received by pre-determined nodes.

Node 2 was the first node to receive a new image captured by the camera. This image was stored in a local folder on Node 2 and would be uploaded to the IPFS using a bash script. Once an image was uploaded to the IPFS setup, an image hash would be generated. Node 2 would use the `Node_2_Set()`function to store the IPFS hash in the smart contract. Node 1 can then use the `Node_1_Get()` functions to read the IPFS hash stored in the smart contract. This process was repeated for all three nodes. The hash obtained could then be used to access the image on the IPFS server. Important debugging messages were printed to the console for the developers and users to track the transfer of the image hash from one node to another.

A JavaScript program was used to automate the task of interacting with the smart contract using the Truffle environment. The JavaScript on each Node was executed when the bash script to upload images to the IPFS was executed. The smart contract code developed for the system can be found on the RADLab Github repository [77].

5.3.6 File Storage

A limitation of blockchain technology is that it is computationally expensive to store data on it [14,380]. The Interplanetary File System (IPFS), a peer-to-peer distributed file-sharing system, was used for storing the images [176]. IPFS by itself provides a tamper-proof method of storing and sharing data between nodes in a network. However, it does not provide any method to timestamp when the data is added to the network, which is vital information when performing missions using multi-robot swarms. Another security flaw of IPFS is that anyone with a copy of the root content identifier (CID) has access to the data, which is problematic when sharing sensitive files between robots.

By integrating Ethereum blockchain with IPFS along with a layer of encryption to the data, it is possible to build a secure information sharing system that overcomes these drawbacks. Figure 5.8 depicts the detailed steps of the data-flow involving encryption, IPFS, and Ethereum. After an image is captured, the node applies asymmetric encryption protocol OpenPGP, which allows encrypting a file with a public key that can be decrypted by only members of the private blockchain, which

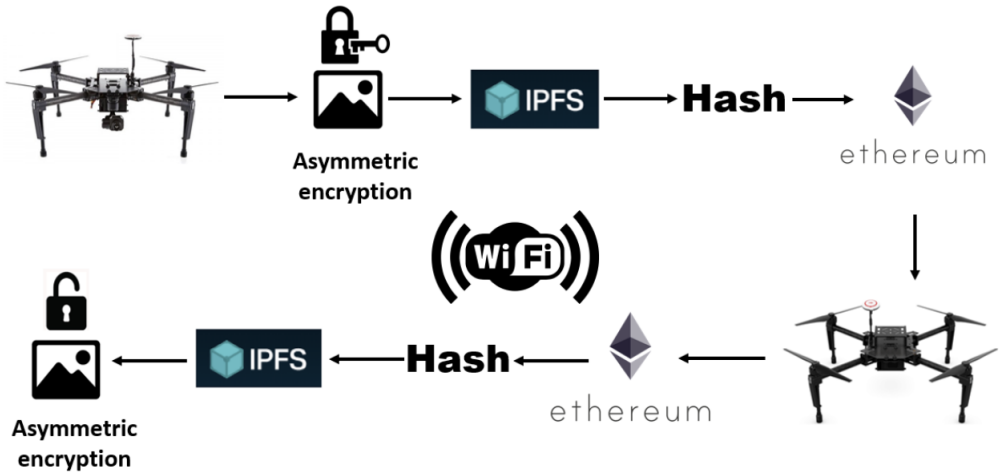


Figure 5.8 Data-flow involved in our experiments involved image capture and encryption, IPFS image upload, and transfer of the image hash over the Ethereum blockchain.

hold the decryption key [251]. After encrypting the image file, it is added to the IPFS, which generates a CID hash, which is stored on the blockchain via the Smart Contract. On the receiver end, the smart contract ensures that the quadrotor node requesting the CID hash is part of the private blockchain. This is ensured by verifying the private key of the EOA requesting the data. After verification, the hash obtained is used to access the data from the IPFS node via a local HTTP gateway. The data downloaded from the IPFS is the encrypted image file. All members of the private blockchain hold the decryption key, which is used to decrypt the downloaded image.

5.4 Experimental Results

The experiments focus on studying the effects of image size, consensus algorithm type, and blockchain difficulty values on-time τ_{image} taken to transfer images across the multi-sUAV network successfully. In the following, we document image transfer flight tests that were conducted for 3 minutes each. The quadrotor 2 captures images using its camera and transfers it to the Ethereum Node 2 that is physically mounted on it. The smart contracts are then deployed to transfer these images across the entire network (Node 2 to Node 1, Node 1 to Node3, and finally from Node 3 back to Node 2). τ_{image} is the total time taken for this image transfer across the entire Ethereum network.

Three wireless network routers were used to create three different sets of experiments. These included a Verizon 4G LTE WiFi mobile hotspot, a Dual-band WiFi 5 router from TP-Link, and a Multi-User MIMO Gigabit router from Linksys. The Verizon 4G LTE WiFi mobile hotspot provided the most mobility as it was mounted on one of the quadrotors. On the other hand, the Linksys router provided the strongest WiFi connection outdoors due to its long-range MIMO configuration. For all image transfer experiments, two consensus algorithms were tested – Ethash (Proof-of-Work) and Clique (Proof-of-Authority).

Figures 5.9 and 5.10 depict the results of our experiments. In each of these graphs, the τ_{image} is averaged across multiple runs of the experiments and expressed in seconds.

5.4.1 Effect of Varying Image Size on τ_{image}

The images from the Node 2 quadrotor camera were resampled to different sizes (100kB, 500kB, 750kB, 1000kB, 2000kB, and 3000kB) before being transferred over the Ethereum network. The images were sent in increasing order of their resampled sizes. Figure 5.9 depicts the τ_{image} values in seconds averaged across all experimental runs and difficulty values for the resampled images for the three different router types. As observed in Figures 5.9a, 5.9c, and 5.9e, the average τ_{image} varied between 1.66 seconds to 3.01 seconds when PoA was used for consensus. In contrast when PoW was used as the consensus protocol, Figures 5.9b, 5.9d, and 5.9f, the average τ_{image} values varied significantly between 5.69 seconds to 28.57 seconds when PoW was used for consensus.

5.4.2 Effect of Varying Block Difficulty on τ_{image}

Figure 5.10 depicts the average τ_{image} values in seconds as a function of difficulty values for the three different router types. As mentioned in Section. 5.3.5, 4 different difficulty levels were used in our experiments. The τ_{image} is averaged across all experimental runs and image sizes for the resampled images and compiled in Figure 5.10. As observed in Figures 5.10a, 5.10c, and 5.10e, the average τ_{image} varied between 1.89 seconds to 2.87 seconds when PoA was used for consensus. In contrast when PoW was used for consensus Figures. 5.10b, 5.10d, and 5.10f, the average τ_{image} values varied significantly between 2.8 seconds to 142.92 seconds.

5.4.3 Effects of WiFi Communications Disruption

In case an Ethereum node mounted on a quadrotor loses WiFi communication with its neighboring nodes, it is important that communication between the remaining nodes continues uninterrupted. This section documents the observed behavior of the Ethereum network in the face of such WiFi communication disruption. The experiments were performed using a hardware-in-the-loop (HWIL) setup. The HWIL approach provided the ability to conduct safe and controlled experiments as per the guidelines of the United States Federal Aviation Administration (FAA) [101]. All HWIL experiments were performed using the DJI Assistant 2 software. The software provided a real-time emulation of the DJI M100 rigid body dynamics to provide telemetry outputs, and velocity and yaw rate inputs into the N1 flight controller.

The three quadrotors were commanded to fly to an altitude of 20 meters above the ground and maintain a triangular formation. Once the quadrotors reached the desired altitude the quadrotors start sending the image hash file among them. After a period of 1 minute one of the nodes was commanded to break flight formation and fly out of range of the remaining nodes. When a time period of 30 seconds elapsed the quadrotor was commanded to fly back in range of the other nodes.

The following two scenarios were tested:

Miner node disruption In the first scenario, the miner node, Node 1 was commanded to break flight formation and go out of WiFi range. Node 1 was also the bootstrap node. The transactions posted by the remaining nodes could not be validated by the miner and thus no new blocks were added to the blockchain. The communication links between the nodes were disrupted and the file transfer between the nodes terminated. On Node 1, the Geth console and the Truffle program terminated, thus disrupting the entire blockchain network between the nodes.

Non-miner node disruption In the second scenario, a non-miner node (Node 3) was commanded to break formation and go out of WiFi range. The rest of the nodes remain unaffected and continued to interact with the smart contract. Due to the round robin arrangement Node 1 and Node 2 continued to exchange information over the blockchain. When the Node 3 came back in range of the blockchain network it was able to rejoin the Ethereum network. This is due to the fact that the Ethereum nodes connect with each other over the TCP port of the bootstrap node. Thus when the Node 3 was within Wi-Fi network range it was able to rejoin the Ethereum network by connecting to the TCP port of Node 1.

5.4.4 Flight Duration and Battery Life

The flight time of multi-rotor sUAVs is critical for tasks such as surveillance, transportation, and search and rescue operations. As such, it is vital that the flight time of blockchain enabled multi-rotor sUAVs are not significantly hampered due to the increased payload. Outdoor flight tests were conducted to observe the difference in flight times of the multi-rotor sUAV with and wihout the Ethereum payload. A total of 5 outdoor flights were conducted where the quadrotor was flown to an altitude of 20m and commanded to hover until the battery ran low. The DJI M100s were equipped with the TB48D battery that has a capacity of 5700mAh [88]. It was found that without the Ethereum payload, the average hovering flight time was approximately 21 minutes and 27 seconds. The non-miner node payload with the Raspberry Pi and associated battery weighed 147.41 grams with an average flight time of approximately 19 minutes and 48 seconds. The miner node payload with the NVIDIA Jetson and associated battery weighed 997 grams with average flight time of approximately 12 minutes and 23 seconds.

5.5 Discussions and Future Work

Several observations are made from the above experiments:

1. For PoA, the difficulty level does not have any significant impact on the average τ_{image}. However, as expected, the average τ_{image} increases significantly with an increase in the difficulty levels for PoW.

2. At Difficulty 4, it was observed that the time taken to validate the transaction increased significantly for PoW. The time to validate transactions could not keep up with the rate at which images were being uploaded to the IPFS. Only 50% images were transmitted across the network when the TP-Link router was used. Moreover, 33% of the images were successfully transferred when the Linksys router was used, and 66% images were successfully transferred when the Verizon mobile hotspot was used.

3. Overall, it was observed that PoA provided a lower average τ_{image} compared to PoW. This behavior is in line with the fact that PoA does not rely on the mining process to verify transactions. On the other hand, PoW relies on the nodes using their computational resources to solve the mining problem to validate transactions in a block, often taking up significant time. As such, PoA can validate transactions quicker than PoW as time goes on and proves to be the faster alternative for data transfer in multi-robot systems.

4. When an Ethereum node experiences WiFi issues or disruptions, it is important that the remaining nodes continue the data transfer without getting affected. A miner node losing WiFi connection can be catastrophic to the network as is observed by our experiments. Having more than one miner node in the network can add redundancy and provide protection against such failures. On the other hand, a non-miner node has the ability to drop out and rejoin the Ethereum network. All the logic that is required to ensure uninterrupted communication in the case of a non-miner node dropping out and rejoining can be coded in the smart contract.

5.5.1 Security Analysis

The security of the smart contracts proposed in this study was evaluated along the dimensions of data confidentiality, integrity, and non-repudiation [19, 124].

- Confidentiality: Given the fact that this was a private Ethereum network, the Ethereum account addresses of the participating quadrotor nodes were known beforehand and were added to the smart contract code. Thus only nodes whose address matched those defined in the smart contract could access the `get()` and `set()` functions of the smart contract. This mechanism provided confidentiality for secure information exchange. The trade-off here was the inability to add

more quadrotor nodes dynamically to the network. In [19], the authors provided a novel procedure to automate the task of authenticating nodes requesting to join the network, which will be explored in future iterations of this work.

- Data integrity: Each image was encrypted using an asymmetric encryption scheme OpenPGP, and then uploaded to the IPFS. When uploading the encrypted data to the IPFS, a second layer of encryption is added by passing the data through a SHA-256 algorithm and encoding to base 58. Cryptographic hashes possess important characteristics such as being deterministic, uncorrelated, unique, and one-way. The hash generated when sending the encrypted image to the IPFS was then shared among the nodes using Ethereum transactions. The images exchanged over the network were protected by this double layer of encryption ensuring that the data was not tampered with.

- Non-repudiation: Transactions on the Ethereum blockchain are signed with a digital signature using a private key of the account issuing the transaction. The use of this approach provides three security advantages: (1) a method to validate the Ethereum accounts in the network, (2) a signature that is unique to the account and cannot be forged, and (3) a guarantee of terms of service that ensures that the transaction data cannot be modified. As soon as the miner validated a transaction, it was recorded with its unique timestamp. Hence, no node could deny their actions as it is already recorded in the tamper-proof logs.

5.5.2 Future Work

- *Extension to other blockchain technologies*: In future work, the smart contract presented in this work will be implemented using other blockchain technologies to compare and contrast the performance of multiple blockchain platforms for multi-sUAV communications. The smart contract developed in this work can be extended to other blockchain technologies such as Hyperledger and Rootstock blockchain for Bitcoin or RSK RBTC [84, 297]. Hyperledger Fabric provides permissioned blockchain and features the necessary framework to allow Ethereum smart contracts to run on its blockchain [84]. The smart contract presented here can be extended by creating an EVM wrapper around the Hyperledger burrow to run the smart contract bytecode written in Solidity. A key component included in Hyperledger fabric is the JSON RPC API Fab3 to mimic the web3.js library used by Ethereum DApps – this simplifies the extension process from Ethereum to Hyperledger.

  Similarly, RSK RBTC provides a framework to extend Ethereum smart contracts [297]. The RVM (RSK Virtual Machine) is compatible with Ethereum Virtual Machine (EVM) at the op-code level. This compatibility allows Ethereum smart contracts to run easily on the RSK-RBTC. RVM is also compatible with the tools used to deploy and interact with EVM smart contracts. Detailed instructions to use Ethereum smart contracts on the RSK-RBTC can be found in [297]. Teams developing other blockchain technologies

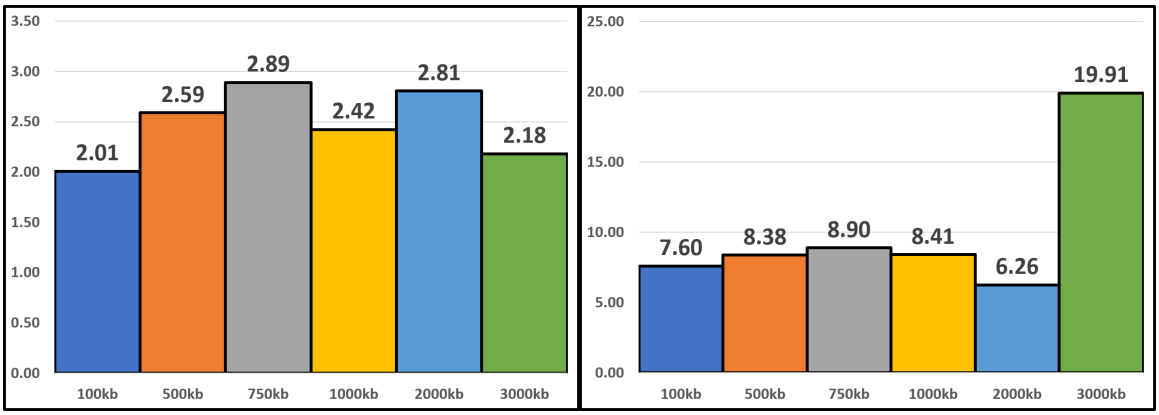
and frameworks such as EOS are working towards creating functionality to extend Ethereum smart contracts to their platforms [74].

- *Integration with multi-sUAV motion planning algorithms*: Integrating a blockchain-enabled multi-sUAV system with a decentralized motion planner is a natural extension to this work. A receding horizon, mixed-integer non-linear programming (RH-MINLP) based motion planner that allows multiple sUAVs navigate along multiple waypoints while securely transferring data over the blockchain will be the next objective of this study [5].

5.6 Conclusion

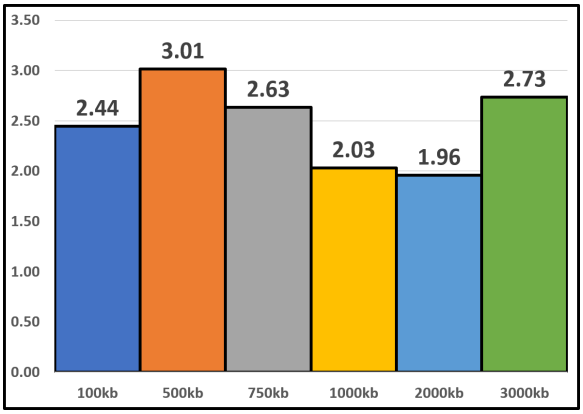
This chapter documented the hardware and software setup, and experimental results for an Ethereum based secure information exchange system mounted aboard multiple quadrotors sUAVs. The hardware architecture leveraged a low-cost NVIDIA GPU for mining during flight and Raspberry Pis as non-miner Ethereum nodes for data collection and storage. The software architecture leveraged Ethereum's ability to program smart contracts for secure, confidential, and tamper-proof data access, and a decentralized file system IPFS for data storage. The system was mounted on 3 DJI M100 quadrotors, and flight tests were performed to collect and securely share images across this 3-quadrotor network. The implementation of a private Ethereum network provided security features such as confidentiality, data integrity, and non-repudiation. The combined use of Ethereum blockchain and IPFS provided a distributed data storage system while avoiding a single point of failure for the network. Experimental results focused on studying the average time taken to transfer an image across the network as a function of image size and consensus algorithm (PoA vs PoW) and the Ethereum difficulty level. The experiments also evaluated the resilience of the Ethereum network in face of WiFi communication disruptions. Finally, the increased sUAV energy consumption due to additional Ethereum hardware payload as inferred by battery life and flight time was documented. The use of the PoA consensus algorithm provided faster image transfer compared to PoW. The image transfer times

point to the feasibility of a blockchain-enabled communication system for multiple sUAVs.

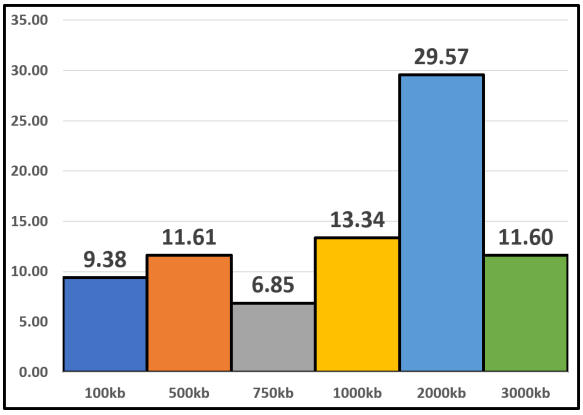


(a) PoA – Linksys Router

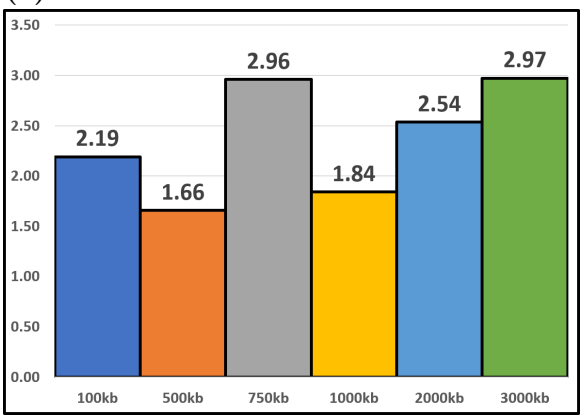
(b) PoW – Linksys Router



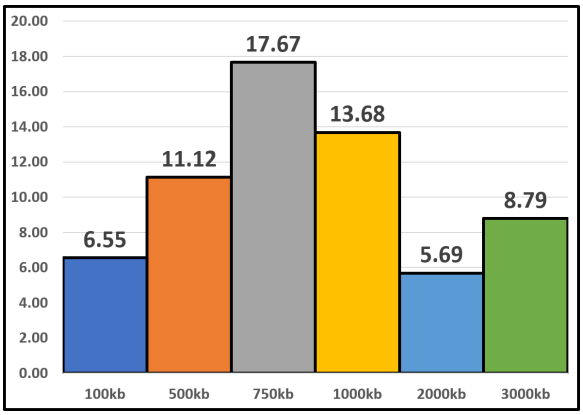
(c) PoA – TP-Link Router



(d) PoW – TP-Link Router

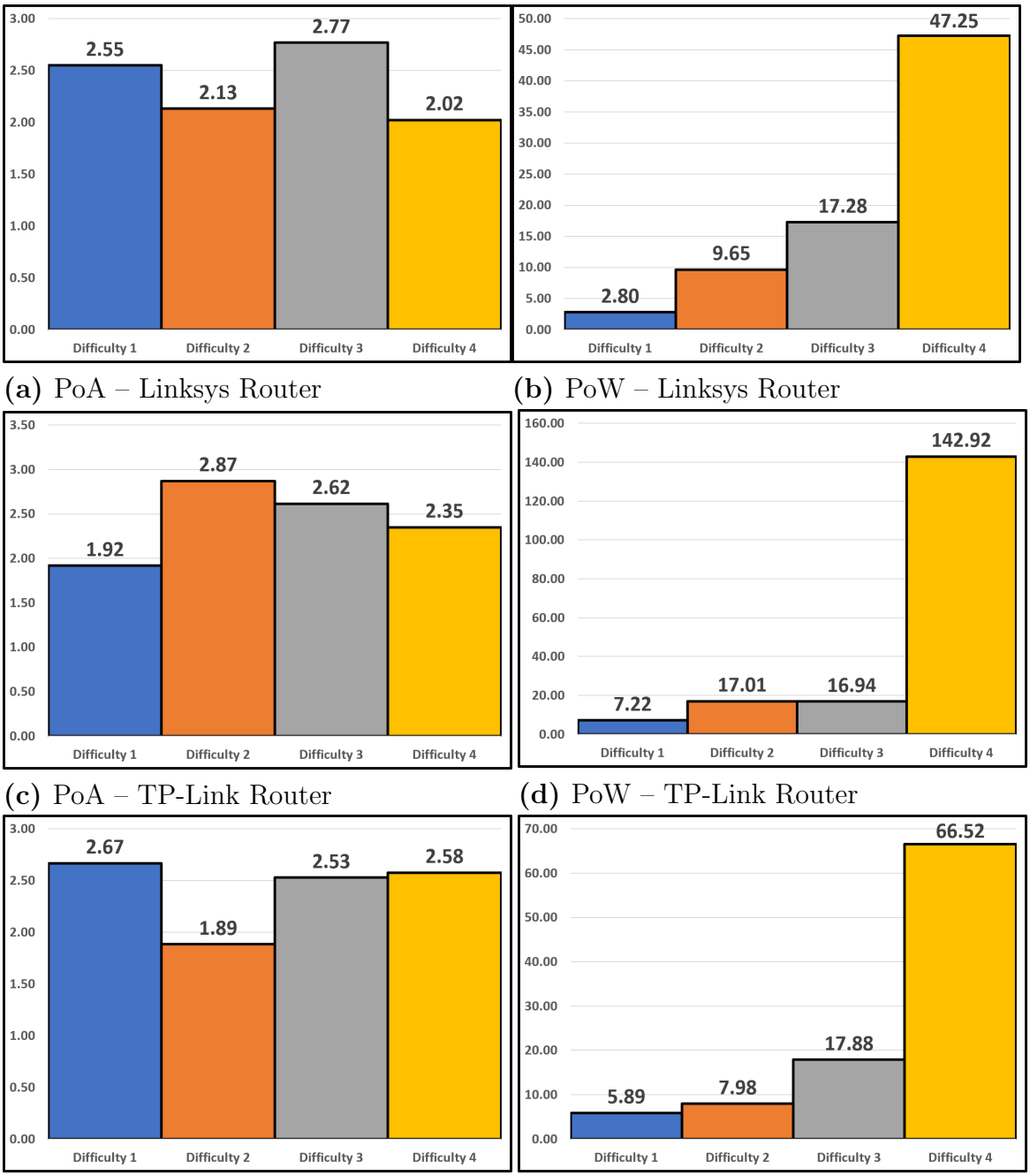


(e) PoA – Verizon Hotspot



(f) PoW – Verizon Hotspot

Figure 5.9 Average τ_{image} in seconds for Proof-of-Authority (PoA) and Proof-of-Work (PoW) tests for different image sizes for 3 different network routers.



(a) PoA – Linksys Router

(b) PoW – Linksys Router

(c) PoA – TP-Link Router

(d) PoW – TP-Link Router

(e) PoA – Verizon Hotspot

(f) PoW – Verizon Hotspot

Figure 5.10 Average τ_{image} in seconds for Proof-of-Authority (PoA) and Proof-of-Work (PoW) tests for different difficulty levels for 3 different network routers.

CHAPTER 6

COMPETITION-BASED ACTIVE LEARNING INSTRUCTION FOR DRONE EDUCATION

6.1 Introduction

Paper: *P. Abichandani, D. Lobo, B. Dimitrijevic, A. Borgaonkar, J. Sodhi, S. Kabrawala, D. Brateris, and M. Kam. " Competition-Based Active Learning Instruction for Drone Education," submitted to Interactive Learning Environments, 2022*

The Association for Unmanned Vehicle Systems International (AUVSI) noted in its seminal economic report that UAVs would be responsible for creating 100,000 jobs by 2025 in the U.S. [146]. The rapid proliferation of drone technology in various applications has led to an increasing need for professionals skilled in sUAV piloting, designing, fabricating, repairing, and programming. Engineering educators have recognized this demand for certified sUAV professionals.

Several studies on curriculum and instructional methodologies to effectively prepare undergraduate students for sUAV careers have been reported [52, 199, 299, 300, 361]. The electro-mechanical integration of motors, sensors, and batteries with an aerodynamic frame in an sUAV, makes these vehicles excellent platforms for multidisciplinary STEM education [69]. Studies reported in literature observed that including robots in STEM classrooms might provide the following key advantages over traditional pedagogy in teaching the theory and applications of STEM: (1) integration of STEM topics in a multidisciplinary domain, (2) efficient transformation of abstract concepts into concrete learning modules for students, (3) combination of STEM theory with its application, (4) hands-on learning that is active and engaging, and (5) a delightful and motivating learning environment [164]. Furthermore, the use

of autonomous robotics in formal and informal learning environments improves math and science learning, as well as critical thinking and problem-solving skills.

Robotics competitions are defined by open-ended problems to engage the students in developing creative solutions. The participants are expected to be self-reliant and are encouraged to work collaboratively with other students to apply their problem-solving methods. The cooperative goals required in competitions make students accountable for their tasks for the sake of their groups [55]. Robotics competitions generally do not have a structured curriculum since the participants are free to develop their own solutions to the problem with the main objective of winning the competition. Competition-Based-Learning (CBL) is a student-centered pedagogical approach based on social and cognitive constructivism [49, 143]. CBL provides a formal setting for the students to achieve the intended learning outcomes (ILOs) while still retaining the creative and open-ended nature of a competition [378]. In the CBL approach, there is an emphasis on the learning journey of the students winning, and the competition is not the primary objective [21]. The students are still required to work in groups to collaboratively solve an open-ended task while completing regular task assignments, which are designed to guide students towards the intended learning objectives of the course [49]. CBL is generally combined with project-based learning (PjBL), problem-based learning (PBL), or cooperative learning (CL) [49]. The use of CBL has been reported in nursing [21], business [82], accounting [151, 275], entrepreneurship [8], mechatronics [117], fashion design [58], social studies [134], software engineering [49], robotics [69], and information technology programs [143].

This study presents a CBL approach, which leverages an informal learning environment such as a Makerspace to engage students and motivate them towards drone-related careers. The approach was tested and validated through a drone design and piloting competition organized at a research institution located in the United

Figure 6.1 Students participating in the drone competition after completing a 14-week workshop sequence that covered drone building, flight simulations, and manual/semi-autonomous piloting.

States. The competition's goal was to win a manual and semi-autonomous piloting challenge while maintaining 100% safety from take-off to landing. Figure 6.1 depicts students actively involved during the drone competition and workshop sessions.

Two key research questions that focused on the CBL approach for drone education guided this study:

- RQ1: What is the student's perceived impact of CBL on their drone building, flight simulation, and piloting skills and knowledge?

- RQ2: What are student perceptions of instruction, Makerspace usage, and challenges associated with a CBL based drone competition?

The CBL pedagogy was encapsulated in a set of indented learning objectives (ILOs), Makerspace-based curricular modules covered during the weekly workshops, and unfettered student access to the Makerspace resources. As depicted in Figure 6.2, the ILOs captured the requirements of modern drone industry professionals via curricular modules i.e., sUAV assembly, testing, and validation using Commercial Off-the-shelf (COTS) components, simulation of drone flight missions, and manual and semi-autonomous drone piloting. Multiple assessment strategies were employed to gain insights into student perceptions about different aspects of CBL.

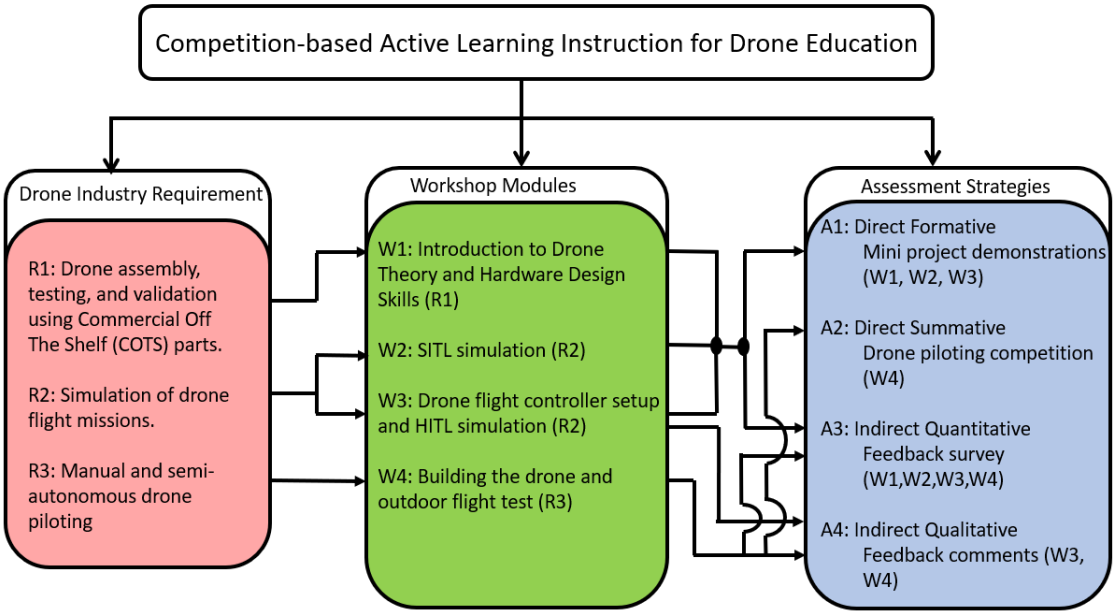


Figure 6.2 Multiple assessment strategies were employed for each of the course modules to obtain quantitative data and qualitative insights about student perceptions.

6.2 Related Work

Educational robotics has seen explosive growth in adoption since the collaboration between LEGO and MIT Media Lab, which developed MINDSTORMS, an educational robotics platform for commercial markets [164]. The authors in [25] provide an in-depth systematic review of the current state of the art in educational robotics. They argue that the benefits of introducing robotics in education have significant potential for positively influencing a students' academic and social skills by creating opportunities to engage in critical thinking and problem-solving through designing, assembling, coding, and operating the robots to perform specific goals. The 147 articles reviewed found that all the studies unanimously suggest that robotics promotes active learning and improves the learning experience. The authors found 16 studies that pointed to increasing participation among students of various backgrounds. Hands-on experience with projects based on robots and drones can help

students visualize complex science, engineering, and technology (SET) concepts and distill them down to real-world application [34, 36, 254, 361].

A constructivist approach to learning, such as competition-based learning (CBL) where the competition is used as a stimulus to maximize the chances of positive learning outcomes, is a promising area of educational research [56]. CBL is generally centered around project-based or problem-based learning while involving the students to work collaboratively on scaled-down versions of real-world problems. Several educational researchers are currently studying the CBL approach's effectiveness in multiple disciplines, including engineering, social studies, nursing, and entrepreneurship [8, 21, 134, 169].

CBL overcomes the current deficiencies associated with traditional learning practices, such as lack of motivation, self-esteem, insufficient practical and real-life experience, and inadequate collaborative exercises [143]. The study in [117] presents a competition-based framework integrated with lab work for mechatronics curriculum to develop skills in core areas necessary to complete a course capstone project successfully. The course was open to undergraduate students pursuing the mechatronics program. At the end of the course, the students were to build a mobile robot based on Arduino. The qualitative feedback gathered from course evaluations suggests that the approach effectively taught the mechatronics curriculum. The CBL approach was further investigated for robotics education for K-12 students in [69]. The robotics competition "Robofest" was an autonomous robotics competition emphasizing math and science problems. The study showed that the students who completed a STEM assessment before the competition achieved higher math and science scores at the end of the round.

CBL, when implemented for nursing education, has been shown to enhance motivation for learning, enforce teamwork spirit among students, promote engagement and self-directed learning, emphasizes knowledge sharing among the group members,

and stimulate students' creativity and innovation to achieve better learning outcomes [21]. In the study [82], the authors argued that the benefits of CBL not only apply to students but also to faculty.

The effectiveness of CBL in overcoming students' learning challenges has also been studied in accounting [151, 275]. In [275], the authors investigated the non-accounting major students' perception of participating in an online accounting quiz competition for the Introduction to Financial Accounting and Reporting course. The questionnaire was specifically developed and distributed to students from the Diploma in Computer Science, Diploma in Public Administration, Diploma in Tourism Management, and Diploma in Office Management and Technology programs. The descriptive statistics results reported mixed feelings about the quiz and its goal in achieving the accounting course's learning objectives. In [151] a specific learning activity was conducted via the preparation of a mind mapping video for the course Advanced Financial Accounting and Reporting on Corporate Social Responsibility topic. The study aimed to examine the perception of using the CBL video as a pedagogical tool in the accounting course. The results from feedback surveys concluded that the winning video proved helpful in learning the accounting course. Moreover, active and independent learning promoted students' self-directed and regulated learning system.

The authors in [8] critically analyzed the impact of entrepreneurial competitions on Omani students' soft skills. They performed a purposive sampling for the study. A sample of 125 students was selected among the 450students from various Higher Education Institutions (HEIs) who participated in the entrepreneurship competition. A questionnaire was distributed two times to the students before and after the competition, and 110 fully completed questionnaires were taken into the research study. The empirical study's findings suggested that competitions positively impacted the students' soft skills and mindset after participating in the competitions. The

study confirmed that competition was an encouraging affair that made the students learn new skills and developments in the business fields. Further, the study confirmed that the competition made it easy to explore the labor market in requirements. The study suggested that higher education institutions should motivate and encourage students to participate in various entrepreneurial competitions, inducing a spirit of entrepreneurship among the young students to enhance their soft skills for self-sufficiency and identify the potentialities vested.

In [134], the authors studied the effectiveness of CBL in meeting the objectives of social studies courses. A peer competition-based mobile learning approach was adopted to conduct the learning activities in the social studies course of an elementary school. The experimental results showed that students' local cultural identity improved along with learning interest and attitude due to CBL.

The above works reveal key insights about the use of CBL such as the use of competitions to promote student motivation, encourage collaboration between the learners, increase likelihood of the learner meeting the intended learning outcomes, and improve learners' self-esteem by providing sufficient real-life practitioner exercises. The variety of studies reported on the use of a CBL-based instruction approach also indicates the versatility of the approach in a wide range of disciplines.

The above body of work provides a glimpse of how CBL can be an exciting alternative to traditional educational instruction methods. However, approaches to successfully integrate a competition with the curriculum's learning objectives are still an active area of research for educational researchers. Specifically, there is a gap in the literature about the application and effectiveness of CBL for drone education. This chapter fills this gap by reporting on the following contributions:

1. Competition-based active learning instructional approaches to teach the building blocks of drone technology careers, i.e., design and fabrication, flight simulations, and commercial drone piloting.

2. A 14-week long curricular structure that encapsulates key learning objectives as demonstrable milestones. These learning objectives were drawn from Electrical and Mechanical Engineering, Computer Science, Control Systems, and aircraft piloting.

3. Multi-methods assessments for students' gain in drone industry skills and knowledge, attitudes towards the CBL approach for drone education, and receiving instruction in an informal environment such as the Makerspace.

6.3 Competition-based Learning for Drone Education

The drone competition was open to undergraduate and graduate students from all engineering backgrounds. A total of 97 students from various backgrounds, including engineering, computer science, arts, and sciences departments, participated in the competition. The students attended weekly workshops as teams of 3-5 members, totaling 22 teams for 14 weeks after which they competed in manual and semi-autonomous piloting challenges while maintaining 100% safety from take-off to landing. Each weekly workshop was approximately 4 hours long. The workshop sessions were conducted at the university Makerspace to accommodate the students and provide easy access to tools and resources crucial to building the drone. The students participated in the workshop sessions as groups and were expected to

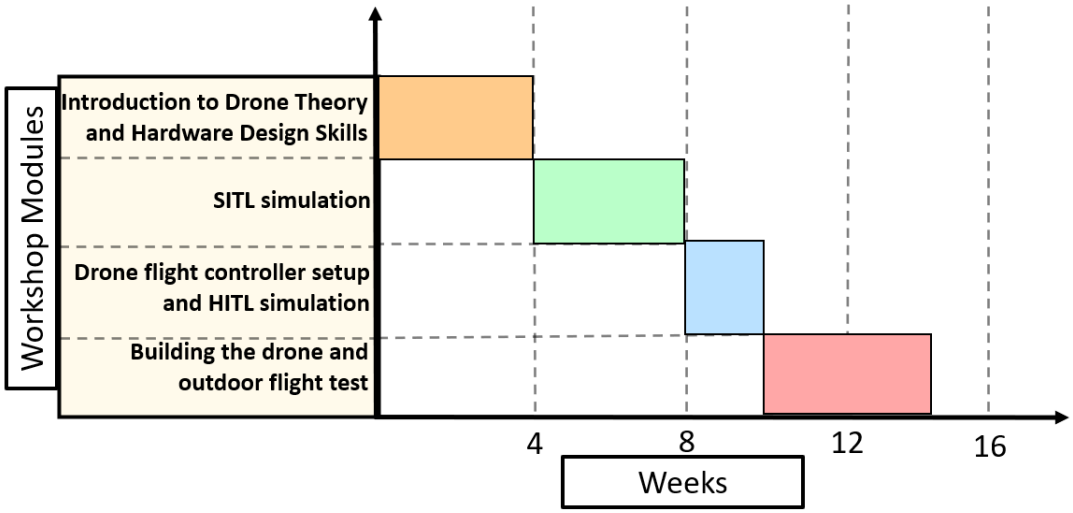


Figure 6.3 Duration for each of the curricular module covered during the workshops.

work collaboratively on designing, building, programming, testing, and validating a quadrotor drone. Figure 6.3 depicts the duration of each of the curricular modules covered in the workshop.

The curricular activities were developed around the *Interactive-Constructive-Active-Passive* (ICAP) modes of cognitive engagement [63]. The ICAP framework enables educators to design and examine active learning approaches due to the observable and assessable nature of the four student engagement modes it outlines. Table 6.1 tabulates the student engagement behaviors observed during the workshops in the context of the ICAP framework. Four workshop modules were developed as part of the CBL approach to cover specific drone industry requirements depicted in Figure 6.2. With each workshop came a set of specific milestones that students were to achieve. Demonstrating these milestones formed the basis of direct formative assessment of gain in students' knowledge and skills. All workshop content was hosted on a Learning Management System (LMS).

Table 6.1 Student Engagement Modes Observed in the Context of the ICAP Framework during the Competition Workshops [63]

#	Workshop Module	Interactive	Constructive	Active	Passive
1.	Introduction to drone theory and hardware design skills	Interact with instructional staff during the lecture, collaborate with team members	Creating 3D printed drone parts, Using the Soldering station	Work with Makerspace tools, using CAD software, and soldering	-
2.	SITL simulation	Discussions on drone simulation software	Creating drone programs based on instructions during workshop session	Installing drone simulation software, performing statistical analysis on telemetry data	-
3.	Drone flight controller setup and HITL simulation	Discussion on FC setup, collaborate with team members to setup the FC	Creating drone programs based on instructions during workshop session	Setting up FC/Radio controller, and performing HITL simulation	-
4.	Building the drone and outdoor flight test	Collaborate with team members on assembly, testing, and validation	Building the drone kit, perform outdoor flight test, design drone landing gear	Researching online resources such as YouTube videos and Online Blogs,	-

6.3.1 Module 1: Introduction to Drone Theory and Hardware Design Skills

The first module of the workshop spanned weeks 1 - 4. The four (4) hour workshop sessions were divided into two parts. The first 2 hours consisted of an interactive lecture session discussing the current state of the drone industry, drone kinematics, drone dynamics, drone navigation and control, sensor data analysis, and drone flight simulators. These interactive lecture sessions consisted of the instructors

demonstrating drone technology using drone subsystems, live flights, and videos while positing drone related questions to engage students in technical conversations as a class. The next part of the workshop session consisted of hands-on activities where the students were introduced to the Makerspace tools, emphasizing using 3D printers and soldering stations.

These interactive lecture sessions consisted of the instructors demonstrating drone technology using drone subsystems, live flights, and videos while positing drone related questions to engage students in technical conversations as a class. The next part of the workshop session consisted of hands-on activities where the students were introduced to the Makerspace tools, emphasizing using 3D printers and soldering stations. Students were guided through the steps of using computer-aided design (CAD) software such as TinkerCAD and Solidworks to develop 3D models of drone parts [326, 352]. The students were free to ask questions to the instructor during the demonstrations of the tools and software. The student teams discussed with their members any solutions to address issues encountered while setting up the CAD software, and brainstorm designs for their drone parts. After developing their designs, the student demonstrated their designs and discussed the intricacies of 3D printing parts for their drone platforms. Students were subsequently immersed in the hands-on experience of soldering drone circuits. Soldering is a crucial skill in building a drone from scratch. Each student team was responsible for soldering the circuit board on their drone.

Key milestones for this module included CAD design and 3D printing different parts of the drone body such as drone frames, landing gear, and mounts for different drone parts, and soldering electronic components of drone circuits such as electronic speed controllers (ESCs) and power management boards.

6.3.2 Module 2: Software in the Loop (SITL) Simulation

During weeks 5 - 8, students were introduced to open-source simulation environments for drone operation. Software simulators provide a safe and reliable way to verify flight plans and motion planning algorithms. Students were introduced to the JMAVSIM and Qground Control (QGC) simulation environment to perform Software-in-the-loop (SITL) simulations [148, 272]. In the SITL phase, the students were first tasked with setting up the drone simulation environment and different flight missions for the drone. The students were introduced to industry-standard practices of setting up drone flight missions, such as setting geo-fences, waypoints, flight modes, including return-to-launch, return-to-home, and loiter. The instructor provided live demonstrations of the simulation environment, and students worked with the instructor and each other regarding the software setup at any time during the demonstration. Figure 6.4 depicts a snapshot of the drone SITL in Qground Control. After the students were comfortable with the drone simulation environment, the next task involved creating autonomous flight missions using scripts. The students were provided a collection of advanced and straightforward flight patterns that they programmed for SITL tests. Programming APIs were provided to the students with appropriate documentation and tutorials. The tutorials covered setting up

Figure 6.4 (Left) Snapshot of JMAVSim flight simulator for the quadrotor. (Right) SITL software tools Qground control, and PX4 autopilot.

the software on different computers, working around any installation challenges, and performing GPS-based missions along specific flight paths. The programming exercises were designed as group activities for students to interact with each other in teams and develop programs for each of the drone flight patterns. Students worked on different flight patterns, including simple shapes such as squares, circles, triangles, and advanced shapes used in the drone industry.

Key milestones for this module included using the JMAVSIM and QGC environments to set manual mission planning using GPS coordinates obtained from a Google Map API, logging telemetry data collected from the flight mission with time-stamps in a CSV format, and perform statistical analysis visualization of the logged data.

6.3.3 Module 3: Drone Flight Controller Setup and Hardware in the Loop (HITL) simulation

From weeks 9 - 10, students graduated to using popular and state-of-the-art open-source drone flight controllers (FCs) to recreate the drone simulations developed in the SITL phase with the FC. This process is also called Hardware-in-the-Loop (HITL) simulation. The knowledge gleaned from the SITL exercises was directly applicable here as flight missions could be tested with the FC. HITL simulations provide closer to real-flight test results since the drone telemetry provided by the FC serves as a close approximation for outdoor flight test results. The next task involved setting up the joystick radio controllers to send flight commands to the drone. The HITL simulations were compatible with the radio controllers, and the students were tasked with operating the drone in simulation using input commands from the joysticks and switches of the radio controller. The HITL simulations also provided the students with a safe and first-hand experience piloting a drone using the joystick (for throttle, roll, pitch, and yaw control) and switches (arm/disarm,

Figure 6.5 (Left) Drone mounted with the electromechanical hardware and battery assembled by a student team by the end of the workshop sessions. (Top-Right) Popular flight controller hardware. (Bottom-right) radio transmitter used to communicate with the drone.

among other functions). The students discussed with their team members and collaboratively worked towards setting up the flight controller, and radio transmitter. The instructional staff worked with student groups one-on-one to debug any technical issues.

Key milestones for this module required the students to perform sensor calibration processes for the FC, radio transmitter, and the electronic speed controllers (ESCs), performing HITL flight missions for their drone. Fig. 6.5 depicts the 450 quadrotor frame with the flight controller mounted and alongside the radio transmitter.

6.3.4 Module 4: Building the Drone and Outdoor Flight Test

From weeks 11 - 14, students were tasked with building a quadrotor drone using the provided kit. The quadrotor frame was a standard 450 model [99]. The students were provided an outdoor arena designated for flight tests, enclosed within nets to provide a safe space for flying the quadrotor. The student teams collaboratively worked towards assembling the drone kit, flight testing the software, and validation.

All students were mandated to use a 45-foot rope, which was to be always tethered to the drone during the flight test to prevent accidental flyaway situations. The teams managed their workload where one member managed the safety rope, one member operated the radio transmitter to fly the drone, while other members observed and documented the flight test.

Specific milestones for this module included successfully and correctly assembling the drone body's electro-mechanical components, re-calibrating the sensors and ESCs, and conducting a safe outdoor flight test under the supervision of the instructional staff.

6.3.5 Drone Competition

The drone competition was held in week 15 of the term, summatively evaluating the students' understanding of the fundamentals of drones. The student teams, which successfully reached all the technical milestones for each workshop module, qualified to participate in the final competition. Conditioning student team participation on their ability to achieve the workshop milestones successfully ensured that the students participating in the competition had a flying drone that was safe and thoroughly tested. In the week leading up to the competition, the student teams could practice their drone piloting skills in the flying arena. Student teams that had trouble getting their quadrotor to fly were provided with hands-on guidance by the instructors. The competition was held in a 55 ft. (L) x 14 ft. (W) x 12 ft. (H) arena enclosed with a protective net. The competition featured two rounds: 1. Round 1: Obstacle course flying. 2. Round 2: Payload carrying and delivery.

1. Round 1: Students competed to pilot their quadrotor through an obstacle course in minimum time in the first round. Students were allowed a total of 3 tries to fly around the obstacle course. Negative points (time penalties) were applied for each flight if there were collisions with the obstacle or the course was not completed. The highest score from the three trials (navigation (flight) time + negative penalties) was used to rank each team's performance.

2. Round 2: The top 10 ranked teams from Round 1 progressed to Round 2. This round featured a payload delivery task, where the payload was an egg. Each team placed an egg in a 3D printed carrying cradle provided by the instructional staff. As seen in Figure 6.1, the cradle had to be suspended at least 12 inches below the drone, measured from the point of the frame that touches the ground to the egg's location. The teams were to fly the drones through the obstacle course while carrying an egg. In order to successfully complete Round 2, each team was required to safely deliver the egg back to the take-off/landing location. The requirements and the penalty system for this round were the same as in Round 1. The teams were scored based on the delivery time and navigating the obstacle course without damaging or dropping the egg outside the delivery location.

The drone competition featured cash prizes for the winning teams. The first prize-winning team got $1000, the second prize-winning team got $750, and the third prize-winning team got $500.

6.4 Assessment Techniques

A total of 97 students registered for the workshops and the competition. As mentioned, the competition was open to students from the undergraduate and graduate programs. 22% of the participants were MS students, 9.5% of the participants were Ph.D. candidates, and 68.5% of the participants were undergraduate students. The competition attracted students from various backgrounds: 20% of the registered participants were Computer Science majors, 74% of the students were from the School of Engineering, and 6% of the students were from the School of Life Sciences and Liberal Arts.

At the beginning of the competition, a brief survey was conducted to understand students' prior experience with drones. Table. 6.2 depicts the students' responses to the survey questions. The responses suggest that 61% of the students had no prior experience with drones, 22% of the students were excited at the prospect of building a drone for the first time, and 17% had prior experience flying a drone. None of the

Table 6.2 Student Responses to the Pre-survey Indicates that a Majority of them had no Prior Drone Building or Piloting Experience

The competition announcement got me interested in drones for the first time.	22%
I was interested, but had no hands-on experience with drones	61%
I had some experience flying drones for fun, but had not built one myself.	17%
I assembled drones using commercially available drone kits and accessories/parts.	0%
The prizes offered were the primary motivation for me to participate in the competition	18% Yes 82% No

students had prior experience building drones using commercial-off-the-shelf drone kits.

A case study approach involving multiple assessment methods was adopted to assess students' drone assembly skills, flight simulation knowledge, piloting acumen, perceived gains in skills and knowledge, and attitudes towards a CBL approach for drone education. Both quantitative and qualitative assessments were conducted to gain multiple insights into the effectiveness of CBL. Table 6.3 presents a summary of the assessment types, methods, targeted course component, and data analysis method.

Table 6.3 Salient Features of the Assessment Strategies

Assessment type used for Research Question (RQ)	Assessment method (Data Sources)	Targeted course component	Data analysis
Direct formative (RQ1)	Workshop milestones	Evaluated student progress in specific milestones 1) Introduction to drone theory, 2) Soldering, 3) 3D printing, 4) SITL simulation, 5) Flight controller setup, 6) HITL simulation, 7) Drone assembly, and 8) Outdoor flight test	Descriptive statistical analysis of task correctness, documentation, and innovation data
Direct summative (RQ1)	Multiple runs of safe flight demonstration	Evaluation of students' safe piloting skills and comprehensive drone systems knowledge	Flight performance data from multiple runs
Indirect quantitative (RQ1, RQ2)	Likert-scale feedback survey	Prior drone building experience, value added by the competition, perceived gain of drone-related skills, value of Makerspace environment	Descriptive statistical analysis for Likert-scale responses
Indirect qualitative (RQ1, RQ2)	Descriptive comments from feedback survey	Overall experience with CBL and drone education in Makerspace	Inductive analysis using the constant comparative method

6.4.1 Direct Formative Assessment

Formative assessments were conducted at the end of each workshop session to assess whether students reached specific milestones. As described in detail in Section 6.3, these milestones included the following topics: 1) Introduction to drone theory, 2) Soldering, 3) 3D printing, 4) SITL simulation, 5) Flight controller setup, 6) HITL simulation, 7) Drone assembly, and 8) Outdoor flight test. These assessments also ensured that the students received appropriate help if they had trouble comprehending the relevant concepts covered during the workshops.

Student progress was assessed on the following dimensions:

1. Milestone correctness: The student team reached the milestones by performing correct technical tasks that met all specifications.
2. Documentation: Each milestone required the students to provide meaningful documentation of the tasks involved. Video demonstrations that clearly showed the task completion according to the specifications were an essential part of the documentation.
3. Innovation: Task solutions demonstrated their creativity and ability to synthesize new approaches from the information provided during the workshops.

Table 6.4 depicts the weekly breakdown (in percent) of students reaching their milestones. It is observed that a majority of the students reached the first three milestones within two weeks. Starting the fourth milestone (SITL simulation), students took up to 3 weeks to successfully reach the milestones. In contrast to only 18.18% of students completing the SITL simulation within one week, 77.27% students finished HITL within a one-week period. This observation indicates that students gained sufficient knowledge and skills about flight simulations from their SITL experience to quickly and complete HITL simulations. As expected, drone assembly was the longest milestone (4 weeks). By the end of week 14, which was the last workshop session, 95.45% of the students completed the outdoor flight test. The remaining students who did not achieve their milestones by the last 2 weeks met with the instructional staff during office hours for additional assistance. The table also

Table 6.4 Percentage of Students Reaching the Workshop Milestones on a Weekly Basis

Week	Introduction to Drone Theory	Soldering	3D printing	SITL sim	FC setup	HITL	Drone Assembly	Outdoor flight test
1	100%							
2		77.27%						
3		22.73%	63.64%					
4			36.36%	18.18%				
5				54.55%				
6				27.27%	36.36%			
7					36.36%			
8					27.27%	77.27%		
9						18.18%		
10						4.55%	18.18%	
11							36.36%	
12							31.82%	27.27%
13							13.64%	50%
14								18.18%

highlights an important part of the CBL philosophy that every student has their own pace of learning and takes their time to find the solution to the task [149].

6.4.2 Direct Summative Assessment

The summative assessment featured a comprehensive demonstration of all functions of the drone via a series of safe and controlled outdoor flight tests in a week 14. The student teams were required to fly their drone using their radio controller for at least 1 minute for a total of 5 flight runs. This assessment indicated that the student groups successfully reached all the milestones of the workshop sessions and served as a qualifier for the final competition. Among the 22 participating teams only 1 team was unsuccessful in qualifying to participate in the final competition.

6.4.3 Indirect Quantitative Assessment: Likert-scale Feedback Survey

The students were requested to complete an online post-competition feedback survey (delivered through the LMS) that featured thematic questions to assess the following three core aspects:

1. Value added by the drone building competition.
2. Value of integrating the Makerspace environment during the drone building process.
3. Overall experience with the workshop sessions and the drone competition.

97 students completed the survey.

Value added by the drone building competition: This section of the survey featured a total of 3 questions. Two (2) questions sought to elicit responses from students about their perceptions of the value added by the competition and workshop to their technical skills and knowledge and their creativity and ability to innovate. The third question was inspired by Fred Reichheld's net promoter score (NPS) question wherein the students were asked to respond to the statement "I will recommend this

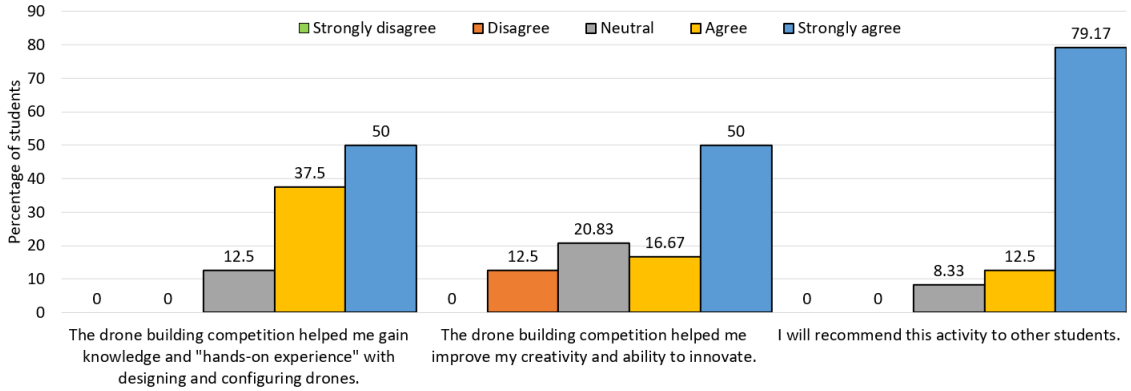


Figure 6.6 Likert Scale based student responses to questions about the drone competition.

activity to other students" [282,283]. The responses for this section of the survey used a Likert Scale of 1 through 5 where a response of 1 indicated strong disagreement and a response of 5 indicated strongly in agreement.

Figure 6.6 depicts student responses to 3 statements in this category on the feedback survey. As depicted in the graphs, 50% of the respondents strongly agreed and 37.5% of the respondents agreed to the statement that the drone competition helped them gain hands-on experience with designing and configuring drones. A total of 66.67% of the respondents strongly agreed or agreed to the statement that the drone competition helped improve their creativity and ability to innovate, while only 12.5% of the respondents disagreed with the statement. These responses suggest that the students had a favorable perception of the drone competition in terms of gaining hands-on experience in designing and building drones while enhancing their creativity and innovative ability.

A total of 79.17% of the respondents strongly agreed to recommend the competition and workshop to other students, indicating that most of the students rated their workshop and competition experience high enough to recommend it to other students.

Value of the Makerspace as a venue for the workshops: This section of the survey used two statements to elicit responses from students about their perceptions of the value added by Makerspace as the venue for the workshops. The responses for this section of the survey used the Likert Scale of 1 through 5, where a response of 1 indicated strong disagreement and a response of 5 indicated strongly in agreement.

Fig. 6.7 depicts the Likert-style responses of the students to the questions regarding their experience at the Makerspace. The results show that 58.33% of the respondents either agreed or strongly agreed with the statement, "Because of the exposure to Makerspace during the drone workshops, I have used the Makerspace for other projects." In addition, 83.34% of the respondents agreed or strongly agreed with the statement, "Because of the exposure to Makerspace during the workshop, I intend on using it for my future class projects and research." These results suggest a positive learning experience at the Makerspace for a majority of the students. As a corollary, the positive experience of using the Makerspace motivated students to leverage it during their future education.

Overall experience with the workshop sessions and the drone competition: Fig. 6.8 depicts the responses of the students regarding their overall experience at the end of the drone workshops and competition. 50% of the respondents were

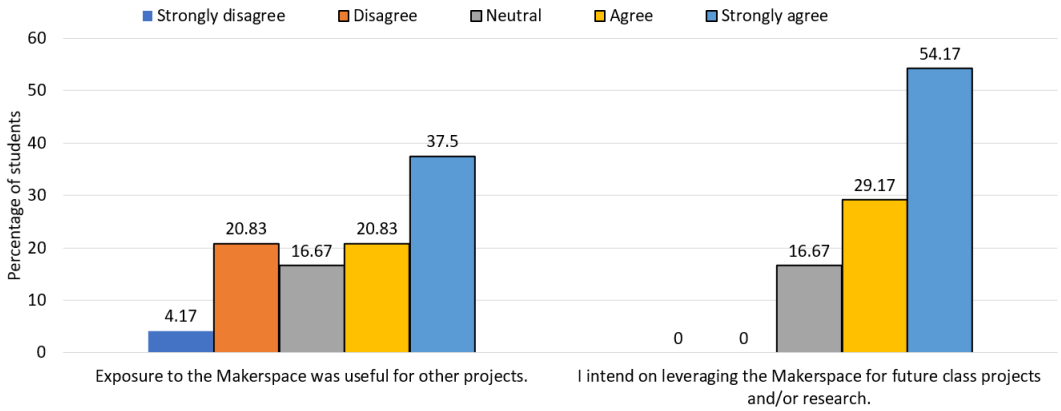


Figure 6.7 Student responses regarding the Makerspace use suggests a positive experience and indicates their intention to use the Makerspace in the future.

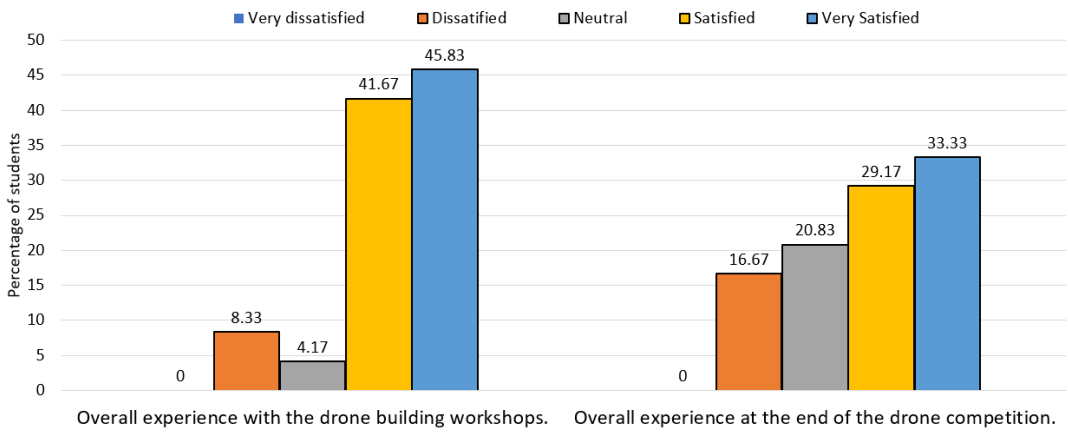


Figure 6.8 Responses regarding the students overall experience with the drone workshop and final competition.

very satisfied with the drone workshops, and 41% were satisfied, suggesting that the workshops proved to be a positive learning experience for students. Only 9% of the respondents were dissatisfied with their workshop experience. The final competition 36% of the respondents were very satisfied, 27% were satisfied, 18% were neutral, and 18% were dissatisfied. These responses are explored further in the indirect qualitative assessment section 6.4.4.

6.4.4 Indirect Qualitative Assessment

At the end of the drone competition, the post-completion feedback survey featured open-ended questions for the students to provide brief descriptive feedback about their experiences. The comments collected at the end of the survey formed the primary data source to unpack insights about the effectiveness and challenges of the CBL approach to drone education.

The prompts in the survey asked the students to provide a brief description of their experience with the drone building workshop and the drone competition. The prompt also asked the students to highlight their positive and negative experiences and provide suggestions for improvement. The feedback responses received were analyzed inductively according to the constant comparative method [302]. Based

on the analysis, three recurring themes were identified. These themes are presented here as follows:

1. **Theme 1 - Problem Solving Skills and Collaborative Learning**
   The drone building workshop consisted of challenging activities. These activities encouraged the students to think of innovative solutions to the given problems, as noted in the following comment *"I love the part each team has their way of approaching the final competition. I saw a lot of different method of solving the same problem."* The students participated in the workshop sessions and the competition as a team, which provided a collaborative learning opportunity since the students had to work together to solve the problems as indicated in the following comment, *"The competition provided an engaging opportunity to connect with engineers from other backgrounds and work together towards solving the drone design challenges."*

2. **Theme 2 - Interest in Drone Technology**
   As noted in Table 6.2, majority of the students participating in the competition had no prior experience with drones. The drone building workshop sessions and competition provided an opportunity to successfully generate interest in drone technology. The following comment echoes this sentiment, *"The drone building workshop was a great experience to learn about the step-by-step procedure of building drones. I plan on using the experiences gathered and build my own drone."* This comment suggests that the students were motivated by the competition along with the comprehensive instruction and learning resources. The students were appreciative of the learning resources and instructors as suggested by the following comment: *"The workshops were very informative, and the instructors/TA were always present and helpful."* The competition environment served as a motivator for students to effectively grasp multiple engineering skills as suggested by the following comment: *"The competition was an excellent way to help students in all aspects of engineering... working with a team, strategizing the approach to the course, deciding the modifications to drone which will be most beneficial for the competition, working hands on with the actual drone from step 1 to final competition, soldering skills, and troubleshooting problems."*

3. **Theme 3 - Challenges and Future Directions**
   The drone competition was open to students of all STEM backgrounds at the university. The emphasis on the competition proved to be an effective motivator for students to learn about drones. However, there were some challenges faced by the participants, which were highlighted in the feedback comments. One of the significant challenges of conducting a drone competition for most first-time drone pilots is providing a safe and supervised space for conducting flight tests. One of the students commented, *"For future iterations of the competition, increased time should be provided for piloting the drones. An extra week or two with the safety nets would drastically help teams piloting skills improve."* This comment suggests a fixed period may be allotted to piloting the drones since drone piloting formed a crucial component of the final competition challenge. One of the students commented, *"One improvement I would like to see is it*

become more challenging. I would like for future competitions to deal with more programming and use of mechanical components such as a grapple system." This comment provides valuable suggestions for future iterations of the competition, emphasizing the programming and designing aspects around the drone platform.

6.4.5 Drone Competition Results

The final competition evaluated the students' piloting skills and the design innovation for the payload delivery task. The scores were assigned based on the total time taken by each team to complete the drone flight mission. The scores include the total flight time in minutes and the penalties received in minutes. The mean of the scores received by the students for round 1 and round 2 of the competition was 1.56 minutes and 1.9 minutes, respectively. The standard deviation observed for the scores for round 1 and round 2 was 0.37 minutes and 0.43 minutes respectively.

6.5 Discussion

The use of multiple methods for assessment was instrumental in helping us explore the answers to the research questions. Quantitative assessment methods used provided evidence about the effectiveness of the CBL approach for drone education. In contrast, qualitative assessment methods helped generate a descriptive understanding of student perceptions, which lent further evidence to the quantitative data [188].

CBL for Drones The quantitative insights obtained suggest that the students received the CBL approach well and were motivated to learning more about drones. Many of the participating students were introduced to drones for the first time and found the competition to be a fun approach to learning more about the field. This observation suggests that the drone competition had a positive effect on the students' motivation for learning new concepts [268]. The qualitative assessment of the feedback comments indicates that the students were encouraged to work on their problem-solving skills and learn collaboratively. The competition motivated the students to

work together to solve the assigned tasks, which is in line with the observation that competitions promote problem-solving skills and collaborative learning [53]. The constructive feedback comments provided by students suggest that they took on the roles of stakeholders to improve the future CBL iterations of this competition.

Makerspace and Student Engagement : The education community has recognized the potential for Makerspaces as learning environments that can stimulate interdisciplinary collaboration and self-directed learning [135]. The current state of the art points to Makerspaces' efficacy in promoting creative thinking and problem-solving skills [182]. The quantitative results for student perception of the Makerspace-centric learning environment suggest that the students found this non-traditional learning environment useful for critical engineering skills and intend to leverage the space for future projects. The qualitative results further suggest that the Makerspace aided in the communication skills of the students through interactions with the Makerspace personnel and provided ample opportunities for hands-on and technical experience with the available instruments [346].

Logistical and Management Challenges : We note that when organizing a competition of such scale, there are several logistical and planning challenges. Ensuring that students keep up with their milestones requires continuous communication between the instructional staff and student team members. Additionally, keeping track of the hardware components of the drone requires inventory management on the part of the instructional staff. During the competition, keeping track of flight runs for each team needs multiple staff members synchronizing closely. Ensuring that the teams were lined up for a smooth transition for flights one after the other requires close coordination across all teams.

6.6 Conclusion

There is a growing number of engineering educators trying to prepare students with skills and knowledge relevant to the drone industry. This chapter reports on a competition-based active instructional approach to preparing undergraduate students for engineering, software, and piloting careers in the drone industry. The approach consisted of 14 workshop sessions that incrementally prepared the students for several aspects of drone building, flight simulations, and piloting. The workshop milestones prepared 97 students from STEM backgrounds at a research university to participate in a drone piloting competition. Multiple assessment methods were used to collect quantitative data and qualitative evidence about the effectiveness of the CBL in the context of drone career preparation. Quantitative assessments indicated that most students successfully reached all technical milestones at the end of the workshop sessions. Qualitative assessments revealed that students found the CBL approach to be an engaging method of introducing drone technology. Challenges encountered during this study were discussed and future improvements were discussed. As educators continue to explore active-learning instructional approaches for drone education, we anticipate that the information provided in this chapter may serve as a guide for future drone educators looking to implement CBL.

CHAPTER 7

PREPARING STUDENTS FOR DRONE CAREERS USING ACTIVE LEARNING INSTRUCTION

7.1 Introduction

Paper: *D. Lobo, D. Patel, J. Morainvile, P. Shekhar and P. Abichandani. "Preparing Students for Drone Careers Using Active Learning Instruction," in IEEE Access, vol. 9, pp. 126216-126230, 2021*

The growth in the accessibility of drones has raised societal concerns because drone pilots without the proper training pose potential threats to safety and privacy [335]. Federal agencies around the world have recognized the need to classify drones and develop legislation to govern the safe operation of such vehicles [237, 377]. For example, the Federal Aviation Administration Part 107 Commercial sUAV pilot test in the United States [100], EU Regulations 2019/947 and 2019/945 [98], Transport Canada Basic Exam [114], Nigerian Civil Aviation Authority [240], and Indian Directorate General of Aviation [86]. These legislations mandate that commercial drone pilots are certified through tests and demonstrate their comprehension of the relevant aspects of safe and lawful drone operation.

Modern drone careers involve drone software programming, designing and fabricating drones using Computer-Aided Design (CAD) and rapid prototyping, safe piloting, and knowing legal requirements to operate drones commercially. Engineering and computer science (C.S.) educators have recognized this demand for certified sUAV professionals. Several studies pertaining to curriculum and instructional methodologies to effectively prepare undergraduate students for sUAV careers have been reported [38, 52, 144, 154, 212, 299, 300, 311, 361]. While these studies focus on the construction, programming, and piloting of drones, they do not report on formal instruction on drone operation's safe and legal practices. The United States Federal

Aviation Administration (FAA) introduced the Part 107 remote drone pilot license test on June 21, 2016. This licensure is mandatory for individuals that intend to become commercial sUAV pilots. As part of the Part 107 framework, the FAA prescribes guidelines for the safe and legal operation of sUAVs under the weight of 55 lbs for commercial purposes. Any individual above the age of 16 can prepare for the test using the references provided by the FAA and become a licensed drone pilot operator.

With this growing need to train engineering graduates in upcoming technologies (drones), it is imperative to develop and assess educational interventions that engage students in the content and provides them hands-on experience in the process. Drones can be an exciting hands-on platform to expose students to the principles of electrical and computer engineering, mechanical engineering, and computer science. This chapter reports an active learning approach to prepare undergraduate students for engineering, software, and piloting careers in the drone industry. The active learning approaches are integrated in an undergraduate course that focuses on the building blocks of drone careers and culminates in a capstone drone project. Figure 7.1 depicts some of the capstone projects developed by students during the course.

Figure 7.1 (Left) Actual drones fabricated by students for their capstone projects (Right) CAD designs and Computational Fluid Dynamics (CFD) simulations for drone projects developed by the students in the class.

Active learning is an effective pedagogical paradigm for high-quality, collaborative, engaging, and motivating education [230,269,318]. In active learning, students are engaged in the learning process rather than being passive recipients of knowledge. The primary role of instructors is to provide learning space and facilitate the learning process [138]. Active learning approaches have shown to be effective strategies in engineering education [59, 107, 152, 349].

Two key research questions that focused on the use of active learning approaches for drone education guided this study:

- RQ1: What is the student perceived impact of an active learning-based drone course on their computer programming skills and knowledge regarding the safe and legal operation of drones?

- RQ2: What are student perceptions of instruction and challenges associated with an active learning drone course?

As depicted in Figure 7.2, drone industry requirements were integrated in active learning based curricular modules. Multiple assessment strategies were employed to gain insights into student perceptions about different aspects of the course.

7.2 Related Works

The use of robots for education has witnessed significant growth since the collaboration between LEGO and MIT Media Lab that resulted in the commercialization of the MINDSTORMS robotics platform [164, 208]. The authors in [25] provide an in-depth systematic review of state-of-the-art educational robotics. They posit that the benefits of introducing robotics in education have the potential for positively influencing a students' academic and social skills. Robotics in education creates opportunities to engage in critical thinking and problem solving through designing, assembling, coding, and operating the robots to perform specific goals. From the 147 articles reviewed, the authors found that all the studies unanimously suggest the inclusion of robotics promotes active learning and helps improve the learning

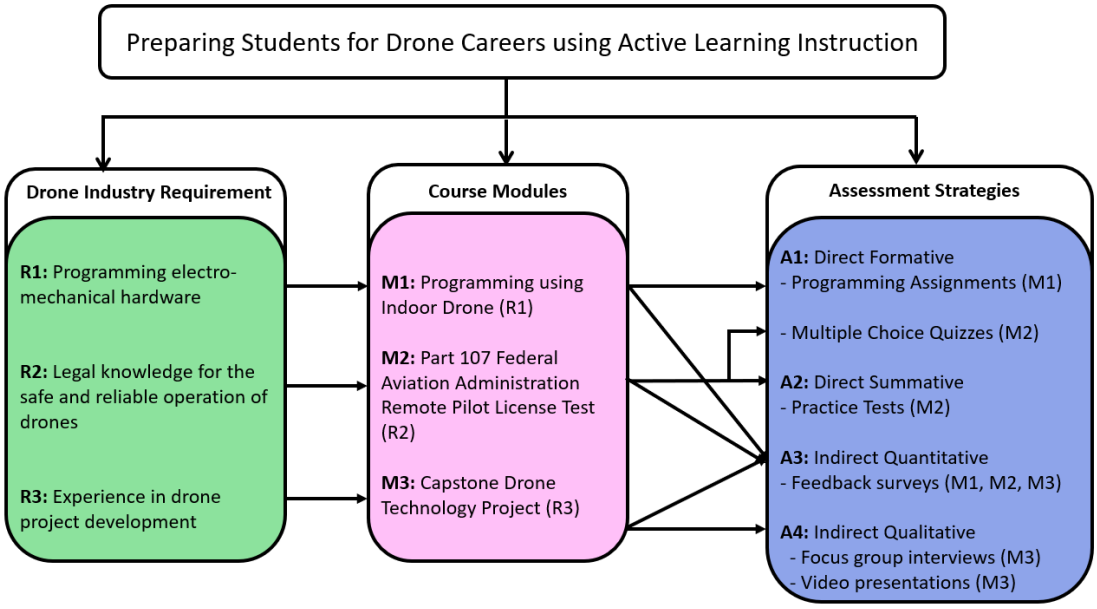


Figure 7.2 A key aspect of this study was the development of active learning based course modules that integrated drone industry requirements. Multiple assessment strategies were employed for each of the course modules to obtain quantitative data and qualitative insights about student perceptions.

experience. The authors found 16 studies that pointed to the effectiveness of increasing participation among students of various backgrounds.

While most educational robotics platforms continue to have the form of ground robots, the recent growth in sUAV technology makes these flying robots a relevant robotics platform to train young engineers looking to join the workforce [81,196,365]. In [361], the authors provide a mapping review of the online courses available for drone education. The study found that of the available courses, only 25% of the massively open online courses (MOOCs) and e-learning courses are related to UAV pilot certification and pilot operation. The authors noted that 56.3% of the online courses are usually a repurposed online version of the courses provided as part of in-person university programs. The length of the courses falls between 4 to 8 weeks with an estimated weekly effort of 4 and 5 hours. Course evaluation methods are limited to theoretical methods such as multiple-choice questions (MCQs) and algebra exercises with numerical answers. The authors concluded that a practical (hands-on)

approach to evaluation needs to be introduced to increase learner engagement and stimulate self-learning and discovery among students.

Engineering schools have recognized the need to prepare professionals entering the job market with experience in the operation of drones, leading to the development of a curriculum focused on this niche [52, 103, 213, 219, 245, 300].

In [300], three courses carrying three credits each related to UAV education were developed, designed, and offered at Southern New Hampshire University with the motivation of integrating UAV education as part of the aeronautical/aerospace engineering program. A UAV lab was also established, equipped with nets for safety and eight cameras for navigation. The courses included weekly lectures, take-home assignments, lab experiments, design projects, a midterm test, and a comprehensive test.

The authors noted that the root cause of the lack of convergence between UAV education and practical application is the absence of experiential learning. The authors focused on team-based learning, project-based education, lab experiments, UAV flight, flight safety, flight tests, UAV laboratory, and hands-on experience. The authors noted that UAV experience requires students to develop skills such as flight planning and flight test execution, and it reinforces skills in data analysis, communication, and teamwork.

In [52], the authors presented an open-access drone programming course for any teacher/student to learn drone programming. The course emphasized programming a drone through practical applications. The course material was developed using robot operating system (ROS) middleware, the 3D simulator Gazebo, and Python programming language. The course has been successfully taught for five years to students from several university engineering degrees. A survey conducted with 21 students during the 2018/19 run of the course showed that 57.9% of the students found the course experience *excellent*, and 42.9% students were *more than satisfied*

with the experience. sUAVs have been used to cover topics on modeling and control for undergraduate and graduate courses as noted in [213], and helping students gain experience by building platforms such as hexacopters [219].

As noted earlier, active learning methods put emphasis on student learning and gives them greater motivation in the process of learning [38, 41, 80, 107, 312, 317]. In [225], the authors described a Project-based Learning (PBL) program for UAV-based remote sensing. The course was designed to help students determine the vigor, temperature, and water stress of crops by using RGB, multispectral, and thermographic sensors onboard a UAV platform. The authors note that students are motivated from the beginning of the learning process to actively participate in UAV flights and subsequent processing and analysis of the registered images. Student satisfaction surveys regarding the effectiveness of PBL conducted over four years have reported consistent positive feedback from the students.

7.3 Main Contributions

While the above body of work points to a growing interest in using drones in undergraduate classrooms, there is still a gap in the literature about preparing students effectively for drone careers. Additionally, the studies related to drone education in schools have not reported on a comprehensive set of assessment strategies that leverage quantitative and qualitative data to provide valuable information on the effectiveness of the curriculum and associated pedagogical approach. This chapter fills this gap by reporting on the following contributions:

1. Active learning-based instructional approaches to teaching the building blocks of drone technology careers, i.e., programming, design and fabrication, and commercial drone pilot credentialing.
2. A course design that covers the requirements for the US FAA Part 107 commercial remote pilot license test in a classroom setting.

3. Multi-methods assessments for students' gains in skills and knowledge and attitudes towards an active learning-based approach for drone education.

We note a growing number of drone-based educational efforts that focus on middle and high school students [103, 299]. These efforts are an exciting development and point to an increase in the use of drones across K-12 in the coming years. The presented work documents drone educational effort in undergraduate settings and congruence with suitable drone industry needs and requirements.

7.4 Active Learning for Drone Education

The active learning approach focuses on exploring drone careers' building blocks and is elucidated in this section. The approach was implemented in an undergraduate class, Drone Science Fundamentals, attended by students from mechanical, electrical, computer, biomedical, chemical engineering, and the computer science (C.S.) departments at a research-focused institution located in the United States. The approach featured three modules: Programming using Indoor Drone, Part 107 Federal Aviation Administration Remote Pilot License Test, and a Capstone Drone Technology Project. The course has been conducted four times since September 2019, and 215 students have registered for this class across four offerings.

Active learning approaches include interactive lectures that featured discussions, indoor drone programming, and a capstone project to immerse individual students and student-led teams in the technical challenges of designing and prototyping drone-based products and services. Each of these active learning approaches were guided by the *Interactive Constructive Active Passive* (ICAP) modes of cognitive engagement [63]. The ICAP framework enables educators to design and examine active learning approaches due to the observable and assessable nature of the four student engagement modes it outlines. Table 7.1 summarizes the observed student

engagement behaviors in the context of the ICAP framework during the 3 course modules.

Table 7.1 Student Engagement Modes Observed in the Context of the ICAP Framework during the 3 course Modules [63]

#	**Course module**	**Interactive**	**Constructive**	**Active**	**Pass - ive**
1.	Python programming with Indoor Drone	Collaborative programming assignments with Indoor drone	Creating drone programs based on instructions during class session	Rewinding and/or pausing Python videos, commenting Python code demonstrated in the classroom	-
2.	FAA Part 107	Discussing legal and societal implications of the FAA regulations during lecture sessions	Articulating applicable FAA regulations for capstone project statements	Rewinding and/or pausing FAA videos, note-taking, use of commercial drone software applications to study airspace categories	-
3.	Capstone Project	Weekly group meetings, Discussions during progress presentations, project demo video creation	Open-ended drone programming and simulations, 3D CAD, 3D printing drone parts and evaluating them for correctness, electro-mechanical assembly of the drone project	Researching online resources such as YouTube videos and Online Blogs	-

The duration of each module is depicted in the left chart in Figure 7.3. The course content was hosted on two learning management systems (LMSs) – Canvas and Moodle [141, 235]. Each module featured topical pages on the LMS with videos, illustrations, and textual content. These modules are described in the sub-sections below.

The instructor and teaching assistant were from the electrical and computer engineering, mechanical engineering, and computer science departments at the university. The researchers in this study including the instructor and teaching assistants are working on integrating modern technology (drones/robotics) and industry-standard computer programming education into the classroom through the use of active learning.

7.4.1 Module 1: Programming using Indoor Drone

A key strength of modern sUAVs is their ability to be programmed for various automated, semi-autonomous, or fully autonomous tasks. There is a growing demand for professionals that can create software/firmware for drones. The key learning objective of this course module was to provide students a foundational introduction to drone task programming using an indoor quadrotor platform and the Python programming language.

Python is considered as one of the most popular programming languages [165]. It is an interpreted, high-level and general-purpose programming language that is concise and easy to read. This module was covered during the first 6-weeks of the semester, where the first two weeks focused on Python programming and the following four weeks focused on using Python programming to perform various automated and semi-autonomous tasks using indoor drones. Two indoor quadrotor platforms were used for the class – the DJI Tello and the LocoDrone by LocoRobo. Both these

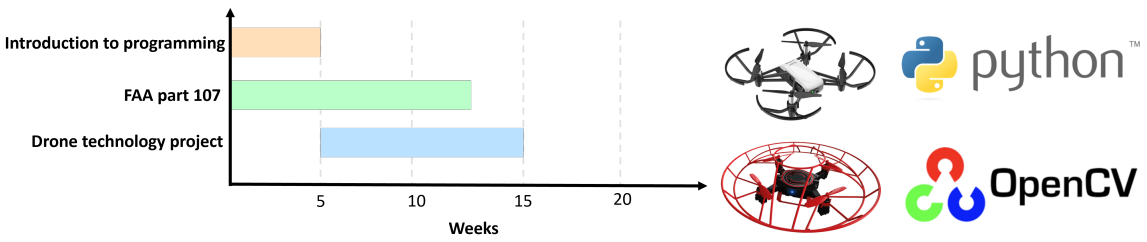


Figure 7.3 (Left) Duration for each module of the course. (Right) Two models of indoor drones were used in the class, the DJI Tello and the LocoDrone by LocoRobo.

quadrotors weigh less than 250 grams and can be programmed using Python. Both quadrotors have flight sensors that allow for safe and stable flight. The quadrotors can be programmed wirelessly using any Python Integrated Development Environment (IDE). Many students in the class had relatively little prior programming experience. The course module introduced the basic constructs of programming using Python and then gently guided students to explore functions, Python modules, data structures, flight patterns, and computer vision using indoor drones.

The module consisted of 3 assignments that were due every two weeks. Students were free to collaborate on the assignments. The first assignment focused on introductory Python exercises. The second and the third assignments focused on students using indoor drones for implementing flight patterns, obstacle avoidance, payload transportation, and computer vision-based decision making. OpenCV was used for the computer vision tasks [250]. Students were free to use Python IDEs of their choice. These free-of-cost IDEs included the powerful Google Colab platform, PyCharm, Spyder by Anaconda, and Microsoft Visual Studio [39, 147, 226, 332].

7.4.2 Module 2: Part 107 Federal Aviation Administration Remote Pilot License Test

The second-course module covered the topics required to pass the FAA Part 107 license test. Given the expansive nature of the test, this module spanned 12 weeks with the primary objective to fully prepare students for taking the license test by the end of the semester.

As per the FAA Part 107 requirements listed on [100], the following topics covered include:

1. Applicable regulations relating to small unmanned aircraft system rating, privileges, limitations, and flight operation,
2. Airspace classification, operating requirements, and flight restrictions,

3. Aviation weather sources and effects of weather on small unmanned aircraft performance,
4. Small unmanned aircraft loading and performance,
5. Emergency procedures,
6. Crew resource management,
7. Radio communication procedures,
8. Determining the performance of small unmanned aircraft,
9. Physiological factors (including drugs and alcohol) affecting pilot performance,
10. Aeronautical decision-making and judgment,
11. Airport operations, and
12. Maintenance and preflight inspection procedures.

Videos created and curated by our team formed a key part of this module and provided an engaging way to teach an otherwise serious set of topics. Class sessions involved students discussing and debating the legal and societal implications of these regulations. Students used commercial drone software applications to study airspace categories and other aviation-related information [13, 29, 163]. The assessment of the FAA curriculum was conducted through 4 quizzes and 2 FAA practice mock-tests. The quizzes and mock-tests featured multiple choice type questions that were hosted on LMS. Students were required to take two mock tests that featured representative questions similar to the actual FAA Part 107 test towards the end of this module. Students could see their results as soon as they completed the test. The students were allowed to attempt the quizzes and practice tests twice, and only their best grade was counted.

7.4.3 Module 3: Capstone Drone Technology Project

The third module featured student teams working on a capstone drone technology project. The project spanned the entire 15 weeks of the semester. The primary

instructional objective of this project was to allow student teams to work together effectively and apply the skills/knowledge learned from the course to build a tangible solution to a real-world problem. Each member was tasked with a meaningful role in developing the drone project similar to teams of corporate, military, or startup professionals. An important part of this team-based project was a well-defined timeline of project goals. The timeline was broken down into the following phases:

1. **Weeks 1- 4: Team Formation and Project Proposal**: In week 1 of the term, the key objectives and timelines of the capstone project were shared in detail with the students. Students were asked to form teams and propose their projects by Week 4 of the term. The teams could have between 2 and 5 members with technical backgrounds that were required for the project. If a student could not find a team independently, they were put in touch with other teams not yet filled to maximum allowed members.

   A template of the proposal document was provided to the students. The template included prompts to engage students to think deeply about their project idea and team composition. The prompts included questions about them: 1) problem statement, 2) commercial/societal significance, 3) technical approach, 4) key innovation, 5) expected outcome and deliverables, and 6) resources needed.

   Students used the Canvas/Moodle forums to float ideas, discuss potential collaborations, and coalesce around a coherent project narrative with other classmates. During this period, students were required to engage with the professor (Abichandani) and the teaching assistant (Lobo) through weekly 1-on-1 meetings during their office hours to craft their projects' scope and technical deliverables appropriately.

   The project categories included commercial, research, and military drones, where some of the common application areas included agriculture monitoring, pesticide spraying, fire-fighting, stealth reconnaissance, medical supply delivery, and surveying. Several drone project outcomes are depicted in Figure 7.1. Almost 90% of the projects involved the actual fabrication of the physical drone. Around 10% of the projects involved 3D CAD, dynamic simulations, and computational fluid dynamics analysis using software applications such as Solidworks and ANSYS [24, 326].

2. **Weeks 5-7: Weekly Project Planning Meetings**: At the end of every class, the professor and T.A. met with each group separately and discussed their project plans while providing feedback on their designs technical approach, and associated bill-of-materials. Most of the students that took this course

did not have any first-hand experience working with drones. Therefore the instructors needed to work closely with the student teams to ensure their project scopes, timelines, costs, and deliverables set them for success in the class. By the end of Week 7, students had a well-developed project plan. The Robotics and Data Laboratory (RADLab), directed by Abichandani, provided the parts necessary for the projects. The parts included drone frames, motors, electronic speed controllers, micro-controllers, batteries, battery chargers, and a power distribution board.

3. **Weeks 8-14: Weekly In-Class Project Progress Presentations**: After the project plan development, students were expected to lead the project execution while having minimal interactions with the instructors. The key requirement during this phase was a 3-minute long weekly in-class progress presentation that outlined their weekly progress. The progress presentation featured three slides that covered the weekly progress, challenges, and expected milestones for next week. Since the projects involved drone hardware and flight tests, students were free to demonstrate their project hardware or videos during the presentation.

4. **Week 15: Project Video**: The final deliverable for the capstone project was a video presentation along with a technical report. The video presentation was to include the following information: 1) Introduction and project statement, 2) Demonstration of the functions of the drone project, 3) Description of design challenges and approaches adopted to solve them, 4) Learning resources referred to while working on the project, 5) Anticipated challenges at the beginning of the project, and 6) Approaches taken to solve the challenges.

7.5 Multi-Methods Assessment and Results

The Drone Science Fundamentals course has completed four successful iterations since September 2019. The course attracted students from various backgrounds: 30% of the registered students were electrical engineering majors, 20% of the students were computer science majors, 40% of the students were mechanical engineering majors, and the remaining 10% of students came from a mix of other engineering backgrounds such as civil engineering, chemical engineering, biomedical engineering, and general engineering.

Approximately 90% students identified as male and approximately 10% students identified as female. Other gender identities were not reported in the data collected. We note that this disproportionately low number of female students in the course is a

topic of concern. Recruiting and outreach efforts are being implemented to encourage female students to enroll in the course.

The class was open to students from all academic standings above the first year. The student registration data showed that 83.33% of the registered students were in their final (senior) year, 14.28% of the students were in their third (junior) year, and 4.76% of the students were in their second year (sophomores).

Overall, the institutional student community is highly diverse, and 48.83% of the registered students identified as "European American or White", 16.27% students identified as LatinX or Hispanic, 27.9% students identified as Asian, and 4.65% students identified as African-American.

A case study approach involving multiple assessment methods was adopted to assess students' Python programming skills, FAA Part 107 knowledge, perceived gains in skills and knowledge, and attitudes towards an active learning-based approach for drone education. Both quantitative and qualitative assessments were conducted to gain multiple insights into the effectiveness of active learning. Table 7.2 presents a summary of the assessment types, methods, targeted course component, and data analysis method.

Table 7.2 Salient Features of the Assessment Strategies

Assessment type used for Research Question (RQ)	**Assessment method (Data Sources)**	**Targeted course component**	**Data analysis**
Direct formative (RQ1)	Programming assignments	Evaluated knowledge of Python programming and indoor drone operation	Program correctness, Documentation, Program readability
Direct formative (RQ2)	Multiple choice quizzes	Evaluated knowledge of FAA curriculum	Descriptive statistical analysis for auto-graded quizzes
Direct summative (RQ2)	Practice Tests	Evaluated knowledge of FAA curriculum	Auto-graded practice test
Indirect quantitative (RQ1, RQ2)	Likert-scale feedback survey and Student Assessment of Learning Gains [303]	Value of the inclusion of indoor drone in programming assignments, Learning gains for FAA part 107, Active learning approach	Descriptive statistical analysis for Likert-scale responses
Indirect qualitative (RQ2)	Focus group interviews	Drone project development	Inductive analysis using the constant comparative method
Indirect qualitative (RQ2)	Video presentations	Analyze drone project outcomes	Inductive analysis using the constant comparative method

Table 7.3 Descriptive Statistics for Grades Obtained by 113 Students on their Python Programming Assignments and FAA quizzes. Each Assignment and Quiz was Worth 10 Points.

Python	Assignment 1	Assignment 2	Assignment 3
Mean	9.11	8.66	8.6
Std. Dev.	2.57	2.83	3.78
FAA	Quiz 1	Quiz 2	Quiz 3
Mean	9.53	9.3	9.02
Std. Dev.	1.89	2.01	2.39

7.5.1 Direct Formative Assessment

Formative assessments were used for Python programming with indoor drones and FAA quizzes.

Python Programming Assignments The programming module featured three assignments that were due every two weeks. The assignments were Python programming-based and used the indoor drone. The assignments were worth 10 points each. These assignments were graded along the following dimensions:

1. Program correctness: The Python program should work correctly for all inputs. The drone completes the tasks as per the requirements. Correct usage of programming constructs.
2. Documentation: The Python program should have meaningful comments that explain the functions/methods being used. Video demonstrations should clearly show the operation of the indoor drone according to the specifications.
3. Readability: Variables and functions should have meaningful names. Code should be organized logically into functions where appropriate.

Table 7.3 depicts the mean and standard deviation of each assignment grade for 113 students. The mean of the grades received by the students for the Python programming assignments 1, 2, and 3 was 9.11, 8.66, and 8.6, respectively. The standard deviation observed for the Python programming assignments 1, 2, and 3 was 2.57, 2.83, and 3.78, respectively.

Multiple Choice Question FAA Quizzes The FAA Part 107 module featured quizzes and practice tests, which followed the same format as the FAA part 107 remote pilot's license exam. The questions on the quizzes featured multiple-choice questions relating to the FAA topics covered that week. The quizzes were created on the LMS for the course and were designed to be auto-graded. The auto-graded quiz provided the students with instant feedback on the questions attempted. Each question had only one correct answer.

Table 7.3 depicts the mean and standard deviation of each FAA quiz grade for 113 students. The mean of the grades received by the students for FAA quizzes 1, 2, and 3 was 9.53, 9.3, and 9.02, respectively. The standard deviation observed for FAA quizzes 1, 2, and 3 was 1.89, 2.01, and 2.39, respectively.

7.5.2 Direct Summative Assessment

Direct summative assessments were used for FAA mock tests and a drone capstone project.

FAA part 107 Mock-tests FAA part 107 mock-tests were conducted to provide the students with a summative assessment of all the FAA curriculum knowledge gleaned from the course. Two mock-tests were conducted, each consisting of 60 questions. Each mock test was designed to be as close to the actual FAA commercial remote pilot license exam. The tests were graded out of 10 points each. The questions on the test were multiple choice questions from the FAA curriculum covered in class. The mock tests were hosted on the LMS. Students were allowed to retake the tests once, if they were not satisfied with their performance the first time. The FAA mock-tests were auto-graded and provided the students with instant feedback about the grasp of the FAA modules. Figure 7.4 depicts the grade distribution for FAA mock-test 1 and FAA mock-test 2. It is observed that 90.26% of the students received 9-points or higher in FAA mock test 1 and 84.07% of the students received 9-points

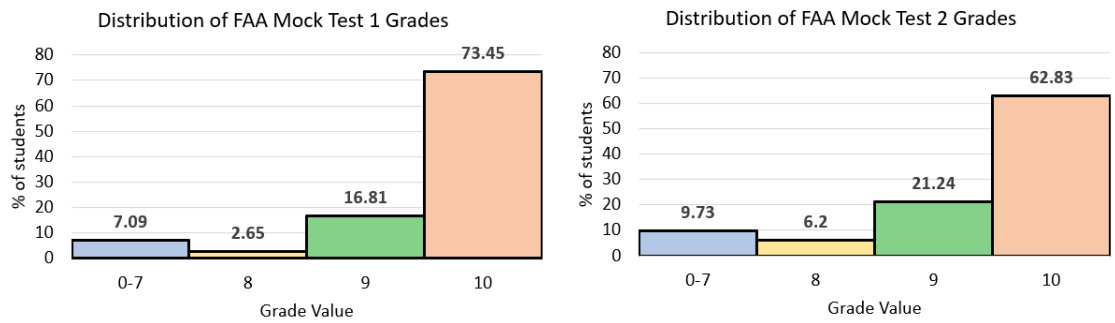


Figure 7.4 Grade distribution for FAA mock-test 1 (left) and mock-test 2 (right) indicates that a majority of students scored 9 or more points. Each mock-test was worth 10 points. Total number of students = 113.

or higher in FAA mock-test 2. In both the FAA mock tests, less than 10% of the students scored 0-6 points. 100% of the students who attempted the FAA commercial remote pilot license exam passed and received their drone pilot's license.

Table 7.4 Rubric for Drone Technology Capstone Project

	Excellent	**Average**	**Poor**
Project Completion	All deliverables met	Some deliverables met	No deliverables met.
Video presentation	Meets all the video presentation requirements, excellent presentation skills	Partly meets the requirements of the video presentation, And mediocre presentation skills	Does not meet any of the requirements of the video presentation, or no video submitted.
Communication	Excellent at communicating challenges and progress made	Average at communicating challenges and progress made	Poor at communicating challenges faced and little to no progress weekly
Innovation	Implemented a unique feature or aspect to the project	Proposed an innovative concept but did not deliver	No innovation implemented
Realizability	project is working, and real-world application demonstrated	Project is working, but real-world applicability, not possible	project is not working, and real-world applicability is not possible.
Report organization	Excellent presentation demonstrates understanding of FAA regulations.	Average presentation, satisfactory understanding of FAA regulations	Poor organization of report, and sub par understanding of FAA regulations

Capstone Project The project submission was evaluated according to the rubric provided in Table 7.4. The rubric was adopted from resources available on [262] while keeping in mind the objectives of the capstone drone project. Based on the final presentations, 70% demonstrated a fully working project as per the proposed description, 20% presented a partially working project where they had a physical model, but the system was not working as intended, and finally, 10% presented only simulations or only CAD designs of the project.

7.5.3 Indirect Quantitative Assessment

A feedback survey was provided to the students towards the end of the course to assess three core course aspects: 1) The value of including the indoor drone in Python programming assignments, 2) students' perceived learning gains in the part 107 FAA curriculum, and 3) student attitude towards active learning approaches. The questions for core aspects 1 and 3 were adapted from [303] and followed a 5-point Likert scale. The 5-point responses were "strongly disagree," "disagree," "neutral," "agree," and "strongly agree." Core aspect two was used to measure the students' perceived learning gains for each module on the part 107 FAA commercial remote pilot license exam. The options were no gain, little gain, moderate gain, significant gain, and exceptional gain. The following discussions report on the findings from the feedback survey.

Value of Including Indoor Drone in Python Programming Assignments

Indoor drones formed an important part of the programming assignments as they turned an otherwise computer-screen-based activity (programming) into a hands-on learning opportunity. The following three statements were posited to students with a goal to understand student attitudes towards the inclusion of indoor drones:

1. "I felt the time used for this activity was beneficial,"
2. "I saw value in the activity," and
3. "I felt the effort it took to do the activity was worthwhile."

Student responses to the above 3 statements are shown in Figure 7.5. Students responded favorably to indoor drone programming activities, with over 50% of the students strongly agreeing to the prompts, 35% indicated moderate agreement to the prompts, and less than 10% of the students had neutral feelings about the prompts. Only 4% of students disagreed with prompt 3, indicating that a tiny fraction of

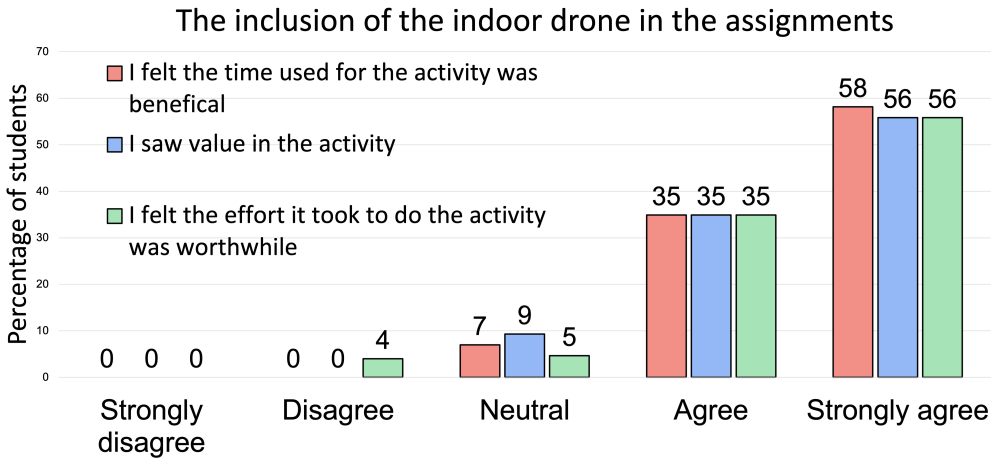


Figure 7.5 Feedback from students on the inclusion of the indoor drone as part of Python programming assignments. Total number of students = 113.

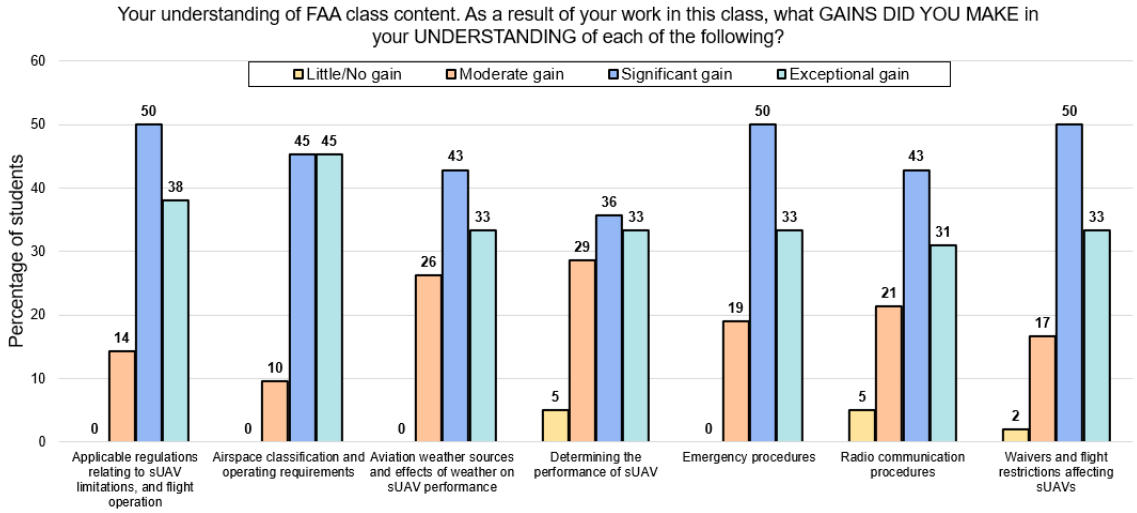


Figure 7.6 Survey responses about learning gains for the FAA Part 107 test content show moderate to exceptional gains for most topics and for most students. Total number of students = 113.

students felt that they put in more than necessary effort to do the indoor drone-based Python programming.

Learning Gains for FAA Part 107 Test Figure 7.6 depicts the learning gains survey responses of students for the FAA Part 107 test content covered in the class. The questions in this survey section were adapted from resources available on [303] and focused on seven key topics covered in the class. The results indicate a moderate

to exceptional gain on most of the FAA part 107 exam topics. In only 3 of the 7 FAA Part 107 topics, 5% (or less) students indicated little gains in their learning.

Active Learning Approach Feedback from students on different active learning aspects incorporated in the course is depicted in Figure 7.7.

The results indicate that the students responded positively to the active learning aspects of the class. Of note is the students' positive response to the "receiving feedback from the instructor" and "the number and pacing of the assignments" aspects of the class with 64%, and 62% of the students selecting *"Strongly Agree"*. The weekly interactions with the instructor provided valuable feedback to the students and provided guidance in completing the project. The one-on-one interactions with the instructor formed a crucial aspect of the active learning process. On average, 90% of the students responded *"Strongly Agree"* or *"Agree"* on these aspects of the class.

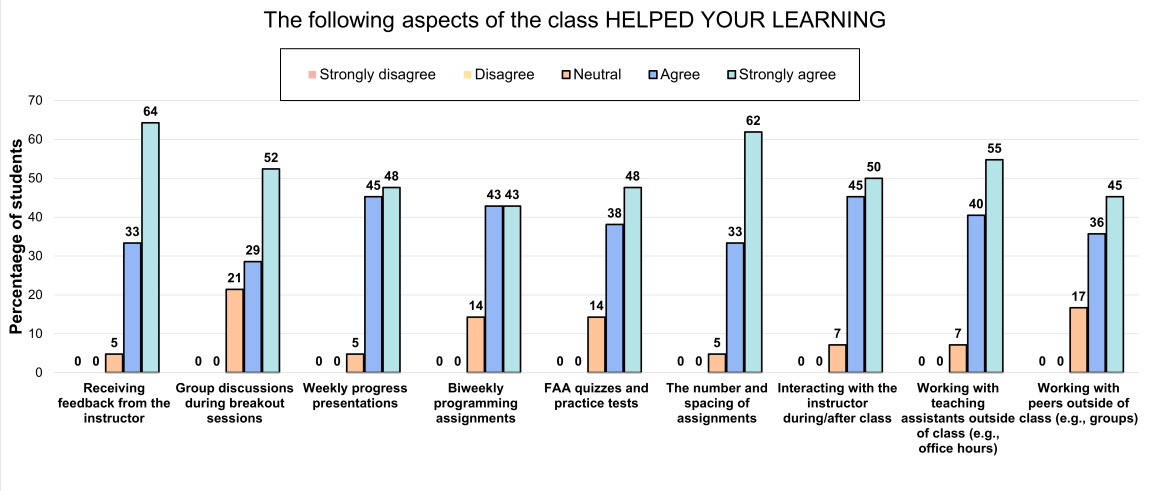


Figure 7.7 Student responses about the different aspects of the class which helped in their learning skewed towards majority positive responses (agree and strongly agree). There were no negative responses recorded. Total number of students = 113.

7.5.4 Indirect Qualitative Assessment

Student focus groups and video presentations were used as two data sources to unpack insights about student attitudes towards the drone capstone project and its effectiveness as an active learning-based instructional approach.

Student Focus Groups Focus group interviews were conducted with project teams to gather data from the students on their thought processes behind project topic selection, planning, and resource management. The focus group interviews were conducted by the professor (Abichandani) and teaching assistant (Lobo) to help the project teams formulate and plan their projects between Weeks 1 and 3 of the term. The focus group sessions allowed the gathering of multiple perspectives and identifying consensus or shared views about the topic under consideration [260]. A total of 13 focus groups were conducted using the Cisco WebEx software [70]. Each focus group was approximately 30-minutes long. At the start of each session, research consent was obtained from each student. The interactions with the students revolved around a set of semi-structured questions which were prepared beforehand to steer the conversation. As the students responded to the questions, follow-up questions were asked to gain deeper insight. Overall, the questions focused on understanding the reasoning and research behind selecting their project topic, team member roles, expected project outcome, technical approach, and resources needed.

The focus group session transcripts were analyzed inductively according to the constant comparative method [302]. Based on the analysis, we present the following three recurring themes along with focus group excerpts:

1. **Theme 1 - Project Selection and Planning**
   The drone project topics selected by the students covered a variety of applications. Student responses noted that students often relied on recognizing opportunities observed in their local surrounding environments in the project selection process. For example, members of one of the student teams who were

working on a drone-based food delivery system commented, *"It is something that we could all relate to because all of us have ordered food from services like company X, and we thought since we were learning about drones, it would just be advantageous to use a drone rather than an automobile."* The student identified a real-world problem of food delivery that could be improved using drone technology.

In addition, students were asked about the first steps they would take towards their project to gain insights into their planning process. Some student teams had a well-formulated roadmap consisting of details about their technical approach and design strategies. These student teams had begun preliminary research about the key technical challenges of their project, as evidenced in the following student comment *"We started doing some research about how we can make drones quieter, and most of what we've seen is lowering the speed at which the blades are rotating or making them longer."* However, it was their first time working on an open-ended, multi-disciplinary project involving drones for many student teams. These teams did not have a well-defined project plan, as evidenced in the student comment, *"We have not officially talked about it to each other, but we would like to have a concept design of what our project is going to look like, and the list of parts needed."*

When asked about the resources students might need for their projects, most of the students stated that they had looked through online sources. For example, one student said *"I believe in a country in Europe, there is a company working on a drone-based medicine delivery solution."*. Students reported that most of their research consisted of studying online, do-it-yourself (DIY) resources such as Instructables, Hackaday, and YouTube [120, 140].

2. **Theme 2 - Drone Project Task Delegation**
   Drone-based projects are multi-disciplinary, requiring students to have a variety of engineering skill-sets. Students were cognizant of their team's strengths, and the delegation of tasks was based on their engineering background, as evidenced by the following comment *"We would allocate tasks based on our skills. I imagine some of us are probably stronger in programming, and some would be strong in CAD"*. Some student teams indicated a willingness to acquire skills outside their background depending on the needs of the project, which is evidenced from the following comment *"We all have a basic understanding of circuits. We are all learning Python now, but we have little background in coding. If we need to research in order to learn new things to code, it is not a problem at all."* We observed that most of the project teams assigned roles based on their engineering/C.S. backgrounds, with a select few students venturing out of their core competencies. For example, an electrical engineering student diving into CFD simulations or a mechanical engineering student working on Python GUI interface coding.

3. **Theme 3 - Anticipated Research Challenges**
   A majority of students anticipated challenges associated with learning new

skills outside their engineering curriculum. Students from the mechanical engineering background repeatedly mentioned that they anticipated needing help with the programming aspects of their project. This sentiment is evidenced in a mechanical engineering student's comment, *"Oh, coding for sure. That is definitely going to be it for me. I'm too worried about the coding aspect. That is probably our weakest point as a team"*. Another anticipated challenge included the procurement of the right drone parts and 3D printed components, which was underscored by the comment, *"I think finding the right materials, and resources, especially for the blade design, will be a challenge"*.

The uncertainties caused by the COVID-19 related campus disruptions also weighed on students' minds as evidenced by this comment, *"Assuming we have the materials and equipment needed. It just really depends on if campus stays open and we have the resources."*

Capstone Project Video The final deliverable for the capstone project was a video presentation. Each of the project teams had to prepare a video, which demonstrated the working of their drone project, a discussion of the challenges faced, and a description of the final outcome. Teams were given a list of questions/prompts to answer in their video. These questions/prompts were: 1) Demonstrate the working of your project, 2) Describe the design challenges you faced and how you resolved them?, 3) What learning resources did you consider in designing your project?, 4) What areas did you anticipate you will need the most help with?, and 5) How did you eventually resolve the challenges?

These questions and prompts were designed for students to think deeply about their journey during the final project. The video presentations were processed through an online caption generator tool to generate transcripts for analysis [253]. The transcripts were analyzed using the constant comparative approach similar to what was used for the focus group interview [302]. The impact of active learning was evident across two recurring themes observed in the videos: 1) project resource management and 2) resolution strategies for project challenges. For both these themes, it was observed that students were actively engaged in tasks similar to those observed in real-world careers, as discussed next.

1. **Theme 1 - Capstone Project Resource Management**
   Overall, students reported high self-efficacy in identifying and procuring resources to complete their projects. Students noted that online tutorials, YouTube videos, and open-source code/CAD designs proved adequate project resources. For example, a student expressed, *"Utilizing resources such as GrabCAD and YouTube, we were able to figure out specific details of real-life flash discharge mechanism."*

   On several occasions, students appreciated that while the onus of identifying, designing, prototyping, and managing the parts for their projects was on them, they could ask for some assistance from the instructors in this matter when they needed it. This sentiment is particularly exemplified in the following student comment *"While parts were designed by our team using 3D models and prototyped at the Makerspace, the selection of the servo was made with the assistance of the professor. These resources were crucial for the development of the prototype."*

2. **Theme 2 - Resolution Strategies for Project Challenges**
   On the whole, teams were able to solve their project challenges with effective resolution strategies. Key observations about resolution strategies employed by students when faced with different challenges are provided in the following:

   (a) Students addressed most design challenges using a combination of rapid prototyping and physical alterations to parts as evidenced in the following comment *"... we started by printing a thinner adapter plate, which reduced our distance from the bolt to the threads in the frame. We additionally ground down the thickness of the mounting end of the legs, which reduced the distance again the bolts had to travel, which allowed us to start the threads and securely mount everything to the frame."*

   (b) Student teams working on advanced design-oriented projects had to develop solutions that fit within the duration of the course term. This required out-of-the-box design decisions, as is described in the following comment *"So one of the biggest design challenges that we faced was deciding on how to create the mechanism to turn it from flying to driving. Our first design use propellers, which were at an angle at all times. This design was not feasible for the class duration because it would require remodeling the entire drone frame...[instead] we found a better method to rotate the arms using a servo."*

   (c) Approximately 10% of the project teams struggled to overcome their challenges. These challenges were primarily logistical in nature and involved late shipping from vendors, faulty parts shipped from vendors, and back-ordered parts on e-commerce websites. For example, a student mentioned that *"the legs that we purchased separately for the assembly ended up being too large to hold the drone up."*

(d) In instances where a team struggled to overcome project challenges, the members approached the instructors for guidance. In one of the video presentations, a student highlighted that *"Building the drone in terms of fitting all the parts in the frame was challenging. However, the availability of the T.A.s during group meetings helped resolve these challenges."* The help provided by the instructors was minimal but proved to be sufficient in resolving most of the issues faced by student teams.

(e) During focus group interviews, the mechanical engineering students anticipated software programming as a challenge. Most of these students addressed this anticipated challenge by teaming up with other students with a programming background.

7.5.5 Discussion

The use of multiple methods for assessment was instrumental in helping us explore the answers to the research questions. Quantitative assessment methods used provided evidence about the effectiveness of active learning methods for drone education, while qualitative assessment methods generate descriptive understanding of student perceptions, which lent further evidence to the quantitative data [185].

Computer Programming and Drones: Computer programming is usually met with lack of student engagement or student resistance [81, 196, 211]. Several approaches have been reported in the literature to mitigate student resistance to active learning strategies [349]. Results of the quantitative assessment suggest that using a drone for computer programming assignments may assist in fostering student interest and engagement (or reduce resistance) to active learning for computer programming by providing a real-world context for an otherwise abstract topic. This is in line with other studies which underscore the importance of carefully designing active learning instruction to align with student interests to promote engagement [41, 312, 317].

FAA Part 107 Test Knowledge Gains: Strong student performance on the summative FAA mock-tests suggests that they could grasp the core technical, legal, and societal implications of the FAA Part 107 commercial drone pilot licensure. Active discussions during the class sessions formed a vital aspect of the FAA module.

Incorporating such types of interactions in the learning paradigm allows students to gather authentic information and explore peer perspectives that help solidify their understanding of critical concepts [44, 184]. When surveyed about different aspects of the module that were helpful for FAA learning, 95% of the students responded positively ("strongly agree" and "agree") to the *interacting with the instructor during/after class* aspect. Additionally, unsupervised online quizzes, similar to the FAA quizzes, where students are allowed multiple attempts may enhance student performance on summative assessments such as the FAA mock-tests [220]. These results resonate with other engineering education studies that note knowledge gains when using active learning [229, 384].

Challenges and Resolution Strategies: The qualitative insights unpacked from the focus group interviews and video presentations suggest that when faced with technical, logistical, and managerial challenges during their capstone projects, a majority of the students were able to resolve them. Active learning approaches are known to have a positive effect on increasing student performance in STEM learning, especially in students who have to propensity to be relatively independent of instructor assistance in their learning approaches [107, 208]. At the same time, a handful of students were unable to resolve all challenges and requested assistance from the instructional staff. Given the heterogeneity in a class attended by final year students from multiple engineering and CS backgrounds, educators may find that some students are relatively less receptive to the independent nature of active learning activities as noted in [208].

7.6 Conclusion

Engineering educators are actively trying to prepare students with skills and knowledge relevant to the drone industry. This chapter reports active learning-based instructional approaches to prepare undergraduate students for engineering, software,

and piloting careers in the drone industry. The approaches were implemented in an undergraduate class that engineering and computer science students attended. The course included using indoor drones to cover drone programming, the use of interactive videos and commercial software to cover the US FAA Part 107 test-prep curriculum, and a capstone drone technology project that focused on the use of CAD software and rapid prototyping for real-world applications. Multiple assessment methods were implemented to provide quantitative evidence and a qualitative understanding of the effectiveness of active learning in the context of drone career preparation. Quantitative assessments indicated that most students excelled at the technical subjects covered in the class. Focus groups and video presentation analysis revealed that students found regular check-in meetings with the professor and teaching assistant crucial for the success of their capstone projects. An interesting observation was that despite students performing well on their programming assignments and FAA quizzes, most students performed project tasks that aligned with their degree curriculum and core competencies when it came to the capstone projects. Further research needs to be conducted to understand the reasons behind this and encourage students to perform project tasks across multiple disciplines. As educators continue to bring drones to their classrooms, we anticipate that active learning approaches will be increasingly employed. The information presented in this chapter may serve as a guide for future drone educators.

CHAPTER 8

CONCLUSION

This work presented in this dissertation began with the aim to study the challenges in the outdoor deployment of multiple quadrotor systems. We identified five critical challenges: 1) wind measurement and simulation, 2) outdoor motion planning, 3) safety and collision avoidance, 4) secure communication, and 5) drone education. Each chapter in this dissertation provides a detailed description of the work we conducted to study and address these challenges.

- **Wind Measurement and Simulation:** Three representative wind speed measurement techniques were reviewed and compared on three metrics including cost of sensors, speed and direction data, sensor resolution, and physical mounting. Each of the techniques have their respective strengths and limitations based on the requirements, such as pressure-flow sensors being the least expensive but provide only wind speed data, where as ultrasonic sensors provide wind speed and direction data simultaneously. The tilt-angle approach can provide wind speed and direction data, however the quadrotors need to be calibrated in wind tunnels to obtain high accuracy wind speed measurements. We also reviewed three representative wind simulation techniques and found that the Dryden wind gust model is suitable for simulating quadrotor missions in large open areas. Additionally we developed an open-source Python implementation of the Dryden wind gust model to simulate wind disturbances for quadrotor missions.

- **Outdoor Motion Planning:** After getting an in-depth understanding of wind disturbances on the performance of sUAVs we presented an experimental evaluation of an OF-GM-SARSA-based RL planner to address the flocking problem for multiple quadrotors operating outdoors. Simulation and experimental evaluation of the performance of a flocking controller in windy environments revealed that the OF-GM-SARSA technique generalizes the behaviors trained in simulation to real-time interactions. While the learned generalized behaviors were robust to wind disturbances in HWIL and flight tests they did not guarantee collisions.

- **Safety and Collision Avoidance:** To ensure safety during outdoor multi-quadrotor flights, we explored the use of CBFs with an optimization based planner. Experimental validation of this decentralized, RH-MINLP + CBF based motion planner revealed that the use of CBFs provided significant

robustness against safety violations and potential mid-flight collisions that could have resulted due to the wind disturbances, in turn reducing mission times. The simulation and experiment results demonstrated the substantial computational time savings and scalability of the framework accorded by the novel simulated annealing-based numerical solution process when compared to a well-established numerical solution method.

- **Secure Communications:** Next we addressed the potential security risks associated with multi-quadrotor communication through the use of a blockchain. We documented the hardware and software setup, and experimental results for an Ethereum based secure information exchange system mounted aboard multiple quadrotors sUAVs. The hardware architecture leveraged a low-cost NVIDIA GPU for mining during flight and Raspberry Pis as non-miner Ethereum nodes for data collection and storage. The software architecture leveraged Ethereum's ability to program smart contracts for secure, confidential, and tamper-proof data access, and a decentralized file system IPFS for data storage. The implementation of a private Ethereum network provided security features such as confidentiality, data integrity, and non-repudiation. The combined use of Ethereum blockchain and IPFS provided a distributed data storage system while avoiding a single point of failure for the network. The results from the outdoor experiment flight test point to the feasibility of a blockchain-enabled communication system for multiple sUAVs.

- **Drone Education:** To address the growing need for sUAV trained professionals in the US, we designed, developed, implemented, and evaluated active learning based approaches for sUAV education. A CBL-based approach provided an inclusive environment for 97 undergraduate and graduate students at NJIT to immerse themselves in the technical challenges of drone building, flight simulations, and piloting. Moving past a competition, a semester-long active learning-based course for undergraduate students was operationalized which focused on hands-on curricular experiences related to engineering, software, and piloting careers in the sUAV industry. Multiple assessment methods were used to collect quantitative data and qualitative evidence about the effectiveness of these active learning approaches. Quantitative assessments revealed that revealed that over 90% of the students achieved all the technical learning objectives, and that most students excelled at the technical subjects covered in the class. Qualitative assessments revealed that students found the CBL approach to be an engaging method of introducing drone technology. Focus groups and video presentation analysis revealed that students found regular check-in meetings with the professor and teaching assistant crucial for the success of their capstone projects. As educators continue to explore active-learning instructional approaches for drone education, we anticipate that the information provided in this dissertation may serve as a guide for future sUAV educators.

8.1 Future Directions

This section presents a list of possible future directions for this dissertation.

Scalability in outdoor flight tests: A key contribution of this dissertation was the experimental validation via outdoor flight tests. Scalability is one of the key metrics used to evaluate the performance of a multi-sUAV system as it provides insights about the ability of motion planners and inter-sUAV communication infrastructure to handle computationally demanding real-world scenarios. While we presented results from outdoor flight tests with up to 6 quadrotors, future works will explore conducting outdoor flight tests with 12 to 15 quadrotors.

Fault Tolerance in Outdoor Flight Tests: When conducting outdoor flight experiments there are several uncontrollable variables that adversely impact the results of the flight tests. These variables can include environmental conditions, hardware failures, drone fly-away, and network disruptions. When an Ethereum node experiences WiFi issues or disruptions, it is important that the remaining nodes continue the data transfer without getting affected. A miner node losing WiFi connection can be catastrophic to the network as is observed by our experiments. Having more than one miner node in the network can add redundancy and provide protection against such failures. Additionally when a drone breaks flight formation and is out of radio communication range, the flight controller software needs in-built fault tolerance to command the drone to execute a Return-to-Home (RTH) procedure. In the case of large scale systems with more than 12 quadrotors in the system, it is essential to have fault-tolerant techniques that ensure the safety of the multi-vehicle system with minimum compromises to the success of the flight mission. Future work will explore approaches to build fault tolerant approaches for large scale multi-quadrotor systems.

Energy-Aware Planning: The energy requirements of quadrotors directly affect the practicality of any outdoor operations [315, 340]. This requirement is especially true when quadrotors operate in windy outdoor environments where they need to compensate for gusts. The quadrotors used in this study (DJI M100) were evaluated in hover flight tests to have average flight time of approximately 1200 seconds. However when the payload of the quadrotor was increased by adding computing hardware, the average flight duration of the quadrotors reduced to approximately 720 seconds. As autonomous drones continue to be studied in various applications, these tasks usually have a long horizon and the energy required to fulfill the task often exceeds the initial capacity of the drone. Thus future efforts will study energy requirements of the drone along with the flight dynamics of the system in developing the optimization [278, 375] or reinforcement learning-based [243] motion planning algorithms.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Pramod Abichandani, Hande Benson, and Moshe Kam. Mathematical programming approaches for multi-vehicle motion planning: Linear, nonlinear, and mixed integer programming. *Foundations and Trends® in Robotics*, 2(4):261–338, 2013.

[3] Pramod Abichandani, Hande Y Benson, and Moshe Kam. Robust communication connectivity for multi-robot path coordination using mixed integer nonlinear programming: Formulation and feasibility analysis. In *2013 IEEE International Conference on Robotics and Automation*, pages 3600–3605. IEEE, 2013.

[4] Pramod Abichandani, Gabriel Ford, Hande Y Benson, and Moshe Kam. Mathematical programming for multi-vehicle motion planning problems. In *IEEE International Conference on Robotics and Automation*, pages 3315–3322. IEEE, 2012.

[5] Pramod Abichandani, Kyle Levin, and Donald Bucci. Decentralized formation coordination of multiple quadcopters under communication constraints. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3326–3332. IEEE, 2019.

[6] Pramod Abichandani, Deepan Lobo, Gabriel Ford, Donald Bucci, and Moshe Kam. Wind measurement and simulation techniques in multi-rotor small unmanned aerial vehicles. *IEEE Access*, 8:54910–54927, 2020.

[7] Pramod Abichandani, Solmaz Torabi, Shambadeb Basu, and Hande Y. Benson. Mixed integer nonlinear programming framework for fixed path coordination of multiple underwater vehicles under acoustic communication constraints. *IEEE Journal of Oceanic Engineering*, 40:864–873, 2015.

[8] Ahmad Abushakra, Firdouse Rahman Khan, Rasha AbdulWahhab, and Hilal Al Maqbali. Innovation in entrepreneurship education through competition based learning role: Students' perspective on the enhancement of soft skills. *Humanities and Social Sciences Reviews*, 7(5):862–869, 2019.

[9] Kevin A Adkins, Christopher J Swinford, Peter D Wambolt, and Gordon Bease. Development of a meteorological sensor suite for atmospheric boundary layer measurement using a small multirotor unmanned aerial system. *International Journal of Aviation, Aeronautics, and Aerospace*, 7(1), 2020.

[10] Maxim Ya Afanasev, Yuri V Fedosov, Anastasiya A Krylova, and Sergey A Shorokhov. An application of blockchain and smart contracts for machine-to-machine communications in cyber-physical production systems. In *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, pages 13–19. IEEE, 2018.

[11] Shubhani Aggarwal, Mohammad Shojafar, Neeraj Kumar, and Mauro Conti. A new secure data dissemination model in Internet of Drones. In *ICC International Conference on Communications (ICC)*, pages 1–6. IEEE, 2019.

[12] Aira. `https://aira.life/`. Accessed on: 05-12-2020.

[13] Airmap for drones. `https://apps.apple.com/us/app/airmap-for-drones/$id1042824733$`. Accessed on 05-20-2021.

[14] Palau Albert. Storing on Ethereum. analyzing the costs. `https://medium.com/coinmonks/storing-on-ethereum-analyzing-the-costs-922d41d6b316`, 2018. Accessed on: 10-24-2019.

[15] Kostas Alexis, George Nikolakopoulos, and Anthony Tzes. Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Engineering Practice*, 19(10):1195 – 1207, 2011.

[16] Abdulrahman Aliyu, Moustafa Elshafei, Abdul-Wahid A Saif, and Mujahid Dhaifullah. Performance evaluation of quadrotor with tilted rotors under wind gusts. In *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 294–299. IEEE, 2016.

[17] Sam Allison, He Bai, and Balaji Jayaraman. Modeling trajectory performance of quadrotors under wind disturbances. In *AIAA Information Systems-AIAA Infotech@ Aerospace*, page 1237. American Institute of Aeronautics and Astronautics SciTech, 2018.

[18] Sam Allison, He Bai, and Balaji Jayaraman. Estimating wind velocity using a neural network with quadcopter trajectories. In *American Institute of Aeronautics and Astronautics Scitech 2019 Forum*, page 1596, 2019.

[19] Randa Almadhoun, Maha Kadadha, Maya Alhemeiri, Maryam Alshehhi, and Khaled Salah. A user authentication scheme of IoT devices using blockchain-enabled fog nodes. In *15th International Conference on Computer Systems and Applications (AICCSA)*, pages 1–8. IEEE, 2018.

[20] Haider AF Almurib, Premeela T Nathan, and T Nandha Kumar. Control and path planning of quadrotor aerial vehicles for search and rescue. In *SICE Annual Conference*, pages 700–705. IEEE, 2011.

[21] Basma Alyazeedi and Diane C Berry. Competition-based learning (cbl) in nursing education. In *Nursing Education Research Conference: Generating and Translating Evidence for Teaching Practice*, 2018.

[22] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *18th European Control Conference (ECC)*, pages 3420–3431. IEEE, 2019.

[23] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pages 3420–3431. IEEE, 2019.

[24] Ansys. "https://www.ansys.com/". Accessed on 05-20-2021.

[25] Saira Anwar, Nicholas Alexander Bascou, Muhsin Menekse, and Asefeh Kardgar. A systematic review of studies on educational robotics. *Journal of Pre-College Engineering Education Research (J-PEER)*, 9(2):2, 2019.

[26] Inc. Applied Technologies. Tri-sonica mini weather sensor. http://www.apptech.com/products/ultrasonic-anemometers/TriSonica-mini/, 2016.

[27] Bilal Arain and Farid Kendoul. Real-time wind speed estimation and compensation for improved flight. *IEEE Transactions on Aerospace and Electronic Systems*, 50(2):1599–1606, 2014.

[28] Senthil Hariharan Arul and Dinesh Manocha. DCAD: Decentralized collision avoidance with dynamics constraints for agile quadrotor swarms. *IEEE Robotics and Automation Letters*, 5(2):1191–1198, 2020.

[29] B4ufly mobile app. https://www.faa.gov/uas/recreational_fliers/where_can_i_fly/b4ufly/. Accessed on 05-20-2021.

[30] Jasper Bakker, Aron Hammond, Daan Bloembergen, and Tim Baarslag. Rlboa: A modular reinforcement learning framework for autonomous negotiating agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 260–268. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[31] Tucker Balch and Ronald C Arkin. Behavior-based formation control for multirobot teams. *IEEE transactions on robotics and automation*, 14(6):926–939, 1998.

[32] Ezedin Barka, Chaker Abdelaziz Kerrache, Hadjer Benkraouda, Khaled Shuaib, Farhan Ahmad, and Fatih Kurugollu. Towards a trusted unmanned aerial system using blockchain for the protection of critical infrastructure. *Transactions on Emerging Telecommunications Technologies*, page e3706, 2019.

[33] Ezedin Barka, Chaker Abdelaziz Kerrache, Nasreddine Lagraa, Abderrahmane Lakas, Carlos T Calafate, and Juan-Carlos Cano. UNION: A trust model distinguishing intentional and unintentional misbehavior in inter-UAV communicatiON. *Journal of Advanced Transportation*, 2018.

[34] Bradley S Barker and John Ansorge. Robotics as means to increase achievement scores in an informal learning environment. *Journal of research on technology in education*, 39(3):229–243, 2007.

[35] Logan E Beaver and Andreas A Malikopoulos. An overview on optimal flocking. *Annual Reviews in Control*, 2021.

[36] Megan Beckett, Geoffrey Borman, Jeffrey Capizzano, Danette Parsley, Steven Ross, Allen Schirm, and Jessica Taylor. Structuring out-of-school time to improve academic achievement. ies practice guide. ncee 2009-012. *What Works Clearinghouse*, 2009.

[37] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[38] Jay Bhuyan, Fan Wu, Cassandra Thomas, Kai Koong, Jung Won Hur, and Chih-hsuan Wang. Aerial drone: an effective tool to teach information technology and cybersecurity through project based learning to minority high school students in the us. *TechTrends*, 64(6):899–910, 2020.

[39] Ekaba Bisong. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 59–64. Springer, 2019.

[40] Sumana Biswas, Sreenatha G Anavatti, and Matthew A Garratt. Particle swarm optimization based co-operative task assignment and path planning for multi-agent system. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–6. IEEE, 2017.

[41] Maura J Borrego, Michael J Prince, Christopher Evan Nellis, Prateek Shekhar, Cindy Waters, and Cynthia J Finelli. Student perceptions of instructional change in engineering courses: A pilot study. In *2014 ASEE Annual Conference & Exposition*, pages 24–1120, 2014.

[42] Kasper T. Borup, Thor I. Fossen, and Tor A. Johansen. A nonlinear model-based wind velocity observer for unmanned aerial vehicles. *IFAC-PapersOnLine*, 49(18):276 – 283, 2016. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

[43] Marcel Bottema. Urban roughness modelling in relation to pollutant dispersion. *Atmospheric Environment*, 31(18):3059–3075, 1997.

[44] John M Braxton, Willis A Jones, Amy S Hirschy, and Harold V Hartley III. The role of active learning in college student persistence. *New directions for teaching and learning*, 2008(115):71–83, 2008.

[45] Rex Edward Britter and Steven R Hanna. Flow and dispersion in urban areas. *Annual review of fluid mechanics*, 35(1):469–496, 2003.

[46] Martin Brossard, Jean-Philippe Condomines, and Silvère Bonnabel. Tightly coupled navigation and wind estimation for mini UAVs. In *AIAA Guidance, Navigation, and Control Conference*, page 1843, 2018.

[47] Caroline Brosy, Karina Krampf, Matthias Zeeman, Benjamin Wolf, Wolfgang Junkermann, Klaus Schäfer, Stefan Emeis, and Harald Kunstmann. Simultaneous multicopter-based air sampling and sensing of meteorological variables. *Atmospheric Measurement Techniques*, 10(8):2773–2784, 2017.

[48] Donald J. Bucci and Pramod K. Varshney. Decentralized multi-target tracking in urban environments: Overview and challenges. In *22nd International Conference on Information Fusion (FUSION)*, 2019.

[49] Juan C Burguillo. Using game theory and competition-based learning to stimulate student motivation and performance. *Computers and education*, 55(2):566–575, 2010.

[50] Ali Tevfik Buyukkocak, Derya Aksaray, and Yasin Yazıcıoğlu. Planning of heterogeneous multi-agent systems under signal temporal logic specifications with integral predicates. *IEEE Robotics and Automation Letters*, 6(2):1375–1382, 2021.

[51] Davide Calvaresi, Alevtina Dubovitskaya, Jean Paul Calbimonte, Kuldar Taveter, and Michael Schumacher. Multi-agent systems and blockchain: results from a systematic literature review. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 110–126. Springer, 2018.

[52] José M Cañas, Diego Martín-Martín, Pedro Arias, Julio Vega, David Roldán-Álvarez, Lía García-Pérez, and Jesús Fernández-Conde. Open-source drone programming course for distance engineering education. *Electronics*, 9(12):2163, 2020.

[53] Rodrigo Canek, Pablo Torres, and Oscar Rodas. Encouraging higher education stem careers through robotics competitions. In *IEEE Integrated STEM Education Conference (ISEC)*, pages 1–6. IEEE, 2020.

[54] Javier Moyano Cano. Quadrotor UAV for wind profile characterization. Master's thesis, Universidad Carlos III de Madrid. Departamento de Informática, Madrid, Spain, 2013.

[55] Iván Cantador and José M Conde. A simple e-learning system based on classroom competition. In *European Conference on Technology Enhanced Learning*, pages 488–493. Springer, 2010.

[56] Chris Carroll. Competition based learning in the classroom. In *ASEE Annual Conference & Exposition*, pages 23–313, 2013.

[57] Abdulla Chaer, Khaled Salah, Claudio Lima, Pratha Pratim Ray, and Tarek Sheltami. Blockchain for 5G: opportunities and challenges. In *IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2019.

[58] Forrest Chan, Jenny Ling Cheung, and Eve Man Hin Chan. The impact of competition-based learning on enhancing students' motivation, engagement and professionalism: A case study of fashion design undergraduates in hong kong. *Applied Degree Education and the Future of Work*, pages 197–211, 2020.

[59] Baiyun Chen, Kathleen Bastedo, and Wendy Howard. Exploring Design Elements for Online STEM Courses: Active Learning, Engagement & Assessment Design. *Online Learning*, 22(2):59–75, 2018.

[60] CS Chen, Yaqing Hou, and Yew-Soon Ong. A conceptual modeling of flocking-regulated multi-agent reinforcement learning. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 5256–5262. IEEE, 2016.

[61] Eric Cheng. *Aerial photography and videography using drones*. Peachpit Press, 2015.

[62] Qiao Cheng, Xiangke Wang, Jian Yang, and Lincheng Shen. Automated enemy avoidance of unmanned aerial vehicles based on reinforcement learning. *Applied Sciences*, 9(4):669, 2019.

[63] Michelene TH Chi and Ruth Wylie. The icap framework: Linking cognitive engagement to active learning outcomes. *Educational psychologist*, 49(4):219–243, 2014.

[64] Wen-Chyuan Chiang, Yuyu Li, Jennifer Shang, and Timothy L Urban. Impact of drone delivery on sustainability and cost: Realizing the UAV potential through vehicle routing optimization. *Applied energy*, 242:1164–1175, 2019.

[65] Am Cho, Jihoon Kim, Sanghyo Lee, and Changdon Kee. Wind estimation and airspeed calibration using a uav with a single-antenna gps receiver and pitot tube. *IEEE transactions on aerospace and electronic systems*, 47(1):109–117, 2011.

[66] Doo-Hyun Cho, Jung-Su Ha, Sujin Lee, Sunghyun Moon, and Han-Lim Choi. Informative path planning and mapping with multiple UAVs in wind fields. In *Distributed Autonomous Robotic Systems*, pages 269–283. Springer, 2018.

[67] François Chollet et al. Keras. `https://keras.io`, 2015.

[68] Konstantinos Christidis and Michael Devetsikiotis. Blockchains and smart contracts for the internet of things. *Ieee Access*, 4:2292–2303, 2016.

[69] CJ ChanJin Chung, Christopher Cartwright, and Matthew Cole. Assessing the impact of an autonomous robotics competition for stem education. *Journal of STEM Education: Innovations and Research*, 15(2), 2014.

[70] Cisco webex. "https://www.webex.com/". Accessed on 05-20-2021.

[71] Luis C. Cobo, Charles L. Isbell, and Andrea L. Thomaz. Object focused q-learning for autonomous agents. In *International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pages 1061–1068, Richland, SC, 2013. International Foundation for Autonomous Agents and Multiagent Systems.

[72] Iain D Couzin, Jens Krause, Richard James, Graeme D Ruxton, and Nigel R Franks. Collective memory and spatial sorting in animal groups. *Journal of theoretical biology*, 218(1):1–12, 2002.

[73] Fei Dai, Ming Chen, Xianglin Wei, and Huibin Wang. Swarm intelligence-inspired autonomous flocking control in uav networks. *IEEE Access*, 7:61786–61796, 2019.

[74] Mike Dalton. EOS pursuing compatibility with ethereum smart contracts. https://cryptobriefing.com/eos-pursuing-compatibility-ethereum-smart-contract-code/. Accessed on: 07-07-2020.

[75] Konstantin Danilov, Ruslan Rezin, Ilya Afanasyev, and Alexander Kolotov. Towards blockchain-based Robonomics: autonomous agents behavior validation. In *International Conference on Intelligent Systems (IS)*, pages 222–227. IEEE, 2018.

[76] Ian De Boisblanc, Nikita Dodbele, Lee Kussmann, Rahul Mukherji, Doug Chestnut, Stephanie Phelps, Gregory C Lewin, and Stephan de Wekker. Designing a hexacopter for the collection of atmospheric flow data. In *2014 Systems and Information Engineering Design Symposium (SIEDS)*, pages 147–152. IEEE, 2014.

[77] Lobo Deepan. Smart contract code. https://github.com/radlab-sketch/Ethereum. Accessed on 10-15-2021.

[78] RAD-Lab Deepan Lobo. Arduino and python code to interface with ultrasonic anemometer using uart. https://github.com/DeepanL/RAD-Lab, 2019.

[79] RAD-Lab Deepan Lobo. Implementation of dryden wind turbulence model in python. https://github.com/radlab-sketch/drydenModelPython, 2020.

[80] Nasrin Dehbozorgi. Active learning design patterns for cs education. In *ACM Conference on International Computing Education Research*, pages 291–292, 2017.

[81] Melih Derya GURER, Ibrahim Cetin, and Ercan Top. Factors affecting students' attitudes toward computer programming. *Informatics in Education*, 18(2):281–296, 2019.

[82] Ashay Desai, Michael Tippins, and JB Arbaugh. Learning through collaboration and competition: Incorporating problem-based learning and competition-based learning in a capstone course. *Organization Management Journal*, 11(4):258–271, 2014.

[83] Decagon Devices. DS-2 sonic anemometer operator's manual. "http://manuals.decagon.com/Manuals/14586_DS2_Web.pdf", 2017. Accessed on 01-26-2020.

[84] Using Hyperledger Fabric to set up Ethereum smart contracts. https://www.devteam.space/blog/how-to-deploy-smart-contract-on-ethereum/. Accessed on: 7-7-2020.

[85] Shawn Dexter. Ethereum difficulty bomb explained. https://www.mangoresearch.co/ethereum-difficulty-bomb-explained/. Accessed on: 05-12-2020.

[86] Guidelines for Remotely Piloted Aircraft System (RPAS). "https://www.dgca.gov.in/digigov-portal/". Accessed on 05-20-2021.

[87] DJI Matrice M100: Quadcopter for Developers. https://www.dji.com/matrice100. Accessed on: 10-24-2019.

[88] Matrice 100 TB48D Battery. https://store.dji.com/product/matrice-100-tb48d-battery. Accessed on 07-07-2020.

[89] Atilla Dogan, Timothy A Lewis, and William Blake. Flight data analysis and simulation of wind effects during aerial refueling. *Journal of Aircraft*, 45(6):2036–2048, 2008.

[90] Geoffrey W Donnell, Jordan A Feight, Nate Lannan, and Jamey D Jacob. Wind characterization using onboard IMU of SUAS. In *Atmospheric Flight Mechanics Conference*, page 2986, 2018.

[91] Digital differential airspeed sensor kit. https://store.drotek.com/digital-differential-airspeed-sensor-kit-. Accessed on 01-26-2020.

[92] Fabio D'Urso, Corrado Santoro, and Federico Fausto Santoro. An integrated framework for the realistic simulation of multi-uav applications. *Computers & Electrical Engineering*, 74:196–209, 2019.

[93] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*, 2020.

[94] Jason Etele and Giovanni Fusina. *Overview of wind gust modelling with application to autonomous low-level UAV control*. Defence R&D Canada, 2006.

[95] Proof of Stake. https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ. Accessed on: 05-12-2020.

[96] Ethereum Virtual Machine. https://github.com/ethereum/wiki/wiki/Ethereum-Virtual-Machine-(EVM)-Awesome-List. Accessed on: 05-12-2020.

[97] Ethereum Homestead documentation. https://ethdocs.org/. Accessed on: 07-07-2020.

[98] Civil drones (Unmanned aircraft). "https://www.easa.europa.eu/domains/civil-drones-rpas". Accessed on 05-20-2021.

[99] F450 frame. Accessed: 09-24-2021.

[100] FAA. Certificated Remote Pilots including Commercial Operators. "https://www.faa.gov/uas/commercial_operators/". Accessed on 05-20-2021.

[101] Educational users. https://www.faa.gov/uas/educational_users/. Accessed on 07-07-2020.

[102] FAA. Unmanned aircraft systems: forecast. https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/unmanned_aircraft_systems.pdf, 2019. Accessed on: 24/01/2020.

[103] Veronica Farr and Gerri Light. Integrated stem helps drone education fly. In *IEEE Integrated STEM Education Conference (ISEC)*, pages 398–401. IEEE, 2019.

[104] Miguel Fernandes and Luís A Alexandre. Robotchain: using Tezos technology for robot event management. *Ledger*, 4, 2019.

[105] Tiago M Fernández-Caramés, Oscar Blanco-Novoa, Iván Froiz-Míguez, and Paula Fraga-Lamas. Towards an autonomous industry 4.0 warehouse: a UAV and blockchain-based system for inventory and traceability applications in big Data-driven supply chain management. *Sensors*, 19(10):2394, 2019.

[106] Eduardo Castelló Ferrer. The blockchain: A new framework for robotic swarm systems. In *Proceedings of the Future Technologies Conference*, pages 1037–1058. Springer, 2018.

[107] Scott Freeman, Sarah L Eddy, Miles McDonough, Michelle K Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*, 111(23):8410–8415, 2014.

[108] Stacey Gage. Creating a unified graphical wind turbulence model from multiple specifications. In *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit*, page 5529, 2003.

[109] Keke Gai, Yulu Wu, Liehuang Zhu, Lei Xu, and Yan Zhang. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet of Things Journal*, 6(5):7992–8004, 2019.

[110] David Galway, J Etele, and Giovanni Fusina. Modeling of urban wind field effects on unmanned rotorcraft flight. *Journal of Aircraft*, 48(5):1613–1620, 2011.

[111] Azad Ghaffari. Operational safety control for unmanned aerial vehicles using modular barrier functions. In *American Control Conference (ACC)*, pages 1719–1724. IEEE, 2020.

[112] Javier Gonzalez-Rocha, Craig A Woolsey, Cornel Sultan, Stephan de Wekker, and Nathan Rose. Measuring atmospheric winds from quadrotor motion. In *AIAA Atmospheric Flight Mechanics Conference*, page 1189, 2017.

[113] P Gousseau, Bert Blocken, and GJF Van Heijst. Cfd simulation of pollutant dispersion around isolated buildings: On the role of convective and turbulent mass fluxes in the prediction accuracy. *Journal of Hazardous Materials*, 194:422–434, 2011.

[114] Getting a drone pilot certificate. `"https://tc.canada.ca/en/aviation/drone-safety/getting-drone-pilot-certificate"`. Accessed on 05-20-2021.

[115] Akshay A Gowardhan, Eric R Pardyjak, Inanc Senocak, and Michael J Brown. A CFD-based wind solver for an urban fast response transport and dispersion model. *Environmental fluid mechanics*, 11(5):439–464, 2011.

[116] Luigi Alfredo Grieco, Alessandro Rizzo, Simona Colucci, Sabrina Sicari, Giuseppe Piro, Donato Di Paola, and Gennaro Boggia. IoT-aided robotics applications: technological implications, target domains and open issues. *Computer Communications*, 54:32–47, 2014.

[117] Radhika Grover, Shoba Krishnan, Terry Shoup, and Maryam Khanbaghi. A competition-based approach for undergraduate mechatronics education using the arduino platform. In *Fourth Interdisciplinary Engineering Design Education Conference*, pages 78–83. IEEE, 2014.

[118] Yuying Guo, Bin Jiang, and Youmin Zhang. A novel robust attitude control for quadrotor aircraft subject to actuator faults and wind gusts. *IEEE/CAA Journal of Automatica sinica*, 5(1):292–300, 2017.

[119] Patrick Y Haas, Christophe Balistreri, Piero Pontelandolfo, Gilles Triscone, Hasret Pekoz, and Antonio Pignatiello. Development of an unmanned aerial vehicle UAV for air quality measurement in urban areas. In *32nd AIAA Applied Aerodynamics Conference*, page 2272, 2014.

[120] Hackaday. `https://hackaday.com/`. Accessed on 05-20-2021.

[121] Edward A Haering Jr. Airdata measurement and calibration. Technical Report 104316, NASA, Cleveland, OH 44135, 1995.

[122] Teuku Mohd Ichwanul Hakim and Ony Arifianto. Implementation of dryden continuous turbulence model into simulink for lsa-02 flight test simulation. *Journal of Physics: Conference Series*, 1005(1):012017, 2018.

[123] Laya Harwin and P Supriya. Comparison of sarsa algorithm and temporal difference learning algorithm for robotic path planning for static obstacles. In *Third International Conference on Inventive Systems and Control (ICISC)*, pages 472–476. IEEE, 2019.

[124] Haya R Hasan and Khaled Salah. Proof of delivery of digital assets using blockchain and smart contracts. *IEEE Access*, 6:65439–65448, 2018.

[125] Haya R Hasan and Khaled Salah. Combating deepfake videos using blockchain and smart contracts. *Ieee Access*, 7:41596–41606, 2019.

[126] Shahab Hasan. Urban air mobility (UAM) market study. Technical report, NASA, 2019.

[127] Sabine Hauert, Severin Leven, Maja Varga, Fabio Ruini, Angelo Cangelosi, Jean-Christophe Zufferey, and Dario Floreano. Reynolds flocking in reality with fixed-wing robots: Communication range vs. maximum turning rate. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5015–5020, 2011.

[128] Chenlong He, Zuren Feng, and Zhigang Ren. A flocking algorithm for multi-agent systems with connectivity preservation under hybrid metric-topological interactions. *PloS one*, 13(2), 2018.

[129] Jianming He and Charles CS Song. Evaluation of pedestrian winds in urban area by numerical approach. *Journal of Wind Engineering and Industrial Aerodynamics*, 81(1-3):295–309, 1999.

[130] Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

[131] Holybro air speed sensor. `https://www.getfpv.com/holybro-air-speed-sensor.html`. Accessed on 01-26-2020.

[132] Alistair John George Howard. *Experimental Characterization and Simulation of a Tethered Aerostat with Controllable Tail Fins*. PhD thesis, McGill University, 2007.

[133] Dapeng Huang, Quan Yuan, and Xiang Li. Decentralized flocking of multi-agent system based on mpc with obstacle/collision avoidance. In *Chinese Control Conference (CCC)*, pages 5587–5592. IEEE, 2019.

[134] Gwo-Jen Hwang and Shao-Chen Chang. Effects of a peer competition-based mobile learning approach on students' affective domain exhibition in social studies courses. *British Journal of Educational Technology*, 47(6):1217–1231, 2016.

[135] Morgan M Hynes and Wendy J Hynes. If you build it, will they come? student preferences for makerspace environments in higher education. *International Journal of Technology and Design Education*, 28(3):867–883, 2018.

[136] Free gas networks. `https://besu.hyperledger.org/en/stable/HowTo/Configure/FreeGas`. Accessed on: 07-07-2020.

[137] Hyperledger Fabric documentaion. `https://wiki.hyperledger.org/display/fabric/Hyperledger+Fabric`. Accessed on: 10-24-2019.

[138] Jung Hyun, Ruth Ediger, and Donghun Lee. Students' satisfaction on their learning process in active learning and traditional classrooms. *International Journal of Teaching and Learning in Higher Education*, 29(1):108–118, 2017.

[139] Brian Ichter, Edward Schmerling, Tsang-Wei Edward Lee, and Aleksandra Faust. Learned critical probabilistic roadmaps for robotic motion planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9535–9541. IEEE, 2020.

[140] Instructables. `"https://www.instructables.com/"`. Accessed on 05-20-2021.

[141] Instructure. Canvas Learning Management System. `"https://www.instructure.com/canvas"`. Accessed on 05-20-2021.

[142] Gill Instruments. Working principle of gill instrument wind sensor. `http://gillinstruments.com/products/anemometer/principleofoperation2.html`, 2016.

[143] Ghassan Issa, Shakir Hussain, and H. Al-Bahadili. Competition-based learning: A model for the integration of competitions with project-based learning using open source lms. *International Journal of Information and Communication Technology Education*, 10:1–13, 01 2014.

[144] Sébastien Jacques, Sébastien Bissey, and Arnaud Martin. Multidisciplinary project based learning within a collaborative framework: a case study on urban drone conception. *International Journal of Emerging Technologies in Learning (iJET)*, 11(12):36–44, 2016.

[145] Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, and Jong Wook Kim. Q-learning algorithms: A comprehensive classification and applications. *IEEE Access*, 7:133653–133667, 2019.

[146] Darryl Jenkins and Bijan Vasigh. *The economic impact of unmanned aircraft systems integration in the United States*. Association for Unmanned Vehicle Systems International (AUVSI), 2013. Accessed on 08-06-2021.

[147] JetBrains. PyCharm. "https://www.jetbrains.com/pycharm/". Accessed on 05-20-2021.

[148] JMAVSim. Accessed: 09-24-2021.

[149] Vickie Johnston and David Collum. Understanding diversity and the educational needs of students with exceptionalities: a case study of simschool. *International Journal of Smart Technology and Learning*, 2(2-3):198–216, 2020.

[150] Roy Jonker and Ton Volgenant. Improving the hungarian assignment algorithm. *Operations Research Letters*, 5(4):171–175, 1986.

[151] Corina Joseph, Mariam Rahmat, et al. Exploring the perception on using the competition based learning winning video as a pedagogical tool in an accounting course. *International Journal of Academic Research in Business and Social Sciences*, 8(2):440–451, 2018.

[152] Carme Julià and Juan Òscar Antolí. Impact of implementing a long-term STEM-based active learning course on students' motivation. *International Journal of Technology and Design Education*, 29(2):303–327, 2019.

[153] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, and Y. Zhang. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet of Things Journal*, pages 1–1, 2019.

[154] sUAS Commercial Remote Pilot Training . https://polytechnic.k-state.edu/academics/degree-options/uas-flight/. Accessed on 05-20-2021.

[155] Savvas Kapartis. Anemometer employing standing wave normal to fluid flow and travelling wave normal to standing wave, March 1999. US Patent 5,877,416.

[156] Aleksandr Kapitonov, Sergey Lonshakov, Aleksandr Krupenkin, and Ivan Berman. Blockchain-based protocol of autonomous business activity for multi-agent systems consisting of uavs. In *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, pages 84–89. IEEE, 2017.

[157] Ian Kar. Estonian citizens will soon have the world's most hack-proof health-care records. https://qz.com/628889/this-eastern-european-country-is-moving-its-health-records\-to-the-blockchain/. Accessed on: 10-24-2019.

[158] J Kelly and A Williams. Forty big banks test blockchain-based bond trading system. https://www.reuters.com/article/banking-blockchain-bonds/forty-big-banks-test-blockchain-based-bond-trading-system-\\idUSL8N16A30H. Accessed on: 10-24-2019.

[159] Chaker Abdelaziz Kerrache, Ezedin Barka, Nasreddine Lagraa, and Abderrahmane Lakas. Reputation-aware energy-efficient solution for FANET monitoring. In *10th IFIP Wireless and Mobile Networking Conference (WMNC)*, pages 1–6. IEEE, 2017.

[160] Tobias Kessler and Alois Knoll. Multi vehicle trajectory coordination for automated parking. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 661–666. IEEE, 2017.

[161] Minhaj Ahmad Khan and Khaled Salah. IoT security: review, blockchain solutions, and open challenges. *Future Generation Computer Systems*, 82:395–411, 2018.

[162] Kashish Khullar. Implementing PBFT in blockchain. https://medium.com/coinmonks/implementing-pbft-in-blockchain-$12368c6c9548$. Accessed on: 05-12-2020.

[163] Kittyhawk: Drone ops & airspace management. https://play.google.com/store/apps/details?id=io.kittyhawk. Accessed on 05-20-2021.

[164] Frank Klassner and Scott D Anderson. Lego mindstorms: Not just for k-12 anymore. *IEEE robotics & automation magazine*, 10(2):12–18, 2003.

[165] Finley Klint. Python Is More Popular Than Ever.

[166] Rex Klopfenstein Jr. Air velocity and flow measurement using a pitot tube. *ISA transactions*, 37(4):257–263, 1998.

[167] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

[168] Georgios Konstantopoulos. Understanding blockchain fundamentals. https://medium.com/loom-network/understanding-blockchain-fundamentals-part-2-proof-of-work-\\proof-of-stake-$b6ae907c7edb$. Accessed on: 05-12-2020.

[169] Daria Kotys-Schwartz, Mary Besterfield-Sacre, and Larry Shuman. Informal learning in engineering education: Where we are—where we need to go. In *Frontiers in Education Conference (FIE)*, pages T4J–1. IEEE, 2011.

[170] Cezary Kownacki and Daniel Ołdziej. Flocking algorithm for fixed-wing unmanned aerial vehicles. In *Advances in Aerospace Guidance, Navigation and Control*, pages 415–431. Springer, 2015.

[171] Raghavendra V Kulkarni and Ganesh Kumar Venayagamoorthy. Particle swarm optimization in wireless-sensor networks: A brief survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(2):262–267, 2010.

[172] Shriyanti Kulkarni, Vedashree Chaphekar, Md Moin Uddin Chowdhury, Fatih Erden, and Ismail Guvenc. Uav aided search and rescue operation using reinforcement learning. *arXiv preprint arXiv:2002.08415*, 2020.

[173] Yoshiaki Kuwata, Tom Schouwenaars, Arthur Richards, and Jonathan How. Robust constrained receding horizon control for trajectory planning. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 6079, 2005.

[174] Youngho Kwon and Jun Hwang. Mathematical modeling for flocking flight of autonomous multi-uav system, including environmental factors. *KSII Transactions on Internet & Information Systems*, 14(2), 2020.

[175] Juraj Labovskỳ and L'udovít Jelemenskỳ. CFD-based atmospheric dispersion modeling in real urban environments. *Chemical Papers*, 67(12):1495–1503, 2013.

[176] Protocol Labs. IPFS Documentation. `https://docs.ipfs.io/`, 2018. Accessed on: 10-24-2019.

[177] S Lacey. The energy blockchain: how Bitcoin could be a catalyst for the distributed grid. `https://www.greentechmedia.com/articles/read/the-energy-blockchain-could-bitcoin-be-a-catalyst\-for-the-distributed-grid`.

[178] Jack W Langelaan, Nicholas Alley, and James Neidhoefer. Wind Field Estimation for Small Unmanned Aerial Vehicles. *Journal of Guidance, Control, and Dynamics*, 34(4):1016–1030, 2011.

[179] Mathieu Le Goc, Lawrence H Kim, Ali Parsaei, Jean-Daniel Fekete, Pierre Dragicevic, and Sean Follmer. Zooids: building blocks for swarm user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 97–109. ACM, 2016.

[180] Hansoo Lee, Jonggeun Kim, Sungshin Kim, and Jaeyong Kim. A comparative study on air data system calibration methods based on machine learning. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6. IEEE, 2019.

[181] Jon Lee and Sven Leyffer. *Mixed integer nonlinear programming*, volume 154. Springer Science & Business Media, 2011.

[182] Karsten Lensing, Benedikt Schwuchow, Siegmar Oehlandt, and Tobias Haertel. How makerspaces help to participate in technology: results of a survey to gain data

about learners' activities in makerspaces. In *World Engineering Education Forum-Global Engineering Deans Council (WEEF-GEDC)*, pages 1–5. IEEE, 2018.

[183] Marko Lepetič, Gregor Klančar, Igor Škrjanc, Drago Matko, and Boštjan Potočnik. Time optimal path planning considering acceleration limits. *Robotics and Autonomous Systems*, 45(3-4):199–210, 2003.

[184] Philippa Levy and Robert Petrulis. How do first-year university students experience inquiry and research, and what are the implications for the practice of inquiry-based learning? *Studies in higher education*, 37(1):85–101, 2012.

[185] Jon A Leydens, Barbara M Moskal, and Michael J Pavelich. Qualitative methods used in the assessment of engineering education. *Journal of Engineering Education*, 93(1):65–72, 2004.

[186] Bohao Li and Yunjie Wu. Path planning for uav ground target tracking via deep reinforcement learning. *IEEE Access*, 8:29064–29074, 2020.

[187] Ting Li and Ying Li. A novel path planning algorithm based on q-learning and adaptive exploration strategy. *Electrical Engineering and Computer Science (EECS)*, 3:105–108, 2019.

[188] Hongjing Liao and John Hitchcock. Reported credibility techniques in higher education evaluation studies that use qualitative methods: A research synthesis. *Evaluation and program planning*, 68:157–165, 2018.

[189] Leo Liberti, Sebastian Sager, and Angelika Wiegele. Mixed-integer nonlinear optimization: a hatchery for modern mathematics. *Oberwolfach Reports*, 16(2):1573–1637, 2020.

[190] F Adhika Pradipta Lie and Demoz Gebre-Egziabher. Synthetic air data system. *Journal of Aircraft*, 50(4):1234–1249, 2013.

[191] Chao Lin, Debiao He, Neeraj Kumar, Xinyi Huang, Pandi Vijaykumar, and Kim-Kwang Raymond Choo. Homechain: A blockchain-based secure mutual authentication system for smart homes. *IEEE Internet of Things Journal*, 2019.

[192] Lars Lindemann and Dimos V Dimarogonas. Control barrier functions for multi-agent systems under conflicting local signal temporal logic tasks. *IEEE control systems letters*, 3(3):757–762, 2019.

[193] Lars Lindemann and Dimos V Dimarogonas. Barrier function based collaborative control of multiple robots under signal temporal logic tasks. *IEEE Transactions on Control of Network Systems*, 7(4):1916–1928, 2020.

[194] Linksys *EA*9500 Max-Stream™ AC5400 MU-MIMO Gigabit WiFi router. `https://www.linksys.com/us/p/$P-EA9500$/`. Accessed on 05-12-2020.

[195] Vincenzo Lippiello, Bruno Siciliano, and Luigi Villani. Position-based visual serving in industrial multi-robot cells using a hybrid camera configuration. *IEEE Transactions on Robotics*, 23(1):73–86, 2007.

[196] Alex Lishinski, Aman Yadav, and Richard Enbody. Students' emotional reactions to programming projects in introduction to programming: Measurement approach and influence on learning outcomes. In *ACM Conference on International Computing Education Research*, pages 30–38, 2017.

[197] Sikang Liu, Kartik Mohta, Nikolay Atanasov, and Vijay Kumar. Towards Search-based Motion Planning for Micro Aerial Vehicles. In *International Conference on Robotics and Automation (ICRA)*, May 2019.

[198] Xiao Liu, Yuanwei Liu, and Yue Chen. Reinforcement learning in multiple-uav networks: Deployment and movement design. *IEEE Transactions on Vehicular Technology*, 68(8):8036–8049, 2019.

[199] Deepan Lobo, Drashti Patel, Jorim Morainvile, Prateek Shekhar, and Pramod Abichandani. Preparing students for drone careers using active learning instruction. *IEEE Access*, 9:126216–126230, 2021.

[200] Vasco Lopes, Luís A Alexandre, and Nuno Pereira. Controlling robots using artificial intelligence and a consortium blockchain. *arXiv preprint arXiv:1903.00660*, 2019.

[201] Chen Lu, Rongbing Li, Jianye Liu, and Tingwan Lei. Calculation method for air data based on information from inertial navigation system and wind field. In *IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, pages 1845–1850. IEEE, 2016.

[202] Vladimir J. Lumelsky and KR Harinarayan. Decentralized motion planning for multiple mobile robots: The cocktail party model. *Autonomous Robots*, 4(1):121–135, 1997.

[203] Jia Luo, Qianbin Chen, F Richard Yu, and Lun Tang. Blockchain-enabled software-defined industrial internet of things with deep reinforcement learning. *IEEE Internet of Things Journal*, 2020.

[204] Wei Luo, Qirong Tang, Changhong Fu, and Peter Eberhard. Deep-sarsa based multi-uav path planning and obstacle avoidance in a dynamic environment. In *International Conference on Sensing and Imaging*, pages 102–111. Springer, 2018.

[205] Yang Lyu, Jinwen Hu, Ben M Chen, Chunhui Zhao, and Quan Pan. Multivehicle flocking with collision avoidance via distributed model predictive control. *IEEE transactions on cybernetics*, 2019.

[206] RW Macdonald. Modelling the mean velocity profile in the urban canopy layer. *Boundary-Layer Meteorology*, 97(1):25–45, 2000.

[207] Sarah P Madruga, Augusto HBM Tavares, Saulo OD Luiz, Tiago P do Nascimento, and Antonio Marcus N Lima. Aerodynamic effects compensation on multi-rotor uavs based on a neural network control allocation approach. *IEEE/CAA Journal of Automatica Sinica*, 9(2):295–312, 2021.

[208] Alejandra J Magana, Camilo Vieira, and Mireille Boutin. Characterizing engineering learners' preferences for active and passive learning methods. *IEEE Transactions on Education*, 61(1):46–54, 2017.

[209] Robert Mahony, Vijay Kumar, and Peter Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics and Automation magazine*, 19(3):20–32, 2012.

[210] Sidra Malik, Volkan Dedeoglu, Salil S Kanhere, and Raja Jurdak. TrustChain: Trust management in blockchain and IoT supported supply chains. In *IEEE International Conference on Blockchain (Blockchain)*, pages 184–193. IEEE, 2019.

[211] John R Maltby and Jan Whittle. Learning programming online: Student perceptions and performance. In *Proceedings of the ASCILITE Conference*. Citeseer, 2000.

[212] C Stoica Maniu, Cristina Vlad, Thomas Chevet, Gauthier Rousseau, Sylvain Bertrand, and Sorin Olaru. Modernizing teaching through experimentation on uavs formations. *IFAC-PapersOnLine*, 52(9):144–146, 2019.

[213] Cristina Stoica Maniu, Cristina Vlad, Thomas Chevet, Sylvain Bertrand, Antonello Venturino, Gauthier Rousseau, and Sorin Olaru. Control systems engineering made easy: Motivating students through experimentation on uavs. In *Preprints of the 21st IFAC World Congress*, 2020.

[214] Matthew Marino, Alex Fisher, Reece Clothier, Simon Watkins, S Prudden, and Chung Sing Leung. An evaluation of multi-rotor unmanned aircraft as flying wind sensors. *International Journal of Micro Air Vehicles*, 7(3):285–299, 2015.

[215] Daniele Marioli, Claudio Narduzzi, Carlo Offelli, Dario Petri, Emilio Sardini, and Andrea Taroni. Digital time-of-flight measurement for ultrasonic sensors. *IEEE Transactions on Instrumentation and Measurement*, 41(1):93–97, 1992.

[216] Philippe Martin and Erwan Salaün. Generalized multiplicative extended kalman filter for aided attitude and heading reference system. In *AIAA Guidance, Navigation, and Control Conference*, page 8300, 2010.

[217] Catherine Massé, Olivier Gougeon, Nguyen Tien, and David Saussié. Modeling and control of a quadcopter flying in a wind field: A comparison between lqr and structured h infinity control techniques. In *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1408–1417, June 2018.

[218] M Anwar Ma'sum, M Kholid Arrofi, Grafika Jati, Futuhal Arifin, M Nanda Kurniawan, Petrus Mursanto, and Wisnu Jatmiko. Simulation of intelligent unmanned aerial vehicle UAV for military surveillance. In *International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pages 161–166. IEEE, 2013.

[219] Eric S McDaniel, Saleh M Alnaeli, David A Juckem, and Warren S Vaz. Motivating undergraduate computing and engineering research via educational and scientific drones. In *IEEE International Conference on Electro Information Technology (EIT)*, pages 536–541. IEEE, 2019.

[220] Mark A McDaniel, Kathleen M Wildman, and Janis L Anderson. Using quizzes to enhance summative-assessment performance in a web-based class: An experimental study. *Journal of Applied Research in Memory and Cognition*, 1(1):18–26, 2012.

[221] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *Proc. of Int. Conf. on Robotics and Automation*, pages 2520–2525, May 2011.

[222] D. Mellinger, A. Kushleyev, and V. Kumar. Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *Proceedings of the International Conference on Robotic and Automation*, May 2012.

[223] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *IEEE International Conference on Robotics and Automation*, pages 2520–2525. IEEE, May 2011.

[224] Yan MENG and Kazuki HIBI. Turbulent measurments of the flow field around a high-rise building. *Wind Engineers, JAWE*, 1998(76):55–64, 1998.

[225] Francisco Javier Mesas-Carrascosa, Fernando Pérez Porras, Paula Triviño-Tarradas, Jose Emilio Meroño de Larriva, and Alfonso García-Ferrer. Project-based learning applied to unmanned aerial systems and remote sensing. *Remote Sensing*, 11(20):2413, 2019.

[226] Microsoft. Microsoft Visual Studio. `"https://visualstudio.microsoft.com/"`. Accessed on 05-20-2021.

[227] US Military Handbook MIL-HDBK. Flying qualities of piloted aircraft, 1997.

[228] US Military Handbook MIL-HDBK. Flying qualities of piloted vehicles. Technical report, MIL-STD-1797A, 1990.

[229] Julie E Mills, David F Treagust, et al. Engineering education—Is problem-based or project-based learning the answer. *Australasian journal of engineering education*, 3(2):2–16, 2003.

[230] Anastasia Misseyanni, Paraskevi Papadopoulou, Christina Marouli, and Miltiadis D Lytras. *Active learning strategies in higher education*. Emerald Publishing Limited, 2018.

[231] Alex Mizrahi. A blockchain-based property ownership recording system. `https://www.paperdue.com/essay/blockchain-based-property-ownership-recording-$2166424$`, 2015. Accessed on: 07-07-2020.

[232] Kartik Mohta, Matthew Turpin, Alex Kushleyev, Daniel Mellinger, Nathan Michael, and Vijay Kumar. QuadCloud: A rapid response force with quadrotor teams. In *Experimental Robotics*, pages 577–590. Springer, 2016.

[233] Amr Mokhtar, Noel Murphy, and Jennifer Bruton. Blockchain-based multi-robot path planning. In *IEEE 5th World Forum on Internet of Things (WF-IoT)*, pages 584–589. IEEE, 2019.

[234] Saikat Mondal, Kanishka P Wijewardena, Saranraj Karuppuswami, Nitya Kriti, Deepak Kumar, and Premjeet Chahal. Blockchain inspired RFID-based information architecture for food supply chain. *IEEE Internet of Things Journal*, 6(3):5803–5813, 2019.

[235] Moodle. Moodle Open Source Learning Management System. `"https://moodle.org/"`. Accessed on 05-20-2021.

[236] David J Moorhouse and Robert J Woodcock. Background information and user guide for MIL-F-8785C, military specification-flying qualities of piloted airplanes. Technical report, Air Force Wright Aeronautical Labs Wright-Patterson AFB OH, 1982.

[237] C Murphy and R Bergman. Legally operating a drone in the agriculture industry. *Iowa State University*, 2020.

[238] Craig WA Murray, M Ireland, and David Anderson. On the response of an autonomous quadrotor operating in a turbulent urban environment. In *AUVSI's unmanned systems conference*, 2014.

[239] NASA Glenn Research Center. Pitot static tube. `https://www.grc.nasa.gov/www/k-12/airplane/pitot.html`.

[240] Guidelines for the operations of remotely piloted aircraft systems/unmanned aerial vehicle (RPAS/UAV) in Nigeria. `https://ncaa.gov.ng/media-center/news/guidelines-for-the-operations-of-remotely-piloted-aircraft-\\systems-unmanned-aerial-vehicle-rpas-uav-in-nigeria/`. Accessed on 05-20-2021.

[241] Patrick Neumann and Matthias Bartholmai. Real-time wind estimation on a micro unmanned aerial vehicle using its inertial measurement unit. *Sensors and Actuators A: Physical*, 235:300–310, November 2015.

[242] Amir Niaraki, Jeremy Roghair, and Ali Jannesari. Energy-aware goal selection and path planning of uav systems via reinforcement learning. *arXiv preprint arXiv:1909.12217*, 2019.

[243] Amir Niaraki, Jeremy Roghair, and Ali Jannesari. Visual exploration and energy-aware path planning via reinforcement learning. *arXiv preprint arXiv:1909.12217*, 2019.

[244] Noboru Noguchi, Jeff Will, John Reid, and Qin Zhang. Development of a master-slave robot system for farm operations. *Computers and Electronics in agriculture*, 44(1):1–19, 2004.

[245] Helmi Norman, Norazah Nordin, Mohamed Amin Embi, Hafiz Zaini, and Mohamed Ally. A framework of drone-based learning (dronagogy) for higher education in the fourth industrial revolution. *International Journal of Engineering & Technology*, 7(3.14):1–6, 2018.

[246] AN Ohood, O Almelhi, A Almarzooqi, AM Abdullah, S Sayadi, and S Issacniwas. Persistent surveillance with small unmanned aerial vehicles (suav): a feasibility study. In *Proc. SPIE*, volume 10796, 2018.

[247] Tim R Oke. Street design and urban canopy layer climate. *Energy and buildings*, 11(1-3):103–113, 1988.

[248] Don Oparah. Ways that the blockchain will change the real estate market.

[249] Proof of Authority chains. `https://openethereum.github.io/wiki/Proof-of-Authority-Chains`. Accessed on: 05-12-2020.

[250] Opencv. `"https://opencv.org/"`. Accessed on 05-20-2021.

[251] OpenPGP. `https://www.openpgp.org/`. Accessed on: 05-12-2020.

[252] Orbitty carrier for NVIDIA Jetson TX2. `http://connecttech.com/product/orbitty-carrier-for-nvidia-jetson-tx$2$-tx$1$/`. Accessed on: 05-12-2020.

[253] Otter Voice Meeting Notes. `https://otter.ai/`. Accessed on 05-20-2021.

[254] Daniel J Pack, Robert Avanzato, David J Ahlgren, and Igor M Verner. Fire-fighting mobile robotics and interdisciplinary design-comparative perspectives. *IEEE Transactions on education*, 47(3):369–376, 2004.

[255] Ross T Palomaki, Nathan T Rose, Michael van den Bossche, Thomas J Sherman, and Stephan FJ De Wekker. Wind estimation in the lower atmosphere using multirotor aircraft. *Journal of Atmospheric and Oceanic Technology*, 34(5):1183–1191, 2017.

[256] Cohn Pamela, Green Alastair, Langstaff Meredith, and Roller Melanie. Commercial drones are here: The future of unmanned aerial systems. `https://www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/commercial-drones-are-here-the-future-of-unmanned-aerial-systems`, 2017. Accessed on 10-24-2019.

[257] Aleksandr I Panov, Konstantin S Yakovlev, and Roman Suvorov. Grid path planning with deep reinforcement learning: Preliminary results. *Procedia computer science*, 123:347–353, 2018.

[258] Venkatasubramani S.R. Pappu, Yande Liu, Joseph Francis Horn, and Jared Cooper. Wind gust estimation on a small VTOL UAV. In *7th AHS Technical Meeting on VTOL Unmanned Aircraft Systems and Autonomy*, 1 2017.

[259] Jay Patrikar, Vishal Dugar, Vaibhav Arcot, and Sebastian Scherer. Real-time motion planning of curvature continuous trajectories for urban uav operations in wind. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 854–861. IEEE, 2020.

[260] Michael Quinn Patton. *Qualitative research & evaluation methods: Integrating theory and practice*. Thousand Oaks, CA: Sage, 2015.

[261] Liam Paull, Jacopo Tani, Heejin Ahn, Javier Alonso-Mora, Luca Carlone, Michal Cap, Yu Fan Chen, Changhyun Choi, Jeff Dusek, Yajun Fang, et al. Duckietown: an open, inexpensive and flexible platform for autonomy education and research. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1497–1504. IEEE, 2017.

[262] Project based learning works. `"https://my.pblworks.org/"`. Accessed on 05-20-2021.

[263] Gabriele Perozzi, Denis Efimov, Jean-Marc Biannic, Laurent Planckaert, and Patricia Coton. Wind estimation algorithm for quadrotors using detailed aerodynamic coefficients. In *2018 Annual American Control Conference (ACC)*, pages 1921–1926. IEEE, 2018.

[264] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225. IEEE, 2006.

[265] Jan Petrich and Kamesh Subbarao. On-board wind speed estimation for uavs. In *AIAA Guidance, Navigation, and Control Conference*, page 6223, 2011.

[266] Pawit Pharpatara, Bruno Hérissé, and Yasmina Bestaoui. 3-d trajectory planning of aerial vehicles using rrt. *IEEE Transactions on Control Systems Technology*, 25(3):1116–1123, 2016.

[267] Carlo Pinciroli, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews, Eliseo Ferrante, Gianni Di Caro, Frederick Ducatelle, Mauro Birattari, Luca Maria Gambardella, and Marco Dorigo. ARGoS: A modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, 6(4):271–295, 2012.

[268] Marzia Pisciotta, Bruno Vello, Claudio Bordo, and Giovanna Morgavi. Robotic competition: A classroom experience in a vocational school. In *Actas del 6th International Conference on Educational Technologies*, pages 151–156, 2010.

[269] Michael Prince. Does active learning work? a review of the research. *Journal of engineering education*, 93(3):223–231, 2004.

[270] Sam Prudden, Alex Fisher, Matthew Marino, Abdulghani Mohamed, Simon Watkins, and Graham Wild. Measuring wind with small unmanned aircraft systems. *Journal of Wind Engineering and Industrial Aerodynamics*, 176:197–210, 2018.

[271] Vedant Prusty and Andrea Rossetti. *Modelling Geometries and Simulation of Fluid Flow in Airlaids for Virtual Material Design.* PhD thesis, Universitätsbibliothek der RWTH Aachen, 2019.

[272] Qground Control. Accessed: 09-24-2021.

[273] Huaxin Qiu and Haibin Duan. A multi-objective pigeon-inspired optimization approach to uav distributed flocking among obstacles. *Information Sciences*, 509:515–529, 2020.

[274] Xiang Li Quan Yuan, Jingyuan Zhan. Outdoor flocking of quadcopter drones with decentralized model predictive control. *ISA Transactions*, 71(1):84–92, November 2017.

[275] Afaf Izzati Nafhah Radzi, Dayang Nazari Awang Drahman, Corina Joseph, Mariam Rahmat, and Khadijah Suria. Competition-based learning strategy of the online introductory accounting quiz for non-accounting majors. *International Business Education Journal*, 13(1):83–94, May 2020.

[276] Paulina Lyubenova Raeva, Jaroslav Šedina, and Adam Dlesk. Monitoring of crop fields using multispectral and thermal imagery from uav. *European Journal of Remote Sensing*, 52(sup1):192–201, 2019.

[277] Shankarachary Ragi and Hans D Mittelmann. Mixed-integer nonlinear programming formulation of a UAV path optimization problem. In *American Control Conference (ACC)*, pages 406–411. IEEE, 2017.

[278] Shirin Rahmanpour and Reza Mahboobi Esfanjani. Energy-aware planning of motion and communication strategies for networked mobile robots. *Information Sciences*, 497:149–164, 2019.

[279] Gunasekaran Raja, Sudha Anbalagan, Vikraman Sathiya Narayanan, Srinivas Jayaram, and Aishwarya Ganapathisubramaniyan. Inter-uav collision avoidance using deep-q-learning in flocking environment. In *IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 1089–1095. IEEE, 2019.

[280] Ellen M Rathje and Kevin Franke. Remote sensing for geotechnical earthquake reconnaissance. *Soil Dynamics and Earthquake Engineering*, 91:304–316, 2016.

[281] Syed Ali Raza. *Autonomous UAV Control for Low-Altitude Flight in an Urban Gust Environment*. PhD thesis, Carleton University, 2015.

[282] Fred Reichheld. *The ultimate question 2.0 (revised and expanded edition): How net promoter companies thrive in a customer-driven world*. Harvard Business Review Press, 2011.

[283] Frederick Reichheld and Stephen R Covey. *The ultimate question: Driving good profits and true growth*, volume 211. Harvard Business School Press Boston, 2006.

[284] Goldman Sachs Research. Drones: reporting for work. `https://www.goldmansachs.com/insights/technology-driving-innovation/drones/`, 2020. Accessed on: 10-24-2019.

[285] Craig W Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

[286] Adriano MC Rezende, Vinicius M Gonçalves, and Luciano CA Pimenta. Safe coordination of robots in cyclic paths. *ISA Transactions*, 2020.

[287] Matthew B Rhudy, Yu Gu, and Haiyang Chao. Wind field velocity and acceleration estimation using a small uav. In *AIAA Modeling and Simulation Technologies Conference*, page 2647, 2014.

[288] Matthew B Rhudy, Yu Gu, Jason N Gross, and Haiyang Chao. Onboard wind velocity estimation comparison for unmanned aircraft systems. *IEEE Transactions on Aerospace and Electronic Systems*, 53(1):55–66, 2017.

[289] Matthew B Rhudy, Trenton Larrabee, Haiyang Chao, Yu Gu, and Marcello Napolitano. Uav attitude, heading, and wind estimation using gps/ins and an air data system. In *AIAA guidance, navigation, and control (GNC) conference*, page 5201, 2013.

[290] Levick Richard. Drone industry just beginning to take off. `https://www.forbes.com/sites/richardlevick/$2018$/$05$/$15$/drone-industry-just-beginning-to-take-off/\#$7323272472$bc`, 2018. Accessed on: 01-26-2020.

[291] Arthur Richards and Jonathan P How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, volume 3, pages 1936–1941. IEEE, 2002.

[292] Johnhenri R Richardson. *Quantifying and Scaling Airplane Performance in Turbulence.* PhD thesis, University of Michigan, 2013.

[293] Charles Richter, Adam Bry, and Nicholas Roy. *Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments*, pages 649–666. Springer International Publishing, Cham, 2016.

[294] Alexander Robey, Haimin Hu, Lars Lindemann, Hanwen Zhang, Dimos V Dimarogonas, Stephen Tu, and Nikolai Matni. Learning control barrier functions from expert demonstrations. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 3717–3724. IEEE, 2020.

[295] Andres Rodriguez, Evan Andersen, Justin Bradley, and Clark Taylor. Wind estimation using an optical flow sensor on a miniature air vehicle. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, page 6614, 2007.

[296] Leopoldo Rodriguez, Jose A Cobano, and Aníbal Ollero. Wind field estimation and identification having shear wind and discrete gusts features with a small uas. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5638–5644. IEEE, 2016.

[297] RSK Documentation. `https://developers.rsk.co/rsk/`. Accessed on: 07-07-2020.

[298] Stuart J Russell and Andrew Zimdars. Q-decomposition for reinforcement learning agents. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 656–663, 2003.

[299] Jae Ryu, Sonja LaPaglia, and Riveraine Walters. Idaho drone league (idrone) to stimulate stem workforce. *Journal of STEM Education: Innovations and Research*, 21(2), 2020.

[300] Mohammad H Sadraey. Unmanned aerial vehicles design education; techniques and challenges. In *ASEE Virtual Annual Conference Content Access*, 2020.

[301] Khaled Salah, M Habib Ur Rehman, Nishara Nizamuddin, and Ala Al-Fuqaha. Blockchain for ai: Review and open research challenges. *IEEE Access*, 7:10127–10149, 2019.

[302] Johnny Saldaña. *The coding manual for qualitative researchers.* Sage, 2015.

[303] Student assessment of their learning gains. `https://salgsite.net/`. Accessed on 05-20-2021.

[304] Salim Mohamed Salim, Riccardo Buccolieri, Andrew Chan, and Silvana Di Sabatino. Numerical simulation of atmospheric pollutant dispersion in an urban street canyon: Comparison between rans and les. *Journal of Wind Engineering and Industrial Aerodynamics*, 99(2-3):103–113, 2011.

[305] Juan Sandino, Fernando Vanegas, Felipe Gonzalez, and Frederic Maire. Autonomous uav navigation for active perception of targets in uncertain and cluttered environments. In *IEEE Aerospace Conference*, pages 1–12. IEEE, 2020.

[306] Guillaume Sartoretti, Yue Wu, William Paivine, TK Satish Kumar, Sven Koenig, and Howie Choset. Distributed reinforcement learning for multi-robot decentralized collective construction. In *Distributed autonomous robotic systems*, pages 35–49. Springer, 2019.

[307] Fabrizio Schiano, Javier Alonso-Mora, Konrad Rudin, Paul Beardsley, Roland Y Siegwart, and Bruno Sicilianok. Towards estimation and correction of wind effects on a quadrotor UAV. In *Proceedings of the International Micro Air Vehicle (IMAV) Conference and Competition*, pages 134–141, 2014.

[308] Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J Pappas. Anytime planning for decentralized multirobot active information gathering. *IEEE Robotics and Automation Letters*, 3(2):1025–1032, 2018.

[309] Tom Schouwenaars, Bart De Moor, Eric Feron, and Jonathan How. Mixed integer programming for multi-vehicle path planning. In *European control conference (ECC)*, pages 2603–2608. IEEE, 2001.

[310] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[311] AHFN Segundo, JS da C Neto, RF da Silva, HA de Oliveira, L de A Marques, and LS Bernardo. Pbl in the development and stability control of an uav with interdisciplinarity between digital control and computer assisted design. *International Journal of Advanced Engineering Research and Science*, 6(6):539–543, 2019.

[312] Shannon B Seidel and Kimberly D Tanner. "What if students revolt?"—considering student resistance: origins, options, and opportunities for investigation. *CBE—Life Sciences Education*, 12(4):586–595, 2013.

[313] Differential pressure sensors sdp3x for drones. `https://www.sensirion.com/en/flow-sensors/differential-pressure-sensors/airspeed-sensor-revolutionizes-workflow-with-px4-vtol-drones/`. Accessed on 01-26-2020.

[314] Sensirion sdp3x airspeed sensor kit (sdp33). `https://store.drotek.com/sdp3x-airspeed-sensor-kit-sdp33`. Accessed on 01-26-2020.

[315] Tina Setter and Magnus Egerstedt. Energy-constrained coordination of multi-robot teams. *IEEE Transactions on Control Systems Technology*, 25(4):1257–1263, 2016.

[316] David F Shanno and Robert J Vanderbei. Interior-point methods for nonconvex nonlinear programming: orderings and higher-order methods. *Mathematical Programming*, 87(2):303–316, 2000.

[317] Prateek Shekhar, Maura Borrego, Matt DeMonbrun, Cynthia Finelli, Caroline Crockett, and Kevin Nguyen. Negative student response to active learning in STEM classrooms: A systematic review of underlying reasons. *Journal of College Science Teaching*, 49(6):45–54, 2020.

[318] Prateek Shekhar, Michael Prince, Cynthia Finelli, Matt Demonbrun, and Cynthia Waters. Integrating quantitative and qualitative research methods to examine student resistance to active learning. *European Journal of Engineering Education*, 44(1-2):6–18, 2019.

[319] Meng Shen, Xiangyun Tang, Liehuang Zhu, Xiaojiang Du, and Mohsen Guizani. Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities. *IEEE Internet of Things Journal*, 6(5):7702–7712, 2019.

[320] Weisen Shi, Junling Li, Nan Cheng, Feng Lyu, Shan Zhang, Haibo Zhou, and Xuemin Shen. Multi-drone 3-D trajectory planning and scheduling in drone-assisted radio access networks. *IEEE Transactions on Vehicular Technology*, 68(8):8145–8158, 2019.

[321] Tomoya Shimura, Minoru Inoue, Hirofumi Tsujimoto, Kansuke Sasaki, and Masato Iguchi. Estimation of wind vector profile using a hexarotor unmanned aerial vehicle and its application to meteorological observation up to 1000m above surface. *Journal of Atmospheric and Oceanic Technology*, 35(8):1621–1631, 2018.

[322] Valentyn N Sichkar. Reinforcement learning algorithms in global path planning for mobile robot. In *International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, pages 1–5. IEEE, 2019.

[323] L. N. C. Sikkel, Guido C. H. E. de Croon, C. De Wagter, and Qiping Chu. A novel online model-based wind estimation approach for quadrotor micro air vehicles using low cost MEMS IMUs. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2141–2146, 2016.

[324] Magdalena Simma. Measuring wind using the internal stabilisation system of a quadrotor drone. Master's thesis, UiT Norges arktiske universitet, 2018.

[325] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38(3):287–308, 2000.

[326] Solidworks. "https://www.solidworks.com/". Accessed on 05-20-2021.

[327] Yao Song, Qing-Hao Meng, Bing Luo, Ming Zeng, Shu-Gen Ma, and Pei-Feng Qi. A wind estimation method for quadrotors using inertial measurement units. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 426–431, 2016.

[328] Yeongho Song, Joonwon Choi, Hyondong Oh, Min Lee, Seunghan Lim, and Jaemoon Lee. Improvement of decentralized flocking flight efficiency of fixed-wing uavs using inactive agents. In *AIAA Scitech 2019 Forum*, page 0391, 2019.

[329] Yunlong Song, Selim Naji, Elia Kaufmann, Antonio Loquercio, and Davide Scaramuzza. Flightmare: A flexible quadrotor simulator. *arXiv preprint arXiv:2009.00563*, 2020.

[330] Christian Speck and Donald J Bucci. Distributed uav swarm formation control via object-focused, multi-objective sarsa. In *Annual American Control Conference (ACC)*, pages 6596–6601. IEEE, 2018.

[331] Nathan Sprague and Dana Ballard. Multiple-goal reinforcement learning with modular sarsa(o). In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 1445–1447, 2003.

[332] Spyder. Spyder. "https://www.spyder-ide.org/". Accessed on 05-20-2021.

[333] Mark William Starnes. *The study of an acoustic resonance anemometer*. PhD thesis, Imperial College London, 2010.

[334] Vahram Stepanyan and Kalmanje S Krishnakumar. Estimation, navigation and control of multi-rotor drones in an urban wind field. In *AIAA Information Systems-AIAA Infotech@ Aerospace*, page 0670. AIAA SciTech, 2017.

[335] Andrei Alexandru Stoica. Emerging legal issues regarding civilian drone usage. *Challenges of the Knowledge Society*, pages 692–699, 2018.

[336] Volker Strobel, Eduardo Castelló Ferrer, and Marco Dorigo. Managing byzantine robots via blockchain technology in a swarm robotics collective decision making scenario. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 541–549. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[337] William Suberg. Factom's latest partnership takes on US healthcare. https://cointelegraph.com/news/factoms-latest-partnership-takes-on-us-healthcare. Accessed on: 10-24-2019.

[338] Ahmed Suliman, Zainab Husain, Menatallah Abououf, Mansoor Alblooshi, and Khaled Salah. Monetization of IoT data using smart contracts. *IET Networks*, 8(1):32–37, 2018.

[339] Jian Sun and Yingzhou Zhang. A reinforcement learning-based decentralized method of avoiding multi-uav collision in 3-d airspace. In *Proceedings of the 3rd International Conference on Computer Science and Artificial Intelligence*, pages 77–82, 2019.

[340] Ning Sun, Yiming Wu, He Chen, and Yongchun Fang. An energy-optimal solution for transportation control of cranes with double pendulum dynamics: Design and experiments. *Mechanical Systems and Signal Processing*, 102:87–101, 2018.

[341] Yaping Sun, Zhaojing Wang, Housheng Su, and Tao Geng. A brief overview of flocking control for multi-agent systems. In *International Conference on Intelligent Robotics and Applications*, pages 48–58. Springer, 2018.

[342] Mark Sutherland. *Urban wake field generation using LES for application to quadrotor flight*. PhD thesis, Carleton University, 2015.

[343] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[344] Nitin Sydney, Brendan Smyth, and Derek A Paley. Dynamic control of autonomous quadrotor flight in an estimated wind field. In *52nd IEEE Conference on Decision and Control*, pages 3609–3616. IEEE, 2013.

[345] Michael Sytsma and Larry Ukeiley. Low order turbulence modeling methods for MAVs flight environment. In *Proceedings of the AIAA Atmospheric Flight Mechanics Conference*, page 7935, 2010.

[346] Pooya Taheri, Philip Robbins, and Sirine Maalej. Makerspaces in first-year engineering education. *Education Sciences*, 10(1):8, 2020.

[347] Tetsuro Tamura. Large eddy simulation on building aerodynamics. In *Proceedings of the seventh Asia-Pacific Conference on Wind Engineering*, pages 131–157, 2009.

[348] FT Technologies. Acu-res technlogy, 2016.

[349] Sneha Tharayil, Maura Borrego, Michael Prince, Kevin A Nguyen, Prateek Shekhar, Cynthia J Finelli, and Cynthia Waters. Strategies to mitigate student resistance to active learning. *International Journal of STEM Education*, 5(1):1–16, 2018.

[350] Amila Thibbotuwawa, Grzegorz Bocewicz, Grzegorz Radzki, Peter Nielsen, and Zbigniew Banaszak. UAV mission planning resistant to weather uncertainty. *Sensors*, 20(2):515, 2020.

[351] Pengzhi Tian and Haiyang Chao. Model aided estimation of angle of attack, sideslip angle, and 3D wind without flow angle measurements. In *AIAA Guidance, Navigation, and Control Conference*, page 1844, 2018.

[352] Tinker CAD. Accessed: 09-24-2021.

[353] Yoshihide Tominaga and Ted Stathopoulos. CFD modeling of pollution dispersion in a street canyon: Comparison between LES and RANS. *Journal of Wind Engineering and Industrial Aerodynamics*, 99(4):340–348, 2011.

[354] AC1750 Wireless Dual Band Gigabit router. `https://www.tp-link.com/us/home-networking/wifi-router/archer-c7/`. Accessed on 05-12-2020.

[355] Nguyen Khoi Tran. Modeling and control of a quadrotor in a wind field. Master's thesis, McGill University, Montreal, Quebec, December 2015.

[356] Nguyen Khoi Tran, Eitan Bulka, and Meyer Nahon. Quadrotor control in a wind field. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 320–328. IEEE, 2015.

[357] Angelo Trotta, Leonardo Montecchiari, Marco Di Felice, and Luciano Bononi. A gps-free flocking model for aerial mesh deployments in disaster-recovery scenarios. *IEEE Access*, 8:91558–91573, 2020.

[358] Roberto G Valenti, Yong-Dian Jian, Kai Ni, and Jizhong Xiao. An autonomous flyer photographer. In *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 273–278. IEEE, 2016.

[359] Gabriele Valentini, Davide Brambilla, Heiko Hamann, and Marco Dorigo. Collective perception of environmental features in a robot swarm. In *International Conference on Swarm Intelligence*, pages 65–76. Springer, 2016.

[360] Gábor Vásárhelyi, Csaba Virágh, Gergő Somorjai, Tamás Nepusz, Agoston E Eiben, and Tamás Vicsek. Optimized flocking of autonomous drones in confined environments. *Science Robotics*, 3(20), 2018.

[361] Omar Velasco and João Valente. Online drone education, a mapping review. In *IEEE Global Engineering Education Conference (EDUCON)*, pages 1286–1289. IEEE, 2020.

[362] Franklin Wireless Corp Ellipsis. `https://www.verizon.com/internet-devices/verizon-ellipsis-jetpack-$mhs900l$/`. Accessed on: 05-12-2020.

[363] Tamás Vicsek, András Czirók, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet. Novel type of phase transition in a system of self-driven particles. *Physical review letters*, 75(6):1226, 1995.

[364] Solovyev Viktor, I Finaev Valery, A Zargaryan Yuri, O Shapovalov Igor, and A Beloglazov Denis. Simulation of wind effect on a quadrotor flight. *ARPN Journal of Engineering and Applied Sciences*, 10(4):1535–1538, 2015.

[365] Patrik Voštinár, Dana Horváthová, and Nika Klimová. The programmable drone for stem education. In *International Conference on Entertainment Computing*, pages 205–210. Springer, 2018.

[366] A Walton and AYS Cheng. Large-eddy simulation of pollution dispersion in an urban street canyon—part ii: idealised canyon simulation. *Atmospheric Environment*, 36(22):3615–3627, 2002.

[367] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. Ieee, 2000.

[368] Chang Wang, Chao Yan, Xiaojia Xiang, and Han Zhou. A continuous actor-critic reinforcement learning approach to flocking with fixed-wing uavs. In *Asian Conference on Machine Learning*, pages 64–79, 2019.

[369] Chao Wang, Jian Wang, and Xudong Zhang. A deep reinforcement learning approach to flocking and navigation of uavs in large-scale complex environments. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1228–1232. IEEE, 2018.

[370] Hao Wang, Jie Duan, Maoli Wang, Jingbo Zhao, and Zhenzhen Dong. Research on robot path planning based on fuzzy neural network algorithm. In *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 1800–1803. IEEE, 2018.

[371] Jia-Ying Wang, Bing Luo, Ming Zeng, and Qing-Hao Meng. A wind estimation method with an unmanned rotorcraft for environmental monitoring tasks. *Sensors*, 18(12):4504, 2018.

[372] Li Wang, Aaron D Ames, and Magnus Egerstedt. Safe certificate-based maneuvers for teams of quadrotors using differential flatness. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3293–3298. IEEE, 2017.

[373] Tong Wang, Ricardo M Lima, Loïc Giraldi, and Omar M Knio. Trajectory planning for autonomous underwater vehicles in the presence of obstacles and a nonlinear flow field using mixed integer nonlinear programming. *Computers & Operations Research*, 101:55–75, 2019.

[374] John Ware and Nicholas Roy. An analysis of wind field estimation and exploitation for quadrotor flight in the urban canopy layer. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1507–1514. IEEE, 2016.

[375] Yazz Warsame, Stefan Edelkamp, and Erion Plaku. Energy-aware multi-goal motion planning guided by monte carlo search. In *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pages 335–342. IEEE, 2020.

[376] Steven Waslander and Carlos Wang. Wind disturbance estimation and rejection for quadrotor position control. In *Proceedings of the AIAA Infotech, Aerospace Conference*, page 1983, 2009.

[377] Jonathan P West, Casey A Klofstad, Joseph E Uscinski, and Jennifer M Connolly. Citizen support for domestic drone use and regulation. *American Politics Research*, 47(1):119–151, 2019.

[378] Grant P Wiggins, Grant Wiggins, and Jay McTighe. *Understanding by design*. Ascd, 2005.

[379] Carl A Wolf, Richard P Hardis, Steven D Woodrum, Richard S Galan, Hunter S Wichelt, Michael C Metzger, Nicola Bezzo, Gregory C Lewin, and Stephan FJ de Wekker. Wind data collection techniques on a multi-rotor platform. In *2017 Systems and Information Engineering Design Symposium (SIEDS)*, pages 32–37. IEEE, 2017.

[380] Gavin Wood. Ethereum yellow paper. `https://github.com/ethereum/yellowpaper,[Oct.30,2018]`, 2014. Accessed on: 10-24-2019.

[381] Xingyu Xiang, Zhonghai Wang, Zijian Mo, Genshe Chen, Khanh Pham, and Erik Blasch. Wind field estimation through autonomous quadcopter avionics. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6. IEEE, 2016.

[382] Bin Xu and Koushil Sreenath. Safe teleoperation of dynamic UAVs through control barrier functions. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7848–7855. IEEE, 2018.

[383] Jie Xu, Kaiping Xue, Shaohua Li, Hangyu Tian, Jianan Hong, Peilin Hong, and Nenghai Yu. Healthchain: a blockchain-based privacy preserving scheme for large-scale health data. *IEEE Internet of Things Journal*, 6(5):8770–8781, 2019.

[384] Aman Yadav, Dipendra Subedi, Mary A Lundeberg, and Charles F Bunting. Problem-based learning: Influence on students' learning in an electrical engineering course. *Journal of Engineering Education*, 100(2):253–280, 2011.

[385] Peng Yan, Chengchao Bai, Hongxing Zheng, and Jifeng Guo. Flocking control of uav swarms with deep reinforcement leaming approach. In *3rd International Conference on Unmanned Systems (ICUS)*, pages 592–599. IEEE, 2020.

[386] Zhe Yang, Kan Yang, Lei Lei, Kan Zheng, and Victor CM Leung. Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet of Things Journal*, 6(2):1495–1505, 2018.

[387] Derrick W Yeo, Nitin Sydney, Derek A Paley, and Donald Sofge. Downwash detection and avoidance with small quadrotor helicopters. *Journal of Guidance, Control, and Dynamics*, 40(3):692–701, 2016.

[388] Soumaya Bel Hadj Youssef, Slim Rekhis, and Noureddine Boudriga. A blockchain based secure IoT solution for the dam surveillance. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2019.

[389] Qing Zhang, Jie Wang, Zhengquan Yang, and Zengqiang Chen. High-frequency feedback robust control for flocking of multi-agent system with unknown parameters. *Automatika*, 60(1):28–35, 2019.

[390] Shulong Zhao, Xiangke Wang, Hao Chen, and Yajing Wang. Cooperative path following control of fixed-wing unmanned aerial vehicles with collision avoidance. *Journal of Intelligent & Robotic Systems*, pages 1–13, 2020.

[391] Wei Zhao, Wen Qiu, Taoyang Zhou, Xun Shao, and Xiujun Wang. Sarsa-based trajectory planning of multi-uavs in dense mesh router networks. In *International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–5. IEEE, 2019.

[392] Zongyu Zuo, Cunjia Liu, Qing-Long Han, and Jiawei Song. Unmanned aerial vehicles: Control methods and future challenges. *IEEE/CAA Journal of Automatica Sinica*, 9(99):1–14, 2022.